KPMG

Software and website costs

Handbook

US GAAP

February 2026

kpmg.com/us/en/frv

Contents

New chapter added in this edition: **

Recent changes and longstanding guidance

Software is an essential element of entities' operations and financial reporting, and for many, developing and selling access to software comprises, or is part of, their core operations. The swift and seemingly unrelenting advance of artificial intelligence (AI) has, for many, only increased software's importance and raised new questions about how to account for the use and development of AI software.

On top of new accounting questions stemming from the rapid advance of AI, in September 2025, the FASB amended US GAAP for internal-use software development via ASU 2025-06. The amendments to Subtopic 350-40, the first substantial changes to the internal-use software guidance in more than 25 years, better align it with today's predominantly agile software development methods. The amendments also narrow the gap between a vendor's accounting for (1) software it develops for sale to customers solely via cloud computing arrangements and (2) software it develops to license to customers.

Organized in a Q&A format, with examples and observations to explain key concepts, and updated for this edition in response to the developments outlined above, this book is intended to help you effectively and efficiently apply Subtopics 350-40, 350-50 and 985-20 on internal-use software, website development costs and external-use software, respectively.

Scott Muir

Department of Professional Practice, KPMG LLP

Acknowledgments

This Handbook has been produced by the Department of Professional Practice (DPP) of KPMG LLP in the United States.

We would like to acknowledge the efforts of the main contributors to this Handbook:

P.K. Barot

Scott Muir

We would also like to acknowledge the current and former members of DPP and other KPMG professionals who contributed significantly to this Handbook: Kimber Bascom, Margaret Gonzales, Dave Hermann, Lori Kozen, Lisa Munro, Charlie Noble, Julie Santoro, Ali Vapra, Landon Westerlund.

About this publication

The purpose of this Handbook is to assist you in understanding and applying the following US GAAP Subtopics.

- Subtopic 350-40, Internal-Use Software;
- Subtopic 350-50, Website Development Costs; and
- Subtopic 985-20, Costs of Software to Be Sold, Leased, or Marketed.

This Handbook is intended for use by financial statement preparers and other interested parties with a working knowledge of software development and software implementation, as well as software, cloud computing and related arrangements.

Accounting literature

Unless otherwise stated, references to Subtopic 350-40 include all of the amendments thereto enacted by the following Accounting Standards Updates:

- No. 2015-05, Customer's Accounting for Fees Paid in a Cloud Computing Arrangement
- No. 2016-19, Technical Corrections and Improvements
- No. 2018-15, Customer's Accounting for Implementation Costs Incurred in a Cloud Computing Arrangement That Is a Service Contract
- No. 2025-06, Targeted Improvements to the Accounting for Internal-Use Software

Organization of the text

Each chapter of this Handbook includes excerpts from the *FASB Accounting Standards Codification*® and overviews of the relevant requirements. Our in-depth guidance is explained through Q&As that reflect questions we have considered and are encountering in practice. We include observations and examples to explain key concepts.

Our commentary is referenced to the Codification and to other literature, where applicable. The following are examples:

- 350-40-25-3 is paragraph 25-3 of ASC Subtopic 350-40
- ASU 2025-06.BC11 is paragraph 11 of the basis for conclusions to ASU 2025-06
- FAS 86.47 is paragraph 47 of Statement of Financial Accounting Standards No. 86, Accounting for the Costs of Computer Software to Be Sold, Leased, or Otherwise Marketed
- SOP 98-1.73 is paragraph 73 of AICPA Statement of Position 98-1, Accounting for the Costs of Computer Software Developed or Obtained for Internal Use
- Topic 11.M is SEC Staff Accounting Bulletin Topic 11.M

Interaction with revenue recognition

While this Handbook addresses software vendors' and cloud service providers' accounting for software development and implementation costs, it does not address revenue recognition for software or cloud computing arrangements.

For the revenue recognition requirements of Topic 606 applicable to software and cloud computing arrangements, see KPMG Handbooks, Revenue recognition and Revenue for software and SaaS, and the latest news on KPMG Financial Reporting View.

February 2026 edition

This edition includes new Questions and Examples derived from complexities we have seen arise in practice since the April 2024 edition was released, including those stemming from the rapid adoption of AI. This edition also includes new chapters 3A and 9. Chapter 3A is written as if ASU 2025-06 has been adopted, while chapter 9 addresses transition to that guidance. In contrast, chapter 3 applies to entities that have not yet adopted ASU 2025-06. Although most of the ASC guidance addressed herein is longstanding, interpretations of that guidance, particularly for new and emerging technologies (e.g. AI) and software development methods, continue to evolve and new questions continue to arise. This means that some positions herein may change, and positions on new issues will emerge.

The following symbols are used throughout this Handbook to indicate the types of revisions made to sections, Questions, Examples and other items.

** new item added in this edition

item significantly updated in this edition

Always check KPMG Financial Reporting View for the latest news on financial reporting developments and to ensure you are using the current version of this Handbook.

ASU 2025-06

ASU 2025-06, Intangibles—Goodwill and Other—Internal Use Software (Subtopic 350-40): Targeted Improvements to the Accounting for Internal-Use Software, issued in September 2025, is intended to modernize old internal-use software guidance written in 1998 to adapt to the agile (i.e. iterative and flexible) methodology predominantly used to develop software today. For many entities, it may also have the effect of more closely aligning the accounting for the development of software sold on a SaaS basis with that of software licensed to customers (i.e. external-use software). The *key* amendments of the ASU:

- change the cost capitalization threshold by:
  - eliminating accounting consideration of software project development stages; cost capitalization begins when (1) management has authorized and committed to funding the project and (2) it is 'probable' the project

will be completed and the software used to perform its intended function (the 'probable-to-complete' threshold); and

- enhancing the guidance around the 'probable-to-complete' threshold (given its new prominence) and providing new examples in Subtopic 350-40 to illustrate its application; and

— modify the website development costs guidance by:

- eliminating Subtopic 350-50 and relocating any remaining relevant guidance into Subtopic 350-40; and
- adding a new example.

ASU 2025-06 does *not* change:

— the existing accounting requirements for external-use software (i.e. software to be sold or licensed) development costs;

— *what* internal-use software costs can be capitalized (e.g. data conversion/migration, training and software maintenance costs continue to be expensed as incurred); or

— when internal-use software cost capitalization ceases (i.e. when the software is 'substantially complete and ready for its intended use').

Abbreviations

We use the following abbreviations in this Handbook:

AI	Artificial intelligence
AICPA	American Institute of Certified Public Accountants
AcSEC	AICPA Accounting Standards Executive Committee (now known as the Financial Reporting Executive Committee, or FinREC)
ASC	Accounting Standards Codification
ASU	Accounting Standards Update
CCA	Cloud Computing Arrangement
EITF	Emerging Issues Task Force
IaaS	Infrastructure-as-a-Service
NRV	Net realizable value
PaaS	Platform-as-a-Service
PCS	Post-contract customer support
PP&E	Property, plant and equipment
R&D	Research and development
SaaS	Software-as-a-Service

Key terms and concepts

Appendix A includes all the glossary definitions from ASC sections 350-40-20 and 985-20-20 (Subtopic 350-50 does not include a glossary section).

In addition to the US GAAP definitions, the following terms and concepts not defined in US GAAP span multiple sections and questions in this Handbook.

Key term or concept	Application in this Handbook
Cloud computing arrangement	Throughout this Handbook, the term 'cloud computing arrangement' (CCA) is used. A CCA refers to a 'hosting arrangement' that does not meet the criteria in paragraph 350-40-15-4A – i.e. no license to software is conveyed by the arrangement. Section 2.5 addresses application of the criteria in paragraph 350-40-15-4A. It also refers to IaaS arrangements that do not provide the customer the right to access cloud service provider software. [ASU 2015-05.BC3, ASU 2018-15.BC2]
Cloud service provider	The entity that sells the CCA (i.e. the hosting services) to the customer. The cloud service provider may or may not be a party to implementation activities undertaken by the customer related to the CCA. The customer may undertake those activities itself or engage an unrelated third party to do so.
External-use software	Throughout this Handbook, this term is used to refer to software sold, leased or otherwise marketed (including licensed) by software vendors to customers.
Hosting service fees	The fees paid to the cloud service provider for the hosting services (i.e. access to the cloud-based solution). If the cloud service provider also provides implementation and/or other services, the amount allocable to the hosting services on a relative stand-alone price basis are the hosting service fees (see section 2.7). [350-40-30-4]
Implementation activities	The EITF decided not to define implementation activities in ASU 2018-15. However, while developing the ASU, the EITF and the FASB staff discussed these examples of implementation activities (not exhaustive): — integration (developing interfaces between the hosted software and the company's other systems); — customization of the company's other systems or the hosted software; — configuration, either of the company's other systems or of the hosted software; — installation; — architecture and design; — coding; — testing; — data conversion or migration; — training; and — business process reengineering.

Key term or concept	Application in this Handbook
	These examples were included in Issue Summary No. 1, Supplement No. 1 for EITF Issue No. 17-A, produced for the October 12, 2017 EITF meeting.
Timing of implementation activities	Frequently, CCA implementation activities – whether undertaken by the customer, the cloud-service provider, or a third party – are integral to the customer's intended use of the cloud-based solution such that the customer will not go live using the cloud-based solution for its intended purpose until the implementation activities are complete. In these circumstances, hosting service fees will often be incurred before the implementation activities are complete.
Agile software development	A method of software development designed to be flexible and iterative, rather than (1) heavily pre-planned or rigid and (2) sequential or linear. While there is an overall development objective, agile projects are typically much more lightly planned and completed through a series of shorter time-frame development 'sprints'. Companies have predominantly moved to agile software development methods, or hybrid versions thereof, from more structured development methods. Hybrid agile development projects combine agile elements with more structured project management features. See section 3.2.60 and section 3A.2.10 for additional information about agile software development.
Performance requirements**	Defined in the ASC Master Glossary as 'what an entity needs the software to do (for example, functions or features)'. Expanding thereon, we believe software's performance requirements generally reflect the functions and tasks an entity wants and expects the software to perform.
Module	US GAAP does not define 'module'. However, based on commonly available definitions of the term in relation to software, we believe 'module' is intended to describe a self-contained set of software functionality capable of performing a series of tasks (or procedures or functions) independent of other parts of the software program, which may contain numerous modules. 'Module' is often used interchangeably with 'component', but the two terms may not be synonymous.
Component	US GAAP does not define 'component'. However, 'component' often refers to a related group of software functions, or a set of software code that can be independently added, removed or modified. A software program therefore might have numerous components, and those components may be able to be implemented or disengaged independent of other components. In cloud infrastructure scenarios, a 'service' is a component that can be deployed independently. An example of a service may be a securities pricing service that provides pricing data to the customer.

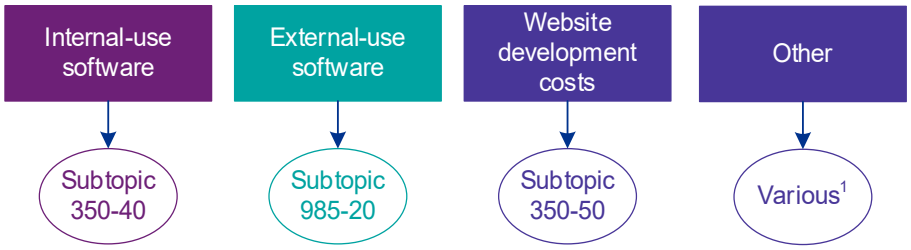
Key term or concept	Application in this Handbook
	'Component' is often used interchangeably with 'module', but the two terms may not be synonymous.
Go-live	US GAAP does not define or use this term. However, it is commonly used when discussing both software and cloud computing arrangements. Go-live generally refers to the point in time that software (hosted or on-premise) is available for users to initiate/process transactions or perform tasks. Generally, this means that testing of the application (or module/component) is complete and that it is in the entity's production environment (i.e. operational for users).
Public cloud	An IT model under which a third-party provider offers access to its infrastructure and/or on-demand computing resources or services over the public Internet.
Private cloud	An IT model under which the third-party provider's resources made available to the customer are not shared with other customers; the provider's services are delivered on a secure, private network.
Hybrid cloud	An IT model that involves a mixture of on-premise, private cloud and/or public cloud resources. For example, in a hybrid cloud arrangement, a customer may obtain a license to, and host, one or more software applications while one or more other applications are accessible only through a private and/or public cloud.

1. Executive summary

New item added in this edition: **
Item significantly updated in this edition: #

Scope#

Multiple US GAAP subtopics address the accounting for software development costs.



[1] For example, Subtopic 340-40 and Subtopic 730-10.

In general:

- Before adopting ASU 2025-06, entities apply Subtopic 350-50 (website development costs) to costs incurred to develop a website. After adopting ASU 2025-06, they apply Subtopic 350-40 to the costs incurred to develop a website. Excluded from the scope of either Subtopic are the costs of hardware and related hardware infrastructure, and the costs to acquire or develop website content. The costs of some types of website content are covered by other Topics (e.g. Topic 926 on film costs), while for others there is no authoritative guidance.
- Subtopic 350-40 applies to costs incurred to acquire or develop software that is solely for the entity's internal use (e.g. ERP or payroll software). This includes software an entity sells access to customers on a SaaS basis – i.e. the customer does not obtain a license to the software because it either (1) does not have the contractual right to take possession of the hosted software or (2) would incur a significant penalty, or it would not be feasible, to do so. Separate sections of Subtopic 350-40 also apply to a customer's costs to implement a cloud-based solution subject to a CCA.
- Subtopic 985-20 applies to costs incurred to develop software to be sold or licensed to third-party customers (hereafter, external-use software), including software that an entity licenses to a customer as part of providing a service.
- Other Topics – e.g. Subtopic 340-40 on costs incurred to fulfill contracts with customers and Subtopic 730-10 on R&D – apply in specific circumstances when required by these Subtopics.

When acquiring, developing and/or implementing software, or acquiring and implementing a cloud-based solution, an entity's first task is to identify the US GAAP guidance that applies to the costs thereof. This is because the guidance

that applies to those costs differs substantially depending on which Topic (or Subtopic), or section within the Subtopic, applies.

For example, the cost guidance in Subtopic 350-40 differs substantially from that in Subtopic 985-20. Meanwhile, the guidance in both of those Subtopics differs from the guidance in the other Topics that may apply. For entities acquiring the right to use software, the guidance in Subtopic 350-40 that applies to software licensing arrangements differs in important respects from the Subtopic 350-40 guidance that applies to CCAs – e.g. how the customer accounts for unpaid license or hosting service fees and the right to use or access the software.

Therefore, inappropriate scoping conclusions can lead to material financial statement errors.

Complex and evolving software transactions and activities complicate the task of identifying the appropriate guidance to apply. For example, the evolution of hybrid cloud arrangements often means:

- a customer is incurring both software license and CCA implementation costs in the scope of the general and CCA implementation cost subsections of Subtopic 350-40, respectively; and
- the vendor's hybrid offering includes (1) software subject to Subtopic 985-20 (for the software that is licensed to customers) and (2) software subject to Subtopic 350-40 (for the software customers access only through the cloud).

Additional challenges arise for traditional software vendors that have historically sold on-premise software licenses, but are transitioning to a cloud-based (e.g. SaaS) model. Significant judgment may be involved in determining:

- when a software product previously subject to Subtopic 985-20 becomes subject to Subtopic 350-40; and
- which Subtopic (i.e. 350-40 or 985-20) applies to new software development (including upgrades and enhancements).

Read more: Chapter 2

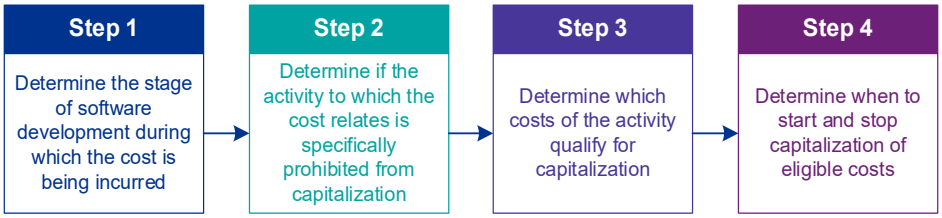
ASU 2025-06's effect on scope**

Subtopic 350-50 heavily leveraged the guidance in Subtopic 350-40 to address accounting for website development costs. ASU 2025-06 eliminates Subtopic 350-50 but incorporates key guidance from Subtopic 350-50 into Subtopic 350-40. It also adds a new example illustrating the application of Subtopic 350-40 to website development. The elimination of Subtopic 350-50 and movement of selected guidance to Subtopic 350-40 are not expected to have any substantive effect on entities' accounting for website development costs.

Initial recognition and measurement

Internal-use software and cloud computing implementation costs

The following steps apply to determine what internal-use software development and implementation costs are capitalized.



The same steps apply regardless of whether an entity is accounting for:

- internally developed internal-use software;
- the development (e.g. customization, modification) or implementation of licensed internal-use software; or
- implementation of a CCA.

Step 1: Costs incurred during the preliminary project and postimplementation-operation stages are expensed as incurred, while costs incurred for development and implementation activities during the application development stage are generally capitalized unless they are expressly required to be expensed as incurred under Step 2.

Step 2: Subtopic 350-40 requires that data conversion/migration and training costs be expensed as incurred, regardless of when they are incurred. Subtopic 720-45 requires business process re-engineering costs, often incurred as part of a software or CCA implementation, to be expensed as incurred.

Step 3: In general, the direct costs of eligible activities are capitalized, while indirect costs (i.e. general and administrative and overhead costs) are not.

Step 4: Subtopic 350-40 contains specific requirements about when cost capitalization can begin and when it must end. Cost capitalization cannot begin before the preliminary project stage is complete, entity management has authorized the project and its funding, and it is probable the project will be completed and used for its intended purpose. Cost capitalization ceases at the earlier of: (1) project abandonment and (2) the point in time when the software (or cloud-based solution) is ready for its intended use (i.e. when all substantial testing is completed).

When an entity acquires an internal-use software license, it capitalizes:

- an intangible asset for the cost of the acquired license (see chapter 7 on presentation); and
- a liability for any fixed and unpaid license fees as of the beginning of the license period.

Capitalized development (e.g. customizations to the license software) and implementation costs become part of the cost-basis of the license intangible asset.

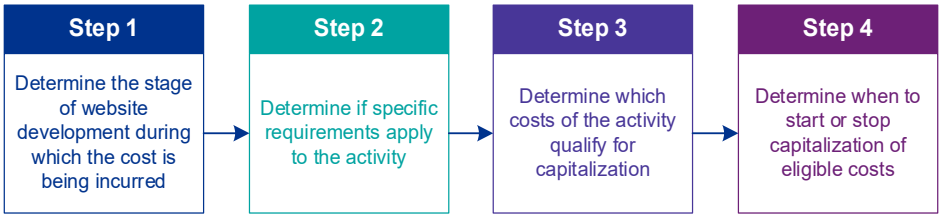
CCAs are executory service contracts. The hosting service fees (i.e. the ongoing subscription fees) due under the CCA are accounted for in the same manner as the entity would account for the fees for other services.

Read more: Chapter 3

Website development costs

Subtopic 350-50 addresses whether website development costs incurred are capitalized or expensed. Capitalizing or expensing depends on the activity and the stage of website development.

The following steps apply to determine what website development costs are capitalized; these steps are broadly consistent with those that apply to internal-use software.



Subtopic 350-50 describes website development activities as occurring in five stages.

- Planning
- Website application and infrastructure development
- Graphics development
- Content development
- Operating.

Subtopic 350-50 requires costs incurred for activities undertaken during the planning and operating stages to be expensed as incurred, while requiring most costs incurred for activities during the other three stages to be capitalized or expensed consistent with the internal-use software guidance in Subtopic 350-40.

The linkage to Subtopic 350-40 means that only direct costs of eligible website development activities are capitalized, and costs of training and inputting, converting or migrating website content are expensed as incurred regardless of the stage of website development.

Third-party website hosting fees are generally expensed as incurred (i.e. as the website hosting services are provided), while costs of hardware (e.g. new servers to host the website) and any other services obtained (e.g. high-speed internet access) are accounted for under other Topics.

Read more: Chapter 4

External-use software

Subtopic 985-20 addresses the accounting for software to be sold, leased or marketed (external-use software). While the title of the subtopic still refers to software that will be 'leased', rights to use software are excluded from the scope of the US GAAP leases guidance (Topic 842) such that 'leased' should be read as 'licensed'.

The following steps apply to determine what external-use software costs should be capitalized.

Subtopic 985-20 requires the expensing of all software development costs incurred to establish the technological feasibility of the software product. Provided there are no unresolved 'high-risk' development issues, technological feasibility of a software product is established upon completion of the product design and either (1) a detailed program design or (2) a working model.

Development costs incurred after technological feasibility is established (production costs) are capitalized to the extent recoverable by the NRV of the software product until the product is available for general release. Unlike under Subtopic 350-40, direct *and* indirect production costs are capitalized under Subtopic 985-20.

Entities apply this same model to costs incurred to develop software product enhancements. A 'product enhancement' is an improvement that extends the life or significantly improves the marketability of the software product.

When developing software to be embedded in a product or process (firmware), capitalization cannot occur before both:

- technological feasibility of the firmware has been established; and
- the other components of the product or process are no longer in the R&D phase.

Software acquired to embed in a product (including a larger external-use software product) or process is expensed at the time of acquisition unless:

- technological feasibility of the software or other product or process with which it will be integrated has been established – in which case the entire cost of the acquired software is capitalized; or
- it has an alternative future use to the entity (e.g. another internal use) – in which case the cost of the acquired software is capitalized to the extent the cost is realizable from the alternative future use and classified in accordance with that alternative future use.

Maintenance and customer support costs are not software development costs. They are generally expensed as incurred, regardless of when incurred – e.g. before or after the software is available for general release.

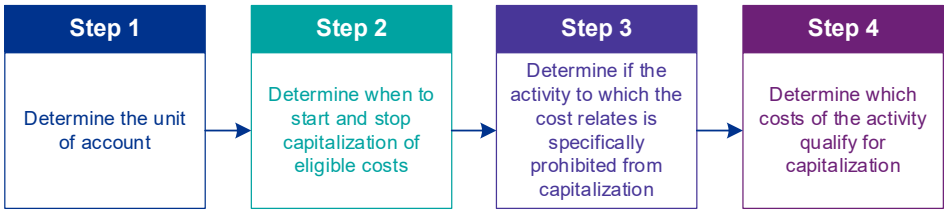
Read more: Chapter 5

ASU 2025-06's effect on initial recognition and measurement**

Removal of project development stages

To better align the guidance with the predominantly agile nature of current software development, ASU 2025-06 eliminates all references to project stages in Subtopic 350-40. This leaves all entities, regardless of software development method (i.e. agile, waterfall, a hybrid of those, or otherwise), to rely solely on the *remaining* criteria in paragraph 350-40-25-12 to determine when development cost capitalization should begin. Those criteria are whether (1) management has authorized and committed to funding the software project and (2) it is probable that the project will be completed and the software used to perform the function(s) intended (the 'probable-to-complete threshold'). Until (1) and (2) are met, all software development costs are expensed as incurred.

The following steps apply to determine what internal-use software development and implementation costs are capitalized.



Probable-to-complete threshold changes

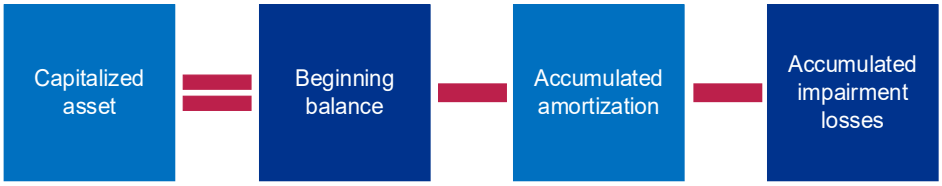
The ASU amends the existing probable-to-complete threshold by adding considerations to the evaluation and providing new examples about its application.

While the threshold itself remains unchanged, the ASU *adds guidance* to state that an entity *does not meet this threshold* if there is 'significant uncertainty' as to the software's development. Significant development uncertainty is assessed for each software project and is considered to exist if either: (1) the software has novel, unique, unproven functions and features or technological innovations that have not yet been proven through coding and testing or (2) the software's significant 'performance requirements' have not yet been defined or remain subject to possible substantial revision.

Read more: Chapter 3A

Subsequent measurement

After initial recognition, software, website development and CCA implementation cost assets capitalized under Subtopics 350-40, 350-50 and 985-20 are all generally measured as follows:



Amortization

Internal-use software and CCA implementation costs

Capitalized internal-use software and CCA implementation costs are generally amortized on a straight-line basis, commencing when the software or cloud-based solution is ready for its intended use (i.e. after all substantial testing is completed).

— Internal-use software is amortized over its estimated useful life, with the restriction that the useful life of an internal-use software license asset cannot exceed the license term. The useful life is reassessed whenever there is a change in the factors, including those specifically provided in Subtopic 350-40, that were important to the entity's existing useful life estimate; in addition, consistent with the general intangible assets guidance in Subtopic 350-30, the useful life should generally be reassessed at least each reporting period for changes in relevant factors.

— CCA implementation cost assets are amortized over the term of the hosting arrangement, which comprises the noncancellable period of the CCA plus any optional renewal periods (1) that are reasonably certain to be exercised by the customer or (2) for which exercise of the option is controlled by the vendor. The term of the hosting arrangement is reassessed if the CCA is modified or the entity exercises a renewal or termination option not already factored into the term. An entity should also reassess the term of the hosting arrangement when there is a change to any one or more of the factors enumerated in Subtopic 350-40 for determining the term of the hosting arrangement.

Any change in the useful life or to the term of the hosting arrangement is accounted for as a change in accounting estimate under Topic 250; see section 3.4 in KPMG Handbook, Accounting changes and error corrections.

Website development costs

Subtopic 350-50 on website development costs does not include amortization guidance.

— Costs capitalized from applying Subtopic 350-40 follow the internal-use software amortization guidance.

— Costs capitalized under other Topics (e.g. computer hardware capitalized under Topic 360) follow the depreciation or amortization guidance in that Topic.

External-use software

Capitalized external-use software costs are amortized over the software's estimated economic life from the point that it is available for general release. Once amortization commences, the *annual* amount of amortization is the greater of:

— the ratio of (1) gross current annual period revenues for the software product as compared to (2) gross current annual period revenues for the software product plus total expected future revenue for the software product (**ratio method**); and

— the amortization that would result for the period from amortizing the software costs on a straight-line basis from the beginning of the annual period over the software product's remaining economic life (**straight-line method**).

An entity should make estimates about the amount of amortization it will recognize for the fiscal year, including the amortization method that will apply (straight-line or ratio), to record interim period amortization.

Abandonment

— When an entity ceases use of internal-use software (whether licensed or internally developed) or a cloud-based solution subject to a CCA, or a module/component thereof, the related asset is accounted for as abandoned.

— Subtopic 350-50 on website development costs does not include abandonment guidance.

  – Costs capitalized from applying Subtopic 350-40 follow the internal-use software abandonment guidance.

  – Costs capitalized under other Topics (e.g. computer hardware capitalized under Topic 360) follow the abandonment guidance, if any, in that Topic.

— Subtopic 985-20 does not include guidance about abandoning a capitalized external-use software asset. However, the practical effect of the 'greater of' amortization model will have the effect, similar to abandonment accounting under Subtopic 350-40, of prospectively accelerating amortization of the external-use software asset to the cease-use date.

Impairment

— Internal-use software, CCA implementation cost assets and any website development costs capitalized from applying Subtopic 350-40 are assessed for impairment under the long-lived assets impairment guidance in Topic 360. Even if an internal-use software asset is part of an asset group that is not impaired under Topic 360, it may need to be written down if it is not yet

completed and it is no longer probable of being completed and placed into service.

— Software capitalized under Subtopic 985-20 is measured at the lower of its amortized carrying amount and NRV on a product-by-product basis at each reporting date.

Read more: Chapter 6

Presentation

Subtopics 350-40, 350-50 and 985-20 provide only limited guidance about financial statement presentation.

Subtopic 350-40	Subtopic 350-50	Subtopic 985-20
— Balance sheet presentation of acquired internal-use software licenses as intangible assets — Balance sheet, income statement and statement of cash flows presentation of CCA implementation costs consistent with how the hosting service fees are presented in those financial statements	— Does not contain presentation guidance, and therefore the financial statement presentation of website development costs is governed by other Topics	— Capitalized software production costs are presented as an amortizable intangible asset — Capitalized software production costs amortization is presented in cost of sales — Follow the presentation requirements of Topic 350 for capitalized production costs

Where those Subtopics do not provide guidance, entities should refer to other Topics that govern financial statement presentation matters.

Read more: Chapter 7

Disclosures

Subtopics 350-40, 350-50 and 985-20 have only limited disclosure requirements.

Subtopic 350-40	Subtopic 350-50	Subtopic 985-20
— Nature of CCAs in which the entity is the customer — Make disclosures as if CCA implementation	— Does not contain disclosure guidance	— Disclose balance of unamortized software production costs for each balance sheet presented

Subtopic 350-40	Subtopic 350-50	Subtopic 985-20
cost assets are a major class of depreciable asset		— Disclose amortization expense and NRV write-down amounts for each income statement period presented — Disclose software R&D costs charged to expense in each income statement period presented — Follow the disclosure requirements in paragraphs 350-30-50-1 – 50-3 for capitalized software production costs

- Subtopic 350-40 refers entities to other Topics for additional disclosure requirements; it only requires the specific disclosures about CCAs outlined above.
- Subtopic 350-50 does not include disclosure guidance. However, entities undertaking website development are still required to make disclosures required by other Topics stemming from website development activities.
- In addition to the specific disclosures outlined above, Subtopic 985-20 includes an example disclosure under Topic 275 (risks and uncertainties) for capitalized software production costs.

Read more: Chapter 8

ASU 2025-06's effect on disclosures**

Subtopic 350-40 expressly states that entities must provide the disclosures required under Subtopic 360-10 (PP&E) for capitalized internal-use software and related amortization, regardless of how the internal-use software is classified on the balance sheet (e.g. as PP&E or an intangible asset) or how it was acquired (e.g. internally developed or licensed from a third party). This may be an important clarification (or change for some) given that *licensed* internal-use software is required to be classified as an acquired intangible asset.

For additional guidance see section 8.2.20.

2. Scope

Detailed contents

New item added in this edition: **
Item significantly updated in this edition: #

Questions

2.4.10 What makes an external marketing plan substantive?

2.4.20 Does devising a substantive plan to market uncompleted internal-use software externally mean it is no longer in the scope of Subtopic 350-40?

2.4.30 Can software that has been marketed externally become internal-use software?

2.4.40 Are thin-client applications internal- or external-use software?

2.4.50 Are freemium apps internal- or external-use software?

2.4.60 Does Subtopic 985-20 apply to third-party owned software an entity obtains the right to market?

2.4.70 What software cost guidance applies to the on-premise software in a hybrid cloud arrangement?

2.4.80 Are costs incurred to migrate internal-use software to a cloud service provider's hosting environment in the scope of Subtopic 350-40?

2.4.90 Are implementation costs incurred to host customer-facing software in another cloud service provider's hosting environment in the scope of Subtopic 350-40?

2.4.100 Can costs be in the scope of both the general and CCA implementation cost guidance in Subtopic 350-40?

2.4.110 Does an acquirer separately account for the embedded software component of a tangible good with embedded software? **

Examples

2.4.10 Substantive marketing plan – mobile app

2.4.20 Not a substantive marketing plan

2.4.30 Software embedded in a product

2.4.40 Software that is part of a process

2.4.50 Mobile gaming app that is free to download

2.4.60 Software vendor accounting for hybrid cloud offering development costs

2.4.70 Customer accounting for hybrid cloud offering development costs

2.5 Hosting arrangements that grant a software license

Questions

2.5.10 Can the customer have a ‘contractual right to take possession of the software at any time during the hosting period’ if such right is not explicit?

2.1 How the standards work#

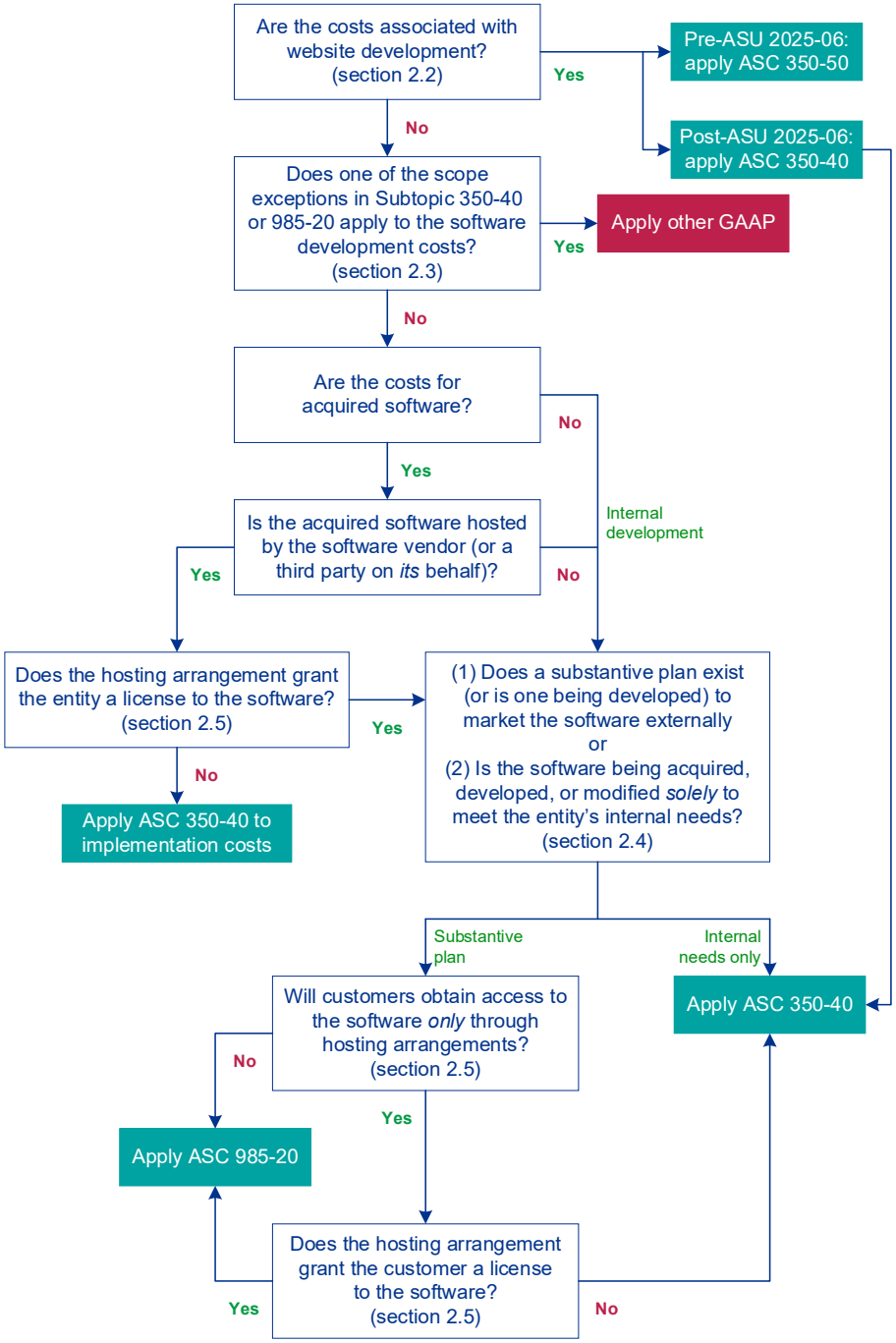
When developing or acquiring software, it is important to identify the US GAAP guidance that applies to the costs of that software. This is because the guidance that applies to those costs differs substantially depending on which Topic (or Subtopic) applies.

For example, the cost guidance in Subtopic 350-40 (internal-use software) differs substantially from that in Subtopic 985-20 (costs of software to be sold, leased or marketed). Meanwhile, the guidance in both of those Subtopics differs from the guidance that may apply if the software is outside the scope of either of those Subtopics – e.g. if an entity's software costs are subject to Subtopic 340-40 (deferred costs – contracts with customers) or Subtopic 730-10 (research and development).

For entities acquiring a right to use software, the guidance in Subtopic 350-40 that applies to software licensing arrangements differs in important respects from the Subtopic 350-40 guidance that applies to CCAs – e.g. how the customer accounts for unpaid license or hosting service fees and the right to use or access the software.

Therefore, inappropriate scoping conclusions can lead to material financial statement errors. Complex and evolving software transactions and activities mean judgment is frequently required to determine the appropriate guidance to apply.

The following flowchart summarizes how to navigate the Topics that can apply to software and website development costs, with references to the appropriate sections in this chapter that discuss the concepts in detail.



2.2 Website development costs

2.2.10 Before adopting ASU 2025-06

Excerpt from ASC 350-50 (pre-ASU 2025-06)

05 Overview and Background

General

05-1 This Subtopic provides guidance on accounting for costs incurred to develop a website, including whether to capitalize or expense the following types of costs:

a. Costs incurred in the planning stage
b. Costs incurred in the website application and infrastructure development stage
c. Costs incurred to develop graphics
d. Costs incurred to develop content
e. Costs incurred in the operating stage.

15 Scope and Scope Exceptions

General

> Overall Guidance

15-1 This Subtopic follows the same Scope and Scope Exceptions as outlined in the Overall Subtopic, see Section 350-10-15, with specific transaction qualifications noted below.

> Transactions

15-2 The guidance in this Subtopic applies to the following transactions and activities:

a. Costs incurred to develop a website.

15-3 The guidance in this Subtopic does not apply to the following transactions and activities:

a. The cost of hardware
b. Acquisitions of servers and related hardware infrastructure.

25 Recognition

General

> Costs Incurred in the Content Development Stage

25-10 Accounting for website content involves issues that also apply to other forms of content or information that are not unique to websites.

Before adopting ASU 2025-06, entities apply Subtopic 350-50 to the following types of costs incurred to develop a website: [350-50-05-1]

— planning stage;
— website application and infrastructure development stage;

— to develop graphics;
— to develop content; and
— operating stage.

Excluded from the scope of Subtopic 350-50 are the costs of hardware (e.g. servers) and related hardware infrastructure. [350-50-15-3]

While the Subtopic includes guidance on certain costs incurred during the 'content development stage' (see section 4.2.40), it does not provide guidance on how to account for the actual content uploaded to the website. The Subtopic merely states that issues related to accounting for content are not unique to *website* content. [350-50-25-10]

2.2.20 After adopting ASU 2025-06**

Excerpt from ASC 350-40 (post-ASU 2025-06)

15 Scope and Scope Exceptions

General

> Transactions

15-2 The guidance in this Subtopic applies to the following transactions and activities:

a. Internal-use software
b. The proceeds of computer software developed or obtained for internal use that is marketed
c. New internal-use software developed or obtained that replaces previously existing internal-use software
d. Computer software that consists of more than one component or module. For example, an entity may develop an accounting software system containing three elements: a general ledger, an accounts payable subledger, and an accounts receivable subledger. In this example, each element might be viewed as a component or module of the entire accounting software system. The guidance in this Subtopic shall be applied to individual components or modules.
e. Costs incurred to develop a website.

ASU 2025-06 eliminates Subtopic 350-50. After adopting ASU 2025-06, entities apply the updated guidance in Subtopic 350-40, which includes selected website development cost guidance migrated from Subtopic 350-50 and a new website development costs example, to all website development costs.

2.2.30 Application issues pre- and post-ASU 2025-06

Question 2.2.10#

How are costs of website content accounted for?

Background: The EITF attempted to address issues pertaining to the accounting for website and other content in Issue No. 00-20 (accounting for costs incurred to acquire or originate information for database content and other collections of information). However, the EITF discontinued its discussion of those issues without reaching a consensus, and neither it, nor the FASB, has engaged in a subsequent project on the topic.

Interpretive response: It depends. US GAAP does not provide specific guidance on how to account for developed or acquired website *content*, so other Topics may apply to certain types of content – e.g. Topic 926 for film content or Topic 928 for music content. An entity follows specific GAAP when it applies.

If specific Topics do not apply, we believe it is appropriate to follow the intangible asset recognition guidance in Subtopic 350-30. Thereunder:

- costs to develop, maintain or restore internally developed content are expensed as incurred; and [350-30-25-3]
- costs of acquired website content are capitalized if they meet (1) either the contractual-legal criterion or the separability criterion for being an 'identifiable' asset or (2) the definition of an asset in FASB Concepts Statement No. 8 (CON 8). [350-30-25-4, CON 8.E16 – E17]

2.3 Subtopic 350-40 and 985-20 scope exceptions

Subtopics 350-40 and 985-20 each exclude specific transactions and activities from their scope, including those in the scope of the other. [350-40-15-4, 985-20-15-2]

This section addresses how to determine if software is outside the scope of *both* of these subtopics.

Section 2.4 addresses how to distinguish internal-use software in the scope of Subtopic 350-40 from external-use software in the scope of Subtopic 985-20.

2.3.10 Subtopic 350-40 scope exceptions

Excerpt from ASC 350-40

15 Scope and Scope Exceptions

General

> Overall Guidance

15-1 The General Subsection of this Section establishes the pervasive scope for this Subtopic. The General Subsections of this Subtopic follow the same Scope and Scope Exceptions as outlined in the Overall Subtopic, see Section 350-10-15, with specific transaction qualifications and exceptions noted below and in the Implementation Costs of a Hosting Arrangement That Is a Service Contract Subsection.

> Transactions

15-4 The guidance in this Subtopic does not apply to the following transactions and activities:

a. Software to be sold, leased, or otherwise marketed as a separate product or as part of a product or process, subject to Subtopic 985-20
b. Software to be used in research and development, subject to Subtopic 730-10
c. Software developed for others under a contractual arrangement, subject to contract accounting standards
d. Accounting for costs of reengineering activities, which often are associated with new or upgraded software applications.

> Other Considerations

15-6 The guidance in this Subtopic does not change any of the provisions in the following Subtopics:

a. Subtopic 985-20
b. Subtopic 720-45.

15-7 The following costs of internal-use computer software are included in research and development and shall be accounted for in accordance with the provisions of Subtopic 730-10:

a. Purchased or leased computer software used in research and development activities where the software does not have alternative future uses
b. All internally developed internal-use computer software (including software developed by third parties, for example, programmer consultants) in either of the following circumstances:
    1. The software is a pilot project (that is, software of a nature similar to a pilot plant as noted in paragraph 730-10-55-1(h)).
    2. The software is used in a particular research and development project, regardless of whether the software has alternative future uses.

Implementation Costs of a Hosting Arrangement That Is a Service Contract

15-8 The Implementation Costs of a Hosting Arrangement That Is a Service Contract Subsections of this Subtopic follow the same Scope and Scope Exceptions as outlined in the General Subsection of this Section, with specific qualifications noted in paragraph 350-40-15-9.

The guidance in Subtopic 350-40 is applied to each individual module or component of internal-use software or cloud-based solution subject to a CCA – e.g. an entity generally determines the appropriate treatment of costs of the fixed asset module separately from the costs of the payroll module of its enterprise resource planning (ERP) system. Therefore, it is possible that different modules or components of a single software application could be scoped differently. [350-40-15-2(d)]

Subtopic 350-40 excludes the following transactions and activities from its scope. [350-40-15-4, 15-7, 730-10-55-1(h)]

- Software to be sold, leased, or otherwise marketed as a separate product or as part of a product or process (external-use software), subject to Subtopic 985-20 (see section 2.4).
- Software to be used in R&D, subject to Subtopic 730-10 (research and development), including:
  - purchased or licensed software to be used in R&D activities that does not have an alternative future use; and
  - internally developed (by the entity or a third party on its behalf) software, regardless of alternative future use, that is a (1) pilot project or (2) for use in a particular R&D project.
- Software developed for others under a contractual arrangement, subject to Topic 606 (revenue from contracts with customers) and Subtopic 340-40 (other assets and deferred costs – contracts with customers).
- Accounting for costs of reengineering activities, which are often associated with new or upgraded software applications, subject to Subtopic 720-45 (business and technology reengineering).

Example 2.3.10

Subtopic 350-40 scope exceptions other than external-use software in the scope of Subtopic 985-20

The following are example transactions and activities that are excluded from the scope of Subtopic 350-40 based on the scope exceptions in paragraph 350-40-15-4(b) – 15-4(d).

Scope exception	Example scenarios
Software to be used in R&D, subject to Subtopic 730-10, including: — purchased or licensed software to be used in R&D and for which the software does not have an alternative future use; and — internally developed internal-use computer software that is a pilot project or will be used in a particular R&D project, regardless of whether the software has an alternative future use.	**Scenario 1** An entity develops software internally to help identify the unique characteristics of a biological compound it is evaluating for use in a pharmaceutical drug candidate. All development costs are expensed as incurred because they are incurred in an R&D activity; there is no consideration of alternative future use. **Scenario 2** Assume the same facts as Scenario 1, except that the entity licenses the software from a third-party software vendor. Unless the software has an alternative future use to the entity, the cost of the software license is expensed as incurred (including any costs to implement the software). **Scenario 3** A bank is attempting to develop a software program that will collate and organize customer transaction data in a particular way. It is initially only implementing the software at a few branches (less than 1% of its total branches) to ascertain whether the core functionality works. If the software works as intended on the test scale, substantial additional development will be needed for it to perform its intended functions across all branches (i.e. at that volume and added complexity). This software development project is a pilot – i.e. it is intended to be deployed to all of the bank's branches and will not be successful unless that occurs. Consequently, the costs are subject to Subtopic 730-10 and expensed as incurred.
Software developed for others under a contractual arrangement, subject to Topic 606 and Subtopic 340-40.	**Scenario 4** An entity (software vendor) enters into an arrangement to develop customized software for a customer with specific needs. The customer will own the software IP upon completion. The arrangement to develop the new software is in the scope of Topic 606, and the related costs are subject to Subtopic 340-40.

Scope exception	Example scenarios
	Scenario 5 An entity (cloud service provider) enters into an arrangement to provide SaaS to a customer. In addition, the entity will develop custom software interfaces for the customer. The interfaces will be used by the customer to connect certain of its on-premise and cloud-based IT applications to the entity's software. The customer will own the IP related to the custom interfaces. The SaaS arrangement, including the interface development services, is in the scope of Topic 606, and the related development costs of the custom interfaces incurred by the entity are subject to Subtopic 340-40. **Scenario 6** Assume the same facts as Scenario 5, except that the arrangement is for the entity (a software vendor) to license its software to a customer. The license arrangement, including the interface development services, is in the scope of Topic 606, and the related development costs of the custom interfaces incurred by the entity are subject to Subtopic 340-40. **Scenario 7** An entity (software vendor) and its customer enter into a license agreement for the vendor's software. As part of the contract, the entity agrees to significantly customize the customer's instance of the software. The arrangement to license and customize the software is in the scope of Topic 606, and the costs incurred by the entity to customize and modify the software for the customer are subject to Subtopic 340-40.
Accounting for costs of reengineering activities, which are often associated with new or upgraded software applications, subject to Subtopic 720-45.	**Scenario 8** An entity (customer) contracts with a software vendor to implement a large-scale, on-premise ERP software system. As part of implementation, the entity incurs costs related to a current state assessment, process reengineering and restructuring the workforce. [720-45-25-2] The costs associated with these reengineering activities are scoped out of Subtopic 350-40 and accounted for under Subtopic 720-45. The costs of the ERP

Scope exception	Example scenarios
	software license and to implement the licensed software are subject to Subtopic 350-40.

Question 2.3.10

Are implementation costs of a cloud-based solution to be used in R&D and without alternative future use expensed as incurred?

Background: Costs of purchased or licensed software to be used in R&D and for which the software does not have an alternative future use are expensed as incurred. [350-40-15-7]

Interpretive response: Yes. Subtopic 350-40 is explicit that the CCA implementation cost guidance is subject to the same scope and scope exceptions as the guidance applicable to purchased or licensed software. This is consistent with the stated intent of the EITF and the FASB for equivalent capitalization treatment for CCA implementation costs and similar costs of licensed internal-use software. [350-40-15-8, ASU 2018-15.BC6]

Question 2.3.20

Are a cloud service provider's costs to implement or provision a customer in a CCA in the scope of Subtopic 350-40?

Background: A cloud service provider may incur costs in connection with a CCA entered into with a customer. For example, the cloud service provider may incur costs to provision the customer and may also incur costs to provide implementation services to the customer.

Interpretive response: No. Costs to implement (or provision) a CCA with a customer are in the scope of Subtopic 340-40. The CCA implementation cost guidance in Subtopic 350-40 is subject to the same scope exceptions as the general Subtopic 350-40 guidance. Costs to provision a CCA customer or provide implementation services to the customer fall under the scope exclusion in paragraph 350-40-15-4(c).

In contrast, any costs incurred by the cloud service provider to upgrade or enhance the hosted software would generally be subject to Subtopic 350-40. This includes any upgrades or enhancements requested by, or enacted for, the customer.

2.3.20 Subtopic 985-20 scope exceptions

Excerpt from ASC 985-20

15 Scope and Scope Exceptions

General

> Overall Guidance

15-1 This Subtopic follows the same Scope and Scope Exceptions as outlined in the Overall Subtopic, see Section 985-10-15, with specific transaction qualifications and exceptions noted below.

> Transactions

15-2 The guidance in this Subtopic applies to the costs, including costs incurred after the date of a business combination or a combination accounted for by a **not-for-profit entity**, of computer software to be sold, leased, or otherwise marketed as a separate product or as part of a product or process, whether internally developed and produced or purchased.

15-3 The guidance in this Subtopic does not apply to the following transactions and activities:

a. Software developed or obtained for internal use (see Subtopic 350-40).
b. Research and development assets acquired in a business combination or an **acquisition by a not-for-profit entity**. If tangible and intangible assets acquired in those combinations are used in research and development activities, they are recognized and measured at fair value in accordance with Subtopic 805-20.
c. Arrangements to deliver software or a software system, either alone or together with other products or services, requiring significant production, modification, or customization of software (see the guidance on costs to fulfill a contract in Subtopic 340-40).

60 Relationships

General

> Research and Development

60-2 For the application of Subtopic 730-10 to costs incurred to purchase or lease computer software developed by others, see paragraph 730-10-25-3.

> Software

60-3 For software-development arrangements that are fully or partially funded by a party other than the vendor that is developing the software and for which technological feasibility of the computer software product has not been established before entering into the arrangement, see Subtopic 730-20 on research and development arrangements.

Subtopic 985-20 excludes the following transactions and activities from its scope. [985-20-15-3]

— Costs of software that is developed or obtained for internal use. Costs are accounted for under Subtopic 350-40.
— Tangible and intangible R&D assets acquired in a business combination; see section 7 of KPMG Handbook, Business combinations.
— Software developed for others under a contractual arrangement, subject to Topic 606 and Subtopic 340-40; see Example 2.3.10.

2.3.30 AI software scoping considerations

Question 2.3.30**

Are there any unique scoping considerations for AI software development?

Interpretive response: No. While AI software (in particular, generative AI software) is of particular interest (now and likely into the future), the accounting guidance that applies to AI software development is the same guidance that applies to the development of any other software.

— **Subtopic 350-40**: Applies to software that is solely for the entity's internal use (internal-use software). Internal-use software can be for true internal use (e.g. payroll processing, inventory management, enterprise resource planning) but also includes software to which an entity sells access to customers solely on a hosted basis. This latter type of internal-use software includes software an individual or entity may access via an on-device (e.g. smartphone) 'thin-client' application, but that has no (or only very limited) functionality when not connected to cloud-based features or functionalities (see Question 2.4.40). Some popular generative and other AI applications qualify as internal-use software because they are only accessed by customers on a hosted basis.

— **Subtopic 985-20**: Applies to costs incurred to develop software to be sold or licensed to third-party customers (external-use software), including software that an entity licenses to a customer as part of a product or providing a service.

— **Other Topics**: Software may be neither internal-use nor external-use; for example, software to be used in R&D is subject entirely to the R&D guidance in Subtopic 730-10, while software an entity develops for others under a contractual arrangement is subject to the contract fulfillment cost guidance in Subtopic 340-40.

Sections 2.3.10 and 2.3.20 discuss identifying software outside the scope of either Subtopics 350-40 or 985-20 and section 2.4 distinguishes between internal- and external-use software.

Once an entity appropriately identifies the internal-use, external-use or other nature of the AI software, Subtopic 350-40, Subtopic 985-20 or any other applicable guidance (e.g. Subtopic 730-10 or Subtopic 340-40) is applied to the AI software development without any specific exceptions or different requirements as compared to any other type of software development.

See chapter 3 (if the entity has not yet adopted ASU 2025-06) or chapter 3A (if the entity has adopted ASU 2025-06) and chapter 5 for guidance on applying Subtopics 350-40 and 985-20, respectively.

Chapter 2 of KPMG Handbook, Research and development, and chapter H of KPMG Handbook, Revenue for software and SaaS, discuss accounting for R&D and contract fulfillment costs, respectively.

2.4 Subtopic 350-40 vs Subtopic 985-20

Excerpt from ASC 350-40

15 Scope and Scope Exceptions

General

> Transactions

15-2 The guidance in the General Subsections of this Subtopic applies to the following transactions and activities:

a. Internal-use software
b. The proceeds of computer software developed or obtained for internal use that is marketed
c. New internal-use software developed or obtained that replaces previously existing internal-use software
d. Computer software that consists of more than one component or module. For example, an entity may develop an accounting software system containing three elements: a general ledger, an accounts payable subledger, and an accounts receivable subledger. In this example, each element might be viewed as a component or module of the entire accounting software system. The guidance in this Subtopic shall be applied to individual components or modules.

Pending Content

Transition Date: (P) December 16, 2027; (N) December 16, 2027 | Transition Guidance: 350-40-65-4

15-2 The guidance in the General Subsections of this Subtopic applies to the following transactions and activities:

…

e. Costs incurred to develop a website.

15-2A Internal-use software has both of the following characteristics:

a. The software is acquired, internally developed, or modified solely to meet the entity's internal needs.
b. During the software's development or modification, no substantive plan exists or is being developed to market the software externally.

15-2B A substantive plan to market software externally could include the selection of a marketing channel or channels with identified promotional,

delivery, billing, and support activities. To be considered a substantive plan, implementation of the plan should be reasonably possible. Arrangements providing for the joint development of software for mutual internal use (for example, cost-sharing arrangements) and routine market feasibility studies are not substantive plans to market software for purposes of this Subtopic. Both characteristics in paragraph 350-40-15-2A must be met for software to be considered for internal use.

15-2C An entity's past practices related to selling software may help determine whether the software is for internal use or is subject to a plan to be marketed externally. For example, an entity in the business of selling computer software often both uses and sells its own software products. Such a past practice of both using and selling computer software creates a rebuttable presumption that any software developed by that entity is intended for sale, lease, or other marketing.

15-3 The General Subsections of this Subtopic provide guidance on when costs incurred for internal-use computer software are and are not capitalized.

15-4A The guidance in the General Subsections of this Subtopic applies only to internal-use software that a customer obtains access to in a **hosting arrangement** if both of the following criteria are met:

a. The customer has the contractual right to take possession of the software at any time during the hosting period without significant penalty.
b. It is feasible for the customer to either run the software on its own hardware or contract with another party unrelated to the vendor to host the software.

15-5 Costs of computer software that is sold, leased, or otherwise marketed as a separate product or as part of a product or process are within the scope of Subtopic 985-20. For example, software designed for and embedded in a semiconductor chip is included in the scope of that Subtopic because it is an integral part of the product. By contrast, software for internal use, though it may be used in developing a product, is not part of or included in the actual product or service sold. If software is used by the vendor in the production of the product or providing the service but the customer does not acquire the software or the future right to use it, the software is covered by this Subtopic. For example, for a communications entity selling telephone services, software included in a telephone switch is part of the internal equipment used to deliver a service but is not part of the product or service actually being acquired or received by the customer. Paragraph 350-40-55-1 includes examples of computer software considered to be for internal use and thus not part of a product or process. Paragraph 350-40-55-2 includes examples of when computer software is not for internal use.

Implementation Costs of a Hosting Arrangement That Is a Service Contract

15-8 The Implementation Costs of a Hosting Arrangement That Is a Service Contract Subsections of this Subtopic follow the same Scope and Scope Exceptions as outlined in the General Subsection of this Section, with specific qualifications noted in paragraph 350-40-15-9.

20 Glossary

Hosting Arrangement

In connection with accessing and using software products, an arrangement in which the customer of the software does not currently have possession of the software; rather, the customer accesses and uses the software on an as-needed basis.

55 Implementation Guidance and Illustrations

General

55-1 The following is a list of examples illustrating when computer software is for internal use:

a. A manufacturing entity purchases robots and customizes the software that the robots use to function. The robots are used in a manufacturing process that results in finished goods.
b. An entity develops software that helps it improve its cash management, which may allow the entity to earn more revenue.
c. An entity purchases or develops software to process payroll, accounts payable, and accounts receivable.
d. An entity purchases software related to the installation of an online system used to keep membership data.
e. A travel agency purchases a software system to price vacation packages and obtain airfares.
f. A bank develops software that allows a customer to withdraw cash, inquire about balances, make loan payments, and execute wire transfers.
g. A mortgage loan servicing entity develops or purchases computer software to enhance the speed of services provided to customers.
h. A telecommunications entity develops software to run its switches that are necessary for various telephone services such as voice mail and call forwarding.
i. An entity is in the process of developing an accounts receivable system. The software specifications meet the entity's internal needs and the entity did not have a marketing plan before or during the development of the software. In addition, the entity has not sold any of its internal-use software in the past. Two years after completion of the project, the entity decided to market the product to recoup some or all of its costs.
j. A broker-dealer entity develops a software database and charges for financial information distributed through the database.
k. An entity develops software to be used to create components of music videos (for example, the software used to blend and change the faces of models in music videos). The entity then sells the final music videos, which do not contain the software, to another entity.
l. An entity purchases software to computerize a manual catalog and then sells the manual catalog to the public.
m. A law firm develops an intranet research tool that allows firm members to locate and search the firm’s databases for information relevant to their cases. The system provides users with the ability to print cases, search for related topics, and annotate their personal copies of the database.

55-2 The following list provides examples of computer software that is not for internal use:

a. An entity sells software required to operate its products, such as robots, electronic game systems, video cassette recorders, automobiles, voice-mail systems, satellites, and cash registers.
b. A pharmaceutical entity buys machines and writes all of the software that allows the machines to function. The pharmaceutical entity then sells the machines, which help control the dispensation of medication to patients and help control inventory, to hospitals.
c. A semiconductor entity develops software embedded in a microcomputer chip used in automobile electronic systems.
d. An entity purchases software to computerize a manual catalog and then sells the computer version and the related software to the public.
e. A software entity develops an operating system for sale and for internal use. Though the specifications of the software meet the entity's internal needs, the entity had a marketing plan before the project was complete. In addition, the entity has a history of selling software that it also uses internally and the plan has a reasonable possibility of being implemented.
f. An entity is developing software for a point-of-sale system. The system is for internal use; however, a marketing plan is being developed concurrently with the software development. The plan has a reasonable possibility of being implemented.
g. A telecommunications entity purchases computer software to be used in research and development activities.
h. An entity incurs costs to develop computer software for another entity under a contract with that other entity.

Excerpt from ASC 985-20

15 Scope and Scope Exceptions

General

> Transactions

15-2 The guidance in this Subtopic applies to the costs, including costs incurred after the date of a business combination or a combination accounted for by a **not-for-profit entity**, of computer software to be sold, leased, or otherwise marketed as a separate product or as part of a product or process, whether internally developed and produced or purchased.

> Other Considerations

15-4 As used in this Subtopic, the terms computer software product, software product, and product encompass a computer software program, a group of programs, and a **product enhancement**.

> Software Subject to a Hosting Arrangement

15-5 The software subject to a **hosting arrangement** is within the scope of this Subtopic only if both of the following criteria are met:

a. The customer has the contractual right to take possession of the software at any time during the hosting period without significant penalty.

b. It is feasible for the customer to either run the software on its own hardware or contract with another party unrelated to the vendor to host the software.

15-7 If the software subject to a hosting arrangement never meets the criteria in paragraph 985-20-15-5, then the software is utilized in providing services and is not within the scope of this Subtopic and, therefore, the development costs of the software should be accounted for in accordance with Subtopic 350-40 on internal-use software (see also paragraph 985-20-55-2).

20 Glossary

Hosting Arrangement

In connection with accessing and using software products, an arrangement in which the customer of the software does not currently have possession of the software; rather, the customer accesses and uses the software on an as-needed basis.

Product Enhancement

Improvements to an existing product that are intended to extend the life or improve significantly the marketability of the original product. Enhancements normally require a product design and may require a redesign of all or part of the existing product.

55 Implementation Guidance and Illustrations

General

> Implementation Guidance

• > Interpretation of Scope

• • > Definition of Software Product

55-1 A software product is most easily defined by describing its necessary qualities. As a product, it is complete and has exchange value. As software, it is a set of programs that interact with each other. A program is further defined as a series of instructions or statements that cause a computer to do work.

• • > Relationship to Revenue Recognition Guidance

55-2 If the entity never sells, leases, or licenses the software (that is, the software meets the criteria in paragraph 985-20-15-5), then the software is used in providing services and the development costs of the software should be accounted for in accordance with Subtopic 350-40. However, if during such software's development or modification (accounted for in accordance with Subtopic 350-40), the entity develops a substantive plan to sell, lease, or otherwise market the software externally, the development costs of the software should be accounted for in accordance with this Subtopic (985-20). Paragraph 350-40-35-7 provides guidance if, after the development of internal-use software is completed, an entity decides to market the software.

• • > Software Marketed as Part of a Product or Process

55-3 The costs of software that is marketed as part of a product or process are included in the scope of this Subtopic. Software is sometimes embedded in a product and sold as part of the product as a whole. Examples are calculators and robots. This type of software is sometimes known as firmware. Also,

some services provided to customers would not be possible without software. Time-sharing and service bureaus are two straightforward examples.

60 Relationships

General

> Intangibles—Goodwill and Other

60-1 For guidance to help determine whether software is developed for internal use or is subject to a plan to be marketed externally, see paragraphs 350-40-15-2 through 15-2C and 350-40-15-5.

Internal-use software is software acquired, internally developed or modified solely to meet the entity's internal needs. This means that: [350-40-15-2A]

- the software is not sold, licensed or otherwise marketed externally; and
- no substantive plan exists or is being developed to market the software externally.

Software that an entity uses to provide services only, including hosting arrangements that do not grant the customer a license to the software (e.g. software- or platform-as-a-service),is not sold, licensed or otherwise marketed externally. Therefore, as long as the second bullet in the preceding paragraph is also met, the software is internal-use software in the scope of Subtopic 350-40. [985-20-15-7, 55-2]

Software that an entity both (1) uses to provide services and (2) sells, licenses or otherwise markets externally is in the scope of Subtopic 985-20 (see Question 2.4.30). [985-20-15-7, 55-2]

2.4.10 Substantive plans to market software externally

A substantive plan must be 'reasonably possible' of implementation. Reasonably possible is "[t]he chance of the future event or events occurring is more than remote but less than likely." [350-40-15-2B, ASC Master Glossary]

A pattern of marketing internal-use software creates a rebuttable presumption that future software will also be marketed, and therefore should not be accounted for under Subtopic 350-40. The following are discussed in chapter 6: [350-40-15-2C]

- Question 6.2.210: determining whether a pattern of marketing internal-use software exists.
- Question 6.2.220: overcoming a rebuttable presumption.

Question 2.4.10

What makes an external marketing plan substantive?

Background: If a 'substantive' plan exists to market software externally, that software is in the scope of Subtopic 985-20. However, if a plan exists that is *not*

substantive, that plan does not alone mean software is not for internal use. [350-40-15-2A, 15-5]

Interpretive response: In addition to being 'reasonably possible' of occurring, the plan should be sufficiently detailed. Judgment will be involved, but we believe a sufficiently detailed plan generally includes details about how the software will be marketed, delivered, priced and any support activities that will be provided. Cost-sharing arrangements and general market feasibility studies alone do not constitute a substantive plan. [350-40-15-2B]

Example 2.4.10
Substantive marketing plan – mobile app

ABC Corp. is designing and developing a mobile application that will allow end-users to catalog and manage their digital media. ABC's marketing team has developed a detailed marketing plan and is actively working with several mobile platform app stores to market and appropriately price the application. Additionally, the marketing team has entered into contracts with digital advertisers to place in-app ads.

Given the status of ABC's negotiations/contracting with the app stores and digital advertisers, ABC concludes that the plan is at least reasonably possible of occurring and sufficiently detailed. Therefore, the plan is substantive and ABC accounts for its software development costs under Subtopic 985-20.

Example 2.4.20
Not a substantive marketing plan

XYZ Corp. provides revenue management services to healthcare entity customers. These services include assistance with medical coding, billing and collections. XYZ recently began to internally develop a software tool that it will use in providing those services. The software helps to ensure appropriate billing codes are mapped for Medicare/Medicaid reimbursements it bills for customers as part of the services.

XYZ will use the software internally to assist with its service offering. However, XYZ has also engaged a marketing firm to survey healthcare providers to determine if there is a potential market for this software product externally – e.g. with healthcare entities that perform XYZ's service activities in-house.

XYZ concludes the survey is a routine market feasibility study and is not a substantive plan to market the software externally. Therefore, XYZ accounts for its software development costs under Subtopic 350-40.

2.4.20 Changes in scope

Question 2.4.20

Does devising a substantive plan to market uncompleted internal-use software externally mean it is no longer in the scope of Subtopic 350-40?

Background: For purposes of this question, assume an entity devises a substantive plan to externally market software it previously concluded was solely for internal use before application development is complete.

Section 6.2.50 discusses the accounting for capitalized internal-use software development costs when that software is sold or licensed externally after development.

Interpretive response: Yes. If an entity devises a substantive plan to externally market internal-use software that is still under development, that software is now in the scope of Subtopic 985-20.

In that event, any application development stage costs that had been capitalized (see section 3.2) before technological feasibility of the software is established under Subtopic 985-20 (see section 5.2) – and therefore would have been expensed as incurred under Subtopic 985-20 – are expensed in the period of the scoping reassessment. If technological feasibility has not been established at the time of the reassessment, *all* of the previously capitalized development costs are expensed upon reassessment, and no further development costs are capitalized until technological feasibility is established. [350-40-35-9, 985-20-55-2]

Question 2.4.30

Can software that has been marketed externally become internal-use software?

Background: As software vendors migrate their software offerings to the cloud (e.g. cease, or make plans to cease, licensing their software to customers), the question arises about whether external-use software can become internal-use software, and if so when does that occur?

This question is important because it affects the subsequent measurement guidance (see chapter 6) that applies to the software. It may also affect the software development cost guidance that will apply to upgrades and enhancements of the software, and whether a particular change to the software is, in fact, an upgrade or enhancement (i.e. because the definitions differ between Subtopic 350-40 and Subtopic 985-20 – see sections 3.2.50 and 5.2.20, respectively).

Interpretive response: Yes, while this notion is not explicitly discussed in either Subtopic 350-40 or Subtopic 985-20, we believe external-use software can become internal-use software.

In general, we believe this occurs when the following conditions are met with respect to the software:

— sales of (1) licenses to the software and (2) PCS that includes the right to specified or unspecified upgrades and enhancements of the software (when the likelihood of their provision is more than remote) have ceased; and

— the entity no longer has a substantive plan to market the software externally other than as a cloud (e.g. SaaS) offering.

After the software is determined to be internal-use, it is subject to Subtopic 350-40 in all respects from that point forward – i.e. accounting for upgrades/enhancements, subsequent measurement, presentation and disclosure.

2.4.30 Software marketed as part of a product or process

Software is often embedded in another product or is part of a service process – commonly referred to as firmware.

Modern products such as robotics, game consoles, household appliances, smartphones and automobiles typically include firmware. Software development costs for firmware are subject to the guidance in Subtopic 985-20. [985-20-55-3]

A service or product offering can include an implied software license to the end customer – i.e. the software license is not an explicitly promised good in the contract with the customer. Software that gets licensed to customers as part of a service or product offering – explicitly or implicitly – is subject to the guidance in Subtopic 985-20. [985-20-55-3]

Example 2.4.30
Software embedded in a product

XYZ Corp. manufactures automobiles. XYZ also embeds internally developed navigation, speech recognition and safety-related software in its automobiles that is a critical part of the end-product sold to customers.

This firmware is frequently the subject of direct marketing efforts to end customers (e.g. highlighted in XYZ's commercials).

The software development costs related to the firmware are subject to Subtopic 985-20.

Example 2.4.40
Software that is part of a process

DEF Inc. offers data analytics services to customers that include monitoring, storing, analyzing and reporting on customers' data. DEF uses internally

developed software to provide these services and could not provide them without this (or substantially equivalent) software.

The software is installed to, and interfaced directly with, the customer's network infrastructure. Customers license the software during the service period for purposes of viewing and manipulating their data and producing tailored data reports.

DEF's software is subject to Subtopic 985-20. While the software is a critical part of DEF's service offering, licensing the software for customers' use is also an integral feature of DEF's offering.

2.4.40 Specific application matters

Question 2.4.40

Are thin-client applications internal- or external-use software?

Background: A 'thin-client' application typically refers to an application that has significantly limited independent functionality (e.g. performs minimal independent processing, if any), and whose principal purpose is to serve as an interface to cloud-based features and functionalities. Thin client applications may be desktop or mobile device applications.

Examples of thin-client applications may include (not exhaustive):

— search engine apps, which permit the user to access the engine's hosted algorithmic functions, but do not return search results when the user's device is not connected to the internet;

— online marketplace apps, which permit the user to search the online marketplace and make purchases only when connected to the internet; and

— online banking apps, which permit the user to access their banking information and make banking transactions only when connected to the internet.

Interpretive response: Thin-client applications are generally considered internal-use software. Such applications are typically not being marketed externally as software products in their own right, and would not qualify as software products under Subtopic 985-20 because their limited independent functionality means they do not have exchange value (i.e. independent marketability) separate from the cloud-based services to which they connect the user. [985-20-55-1]

Question 2.4.50

Are freemium apps internal- or external-use software?

Background: Many apps, desktop and mobile, are offered free to the public for download. The software vendor monetizes the app from things like in-app ads, in-app purchases (e.g. of virtual goods to be used by players in a gaming app), or premium services (e.g. ad-free gameplay, additional cloud storage), rather than from downloads or licensing of the app.

Interpretive response: In general, freemium apps that are not solely thin-client apps (see Question 2.4.40) are external-use software in the scope of Subtopic 985-20. The software vendor's monetization strategy does not affect whether the app qualifies as a software product under Subtopic 985-20 and is being licensed to customers with the intent to earn revenue from its central software functionality (e.g. as a game). [985-20-55-1]

Example 2.4.50

Mobile gaming app that is free to download

ABC Corp. develops mobile gaming apps. ABC's business model is to offer its gaming apps free for download from a platform-specific app store. ABC's gaming apps include ads the user sees, and cannot bypass, while using the apps and offer virtual goods for in-app play customers can purchase for a fee.

When developing its newest gaming app, ABC concludes the software is external-use software subject to Subtopic 985-20. The fact that the app is free for initial download does not change this conclusion. The app qualifies as a software product under paragraph 985-20-55-1 because it will be a complete software program with substantive independent exchange value (i.e. ABC could charge for downloads), and ABC has a substantive plan to market (i.e. license) the software to customers.

Question 2.4.60

Does Subtopic 985-20 apply to third-party owned software an entity obtains the right to market?

Interpretive response: Yes. Subtopic 985-20 does not require that the entity own the underlying software. The cost of the rights acquired from the software owner are subject to the guidance in Subtopic 985-20.

Question 2.4.70

What software cost guidance applies to the on-premise software in a hybrid cloud arrangement?

Background: It is increasingly common for arrangements to include on-premise and/or on-device software licenses and SaaS (cloud only) features and functionality – e.g. license to an on-premise or on-device software application and file storage (including sharing and collaboration through a web host) – or software sold through SaaS subscriptions, but with a substantive offline mode. These arrangements are often referred to as 'hybrid cloud arrangements'.

Under Topic 606, there is often significant judgment in determining whether the vendor has one or more performance obligations in these arrangements. See Question C310 in KPMG Handbook, Revenue for software and SaaS.

Interpretive response: The following separately addresses software vendor and customer considerations.

Software vendors

Regardless of a software vendor's revenue recognition conclusions under Topic 606, we believe on-premise or on-device software that is licensed to the customer is subject to Subtopic 985-20 for the software vendor. This includes software sold in hosting arrangements that meets the license criteria in paragraph 985-20-15-5.

The scope of Subtopic 985-20 does not distinguish between:

— licensed software for which the licenses thereto are distinct from other non-license services, including SaaS or other cloud-based elements, with which they are sold under Topic 606; and

— licensed software for which the licenses thereto are not distinct from similar non-license services.

However, we believe an exception to external use categorization arises when the licensed on-premise or on-device software is merely a thin-client application (see Question 2.4.40).

Customers

Subtopic 350-40 includes guidance on multiple-element arrangements, including allocating costs to those elements. Section 2.7 discusses these requirements.

There is no optional practical expedient in Subtopic 350-40 to not separate elements in a multiple-element arrangement. This means the customer in a hybrid cloud arrangement must separately account for:

— the on-premise or on-device license(s), including any hosted software elements that meet the criteria in paragraph 350-40-15-4A (see section 2.5), which will be accounted for under the general guidance in Subtopic 350-40; and

— the cloud-based elements, which will be subject to the 'implementation costs of a hosting arrangement that is a service contract' guidance in Subtopic 350-40.

Example 2.4.60

Software vendor accounting for hybrid cloud offering development costs

This example is a continuation of Example C310.2 in KPMG Handbook, Revenue for software and SaaS. ABC's analysis leading to the conclusion that the on-premise software license and cloud features are not distinct from each other is not reproduced here.

ABC Corp.'s core solution to customers is marketed as a cloud service. However, the solution includes on-premise software subject to an end-user license agreement. Customers can perform many of the solution's functionalities when they are not connected to ABC's cloud (i.e. they are offline, using the on-premise software only), but other functionalities are accessible only if connected to ABC's cloud.

ABC does not sell the on-premise software or functionalities separate from its cloud service, and customers cannot access the cloud functionalities without the on-premise software; that software is what permits the customer to access the cloud features.

While there are substantive capabilities available to the customer in offline mode, without the cloud features, customers would not be able to complete projects using the software. This is because the offline mode permits the customer to perform only some tasks toward completing projects using the software; other significant features and functionalities integral to completing projects are available only when using the software while connected to the cloud. The ability to create and complete entire projects is the reason customers acquire ABC's solution.

There are (1) no other on-premise applications that work together with the cloud component of the solution and (2) no other on-premise or cloud solutions available that customers could combine with ABC's on-premise software and achieve the functionality provided by ABC's overall solution. Customers would have to backtrack substantially or do a significant amount of re-work to achieve the same results outside of ABC's application.

ABC accounts for this offering as a single, combined performance obligation under Topic 606.

Despite this revenue recognition conclusion, the substantive offline capabilities of the on-premise software mean that the software is not a thin-client app or similar. Therefore, ABC accounts for:

- the development costs of the on-premise software under Subtopic 985-20; and
- the development costs of the software that underlies the cloud-based features of the offering that can *only* be accessed when connected to ABC's cloud, and will never be licensed to ABC's customers, under Subtopic 350-40.

Example 2.4.70

Customer accounting for hybrid cloud offering development costs

Customer enters into a three-year contract for ABC's offering described in Example 2.4.60. In addition to the core offering, Customer engages ABC to perform implementation services, principally to configure the offering for Customer's needs and to install the on-premise software, at the outset of the arrangement. The configuration services affect both the on-premise and cloud-based software.

Customer agrees to pay $3 million in core offering fees for the three-year subscription, with $1 million payable annually in advance, plus another $1 million for the implementation services, due at the outset of the arrangement. The core offering fees do not break out any amount for the on-premise software license element, and the implementation fees do not separately price the configuration or installation services or refer to separate such services for the cloud features of the offering and the on-premise software.

Customer accounts for the arrangement under Subtopic 350-40 as follows.

- Customer allocates the $4 million in contractual fees to the following elements on the basis of relative stand-alone prices (see section 2.7):
  - on-premise software license;
  - implied PCS for the on-premise software – i.e. technical support and when-and-if available updates, upgrades and enhancements;
  - cloud subscription;
  - configuration of the cloud-based features;
  - configuration of the on-premise software; and
  - installation of the on-premise software.

  The software license and the configuration and installation services are all provided at the outset of the arrangement.
- Under the general sections of Subtopic 350-40, Customer recognizes:
  - an intangible software license asset equal to the sum of (1) the portion of the $2 million paid upfront allocable to the on-premise software license and configuration and installation of that software and (2) the amount of the liability recognized in the next sub-bullet; and
  - a liability for the portion of the unpaid subscription fees ($1 million to be paid in each of Years 2 and 3) allocable to the on-premise software license and the related configuration and installation services.
- Under the 'implementation costs of a hosting arrangement that is a service contract' sections of Subtopic 350-40, Customer recognizes:
  - a CCA implementation cost asset equal to the sum of (1) the portion of the upfront $2 million payment allocable to the cloud configuration services and (2) the amount of the liability in the next sub-bullet; and
  - a liability for the portion of the unpaid subscription fees allocable to the cloud configuration services.

— Customer recognizes prepaid cloud subscription fees equal to the portion of the upfront $2 million payment allocable to the cloud subscription; and

— Customer recognizes prepaid PCS equal to the portion of the upfront $2 million payment allocable to the on-premise software PCS that will be provided over the arrangement term.

Question 2.4.80

Are costs incurred to migrate internal-use software to a cloud service provider's hosting environment in the scope of Subtopic 350-40?

Background: Assume an entity currently hosts internal-use software it has licensed from a software vendor on its own servers. The entity decides it wants to migrate the software from its own IT environment to a third-party cloud service provider's hosting environment (e.g. Amazon Web Services or Microsoft Azure).

The entity incurs the following costs related to the migration:

— configuration of the hosting environment;
— installation of the software vendor's software into the cloud service provider's hosting environment;
— testing; and
— business process reengineering for IT management.

For purposes of this question, it is assumed there is no lease in the entity's hosting arrangement with the cloud service provider. See chapter 3 of KPMG Handbook, Leases, for guidance about identifying a lease.

Interpretive response: Yes. We believe the 'Implementation Costs of a Hosting Arrangement That Is a Service Contract' subsections of Subtopic 350-40 apply. This is because the basis for conclusions to ASU 2018-15 lists infrastructure-as-a-service (IaaS) as an example of a hosting arrangement subject to that guidance, and the example arrangement in the background is an example of IaaS. [ASU 2018-15.BC2]

In applying the guidance in Subtopic 350-40 to the background example, the entity's specific facts and circumstances, and the nature of the migration costs, will determine whether the specific costs are capitalizable or should be expensed as incurred.

Alternative view

We are aware of an alternative view that the arrangement described in the background between the entity and the cloud service provider is not in the scope of the 'Implementation Costs of a Hosting Arrangement That Is a Service Contract' subsections of Subtopic 350-40. This is based on the definition of a hosting arrangement as "an arrangement in which the customer of the software does not currently have possession of the software." [350-40 Glossary]

In the background example, the entity currently has possession of the vendor's software. Even after the migration, the entity will still have possession of the

software because the cloud service provider is hosting the software on the entity's behalf (not on the software vendor's behalf).

We do not believe the alternative view is appropriate given the discussion in the basis for conclusions to ASU 2018-15, and our observations of the deliberations of the EITF and the FASB before issuing the ASU (and of previously issued ASU 2015-05).

We further note that even if the alternative view were applied, it would appear the referenced subsections of Subtopic 350-40 could still be applied by analogy. The basis for conclusions to ASU 2018-15 notes the EITF decided to stay silent about, and therefore not prohibit, analogies to this guidance. The public EITF discussion about analogies further suggested that neither the EITF, nor the FASB, wanted to limit the application of this guidance by analogy. Because of this, even if the alternative view were applied, we believe it would be acceptable (though not required) for the entity to apply the 'Implementation Costs of a Hosting Arrangement That Is a Service Contract' subsections of Subtopic 350-40 if no other US GAAP addressed the migration costs incurred. [ASU 2018-15.BC15]

Question 2.4.90

Are implementation costs incurred to host customer-facing software in another cloud service provider's hosting environment in the scope of Subtopic 350-40?

Background: A cloud service (e.g. SaaS) provider may not host its customer-facing software (i.e. the software its customers obtain access to in CCAs) in its own IT environment. Instead, it might engage another cloud service provider (e.g. Amazon Web Services, Microsoft Azure) to host that software.

The question arises about whether the guidance in Question 2.4.80 applies to the SaaS provider in the preceding paragraph.

Interpretive response: Yes. The SaaS provider in the background is the customer in a CCA (IaaS). Therefore, the guidance in Question 2.4.80 applies even though the software that will be hosted by the IaaS provider is customer-facing software.

Question 2.4.100

Can costs be in the scope of both the general and CCA implementation cost guidance in Subtopic 350-40?

Background: Consider an example in which a customer in a CCA develops software that it will own for the purpose of interfacing the hosted software subject to the CCA with other, internal-use software it licenses or accesses through another CCA. In variations of this example, the developed interface

may reside either on-premise or in the cloud service provider's hosting environment.

Questions arise about whether:

— the interface is itself internal-use software (subject to the general guidance subsections of Subtopic 350-40), or is instead an implementation cost of the new CCA (subject to the implementation costs of a hosting arrangement that is a service contract subsections of Subtopic 350-40); and

— costs required to be expensed as incurred under the internal-use software or CCA implementation cost guidance can be considered for capitalization under the other.

Interpretive response: No. Costs cannot be both internal-use software costs and CCA implementation costs; they are one or the other. This means that if an internal-use software cost or a CCA implementation cost is required to be expensed as incurred based on the internal-use software guidance or CCA implementation cost guidance, respectively, it cannot be reconsidered for capitalization under the other. [350-40-15-3, 15-4D]

As a practical matter, we do not believe an entity would reach a different conclusion even if it did so because entities apply the same guidance to determine what costs are capitalizable. [350-40-30-5]

In the background example, because the customer owns the interface software IP, the software development costs incurred by the customer are internal-use software costs subject to the general guidance on internal-use software, rather than the implementation costs of a hosting arrangement that is a service contract guidance. Any of those costs required to be expensed as incurred under the internal-use software guidance cannot be reconsidered for capitalization as CCA implementation costs.

Because the customer owns the software IP, the conclusion that the software development costs are internal-use software costs, rather than CCA implementation costs, would be the same regardless of where the interface will reside when it is in production – i.e. on-premise or in the cloud service provider's hosting environment. However, if the interface will reside in the cloud service provider's hosting environment, the software could be in the scope of Subtopic 985-20 if it is determined that it will be licensed to the cloud service provider.

Question 2.4.110**

Does an acquirer separately account for the embedded software component of a tangible good with embedded software?

Background: Entities often acquire tangible goods with embedded software components, such as cars, network equipment and industrial robots. In such cases, a question arises as to whether the embedded software component should be accounted for separately from the tangible asset component.

Topic 606 addresses this question for sellers of such tangible goods. See Question C290 in KPMG Handbook, Revenue for software and SaaS.

Interpretive response: US GAAP does not explicitly address this question. The FASB considered adding guidance to US GAAP for this issue in ASU 2025-06 but ultimately decided not to do so.

We believe in practice that acquiring entities generally account for any embedded software that is integral (or essential) to the functionality of a tangible asset simply as *part of* that tangible asset (i.e. not as a separate unit of account), and account for that tangible asset under the applicable tangible asset accounting guidance (e.g. Topic 360 on PP&E or Topic 330 on inventory). In explaining its decision not to amend US GAAP for this question in the basis for conclusions to ASU 2025-06, the FASB essentially affirmed its understanding of this common practice. [ASU 2025-06.BC59]

Entities *developing* such software ('firmware') – e.g. the car, network equipment or robot manufacturer – apply the external-use software guidance in Subtopic 985-20 to firmware development (see chapter 5).

2.5 Hosting arrangements that grant a software license

Excerpt from ASC 350-40

15 Scope and Scope Exceptions

General

> Transactions

15-4A The guidance in the General Subsections of this Subtopic applies only to internal-use software that a customer obtains access to in a **hosting arrangement** if both of the following criteria are met:

a. The customer has the contractual right to take possession of the software at any time during the hosting period without significant penalty.
b. It is feasible for the customer to either run the software on its own hardware or contract with another party unrelated to the vendor to host the software.

15-4B For purposes of the guidance in paragraph 350-40-15-4A(a), the term without significant penalty contains two distinct concepts:

a. The ability to take delivery of the software without incurring significant cost
b. The ability to use the software separately without a significant diminution in utility or value.

15-4C Hosting arrangements that do not meet both criteria in paragraph 350-40-15-4A are service contracts and do not constitute a purchase of, or convey a license to, software.

15-4D Implementation costs of a hosting arrangement that does not meet both criteria in paragraph 350-40-15-4A shall be accounted for in accordance with the Implementation Costs of a Hosting Arrangement That Is a Service Contract Subsections of this Subtopic.

20 Glossary

Hosting Arrangement

In connection with accessing and using software products, an arrangement in which the customer of the software does not currently have possession of the software; rather, the customer accesses and uses the software on an as-needed basis.

Excerpt from ASC 985-20

15 Scope and Scope Exceptions

General

> Software Subject to a Hosting Arrangement

15-5 The software subject to a **hosting arrangement** is within the scope of this Subtopic only if both of the following criteria are met:

a. The customer has the contractual right to take possession of the software at any time during the hosting period without significant penalty.
b. It is feasible for the customer to either run the software on its own hardware or contract with another party unrelated to the vendor to host the software.

15-6 For purposes of criterion (a) in paragraph 985-20-15-5, the term significant penalty contains two distinct concepts:

a. The ability to take delivery of the software without incurring significant cost
b. The ability to use the software separately without a significant diminution in utility or value.

20 Glossary

Hosting Arrangement

In connection with accessing and using software products, an arrangement in which the customer of the software does not currently have possession of the software; rather, the customer accesses and uses the software on an as-needed basis.

A 'hosting arrangement' is an arrangement that allows customers to access and use software on an as-needed basis without having possession of it. [350-40 Glossary, 985-20 Glossary]

This means the customer in the arrangement does not download the software onto servers or computers that it owns or leases. Rather, the software is hosted by the software vendor. The functionalities of the software are made

available to the customer through the internet or a dedicated line. See also Questions 2.5.50 and 2.5.60.

In accounting for hosting arrangements (as the software vendor or the customer), the key question is whether:

— the arrangement grants the customer a license to the hosted software; or
— instead, the customer is solely being provided access to the hosted software as a service.

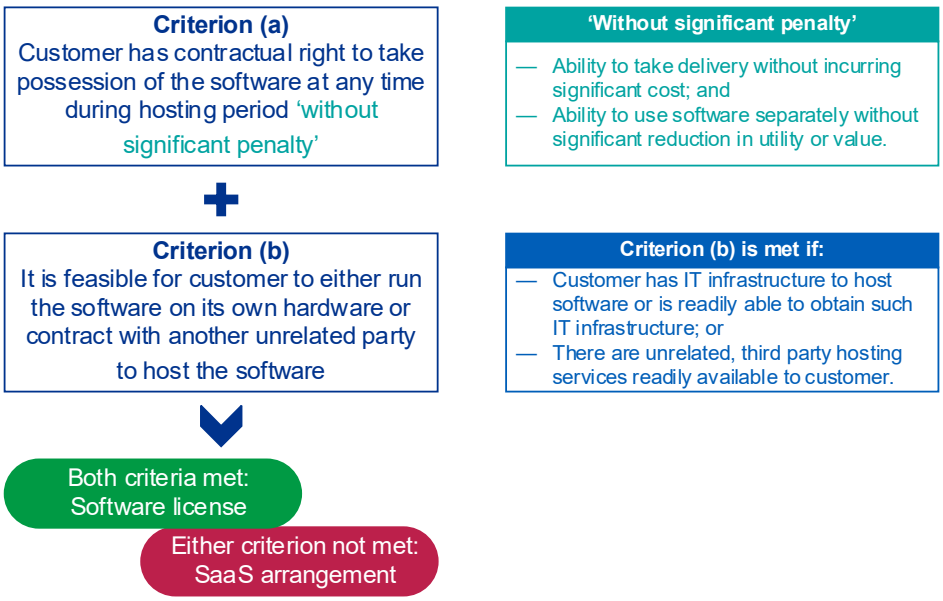
This question determines:

— for the software vendor, whether its software development costs are subject to Subtopic 350-40 or Subtopic 985-20; and

— for the customer, whether the arrangement is in the scope of the general internal-use software guidance in Subtopic 350-40 or the 'implementation costs of a hosting arrangement that is a service contract' guidance in that Subtopic.

The identical guidance in Subtopics 350-40 and 985-20 states that, for accounting purposes, a hosting arrangement does not include a license to the hosted software unless it meets both of the criteria included in the diagram below. A hosting arrangement that does not meet both of those criteria is a service contract – i.e. a CCA, such as SaaS or PaaS. [350-40-15-4A – 15-4C, 985-20-15-5 – 15-7]

Questions 2.5.10 – 2.5.60 and the related examples that follow address application questions surrounding those criteria.



Question 2.5.10

Can the customer have a 'contractual right to take possession of the software at any time during the hosting period' if such right is not explicit?

Interpretive response: Yes. If the customer has an enforceable right to take possession of the software as a matter of law, the fact that the contract does not explicitly provide for that right does not matter.

It may be the case, in some jurisdictions, that the relevant laws or regulations, or the software vendor's customary business practices, provide the entity with that enforceable right even if neither the entity, nor the vendor, intended for the contract to convey that right.

Question 2.5.20

What does 'at any time' mean?

Interpretive response: In the context of criterion (a) in paragraphs 350-40-15-4A and 985-20-15-5, we believe that a contractual right to take possession of the software can meet the criterion even if the customer does not have the right to take possession of the software at every single point in time during the hosting arrangement. Judgment is required based on the specific facts and circumstances.

The following table provides our analysis of some common situations encountered in practice.

Scenario	Take possession at any time?
The customer has the right to take possession of the software *at every single point in time during the contract.*	Yes. The customer has the contractual right to take possession of the software at any time during the hosting period.
The customer has the right to take possession of the software throughout the hosting arrangement *except for*: — the last few days of the hosting arrangement (or last few months of a long-term hosting arrangement); or — a few days of each month (e.g. the last five days or the first five days of each month) in a long-term hosting arrangement.	Yes. The customer has the right to take possession of the software at any time during the hosting period; the restrictions are not substantive in terms of meaningfully restricting the customer's rights to take possession of the hosted software.
The customer has the right to take possession of the software either: — only at sporadic or specific points in time (e.g. only on the last day of each year) during the hosting arrangement; or	No. The customer does not have the right to take possession of the software at any time during the hosting period. Therefore, the arrangement does not include a

Scenario	Take possession at any time?
— only upon the occurrence of a contingent event that is neither within the control of the entity, nor reasonably certain to occur.	software license – i.e. it is a cloud computing arrangement.
The customer has the right to take possession of the software if the vendor materially breaches the contract.	No. The customer does not have the right to take possession of the software at any time during the hosting period. Therefore, the arrangement does not include a software license – i.e. it is a cloud computing arrangement.

Question 2.5.30

What costs does an entity consider in determining if the customer will incur a 'significant' penalty from taking possession of the software?

Background: Having the right to take possession of the software 'without significant penalty' includes the ability to take delivery of the software without incurring significant cost. [350-40-15-4B(a)]

Interpretive response: The focus of the analysis should be on direct and incremental costs. The mere existence of some level of cost in connection with taking possession of the software would not, by itself, necessarily result in a 'significant penalty'.

Direct and incremental costs include forfeited hosting (or other upfront) fees, termination fees or penalties incurred to cancel the hosting arrangement. Penalties include hosting fees the customer is required to continue to pay to the software vendor after termination of the vendor's hosting services. Similarly, if the customer would incur significant 'switching costs', that would also generally constitute a penalty. For example, if the customer would be required to invest in the IT infrastructure to host/support the software but would not receive a commensurate reduction in hosting fees under the contract, that deficiency would generally be considered a penalty from taking possession of the software.

Although determining whether penalty costs are significant will require judgment, we believe costs that exceed 10% of the total contract fees (e.g. software license fees as well as any initial, non-cancellable PCS and/or hosting service fees) usually would be a strong indicator that the costs are significant.

However, all facts and circumstances should be considered, and a penalty of less than 10% might be considered significant if it creates a substantial disincentive for the customer to take possession of the software.

Question 2.5.40

What constitutes a significant diminution in utility or value?

Background: Having the right to take possession of the software 'without significant penalty' includes the ability to use the software separately without a significant diminution in utility or value. [350-40-15-4B(b)]

Interpretive response: Determining whether a customer in a hosting arrangement will suffer a significant diminution in utility or value of the software if it takes possession of the software may depend on a variety of factors, including the following (not exhaustive).

- Whether taking possession negates the customer's right to receive one or more specified upgrades or unspecified updates or upgrades that are considered integral to maintaining the utility of the software; see Question C170 in KPMG Handbook, Revenue for software and SaaS. If the customer forfeits its right to receive integral updates or upgrades by taking possession, this would indicate it will incur a significant diminution in utility and value of the software from terminating the hosting services.
- Whether there are significant features or functionalities available to the customer when the software is hosted by the vendor that would no longer be available to the customer if it took possession. This would indicate the customer will incur a significant diminution in utility or value of the software from terminating the hosting services.
- Whether incremental resources must be obtained by the customer to maintain the functionality of the software if it elects to take possession. For example, the customer may need to obtain a license or access to an additional software product or implement additional manual procedures to compensate for a loss of utility in the software included in the hosting arrangement. The need to obtain incremental resources to maintain the functionality of the software would indicate that the customer will experience a significant reduction in utility of the software from taking possession of it.
- If the customer has the ability to transfer the hosting services to another provider, while retaining the right to future specified or unspecified upgrades and enhancements of the software, that may be an indicator that it would not incur a significant diminution of utility or value from taking possession of it.

However, we believe a significant diminution in utility or value from taking possession of the software does not impose a significant penalty on the customer if:

- there are readily available resources (e.g. on-premise software, third-party hosting services or hardware that is sold separately) that can replace the significant diminution in utility or value; and
- the cost to obtain those readily available resources is comparable to the cost of the hosting services they will replace.

Importantly, even if both criteria in the preceding paragraph are met, the customer may still incur a significant penalty. For example, as outlined in Question 2.5.30, the customer may have to pay a significant termination fee, forfeit a significant upfront fee, or continue to pay the vendor hosting fees. In any of those cases, the fact it can replace a significant diminution in utility or value at a cost comparable to the software vendor's hosting services does not mean the customer would not incur a significant penalty from taking possession of the software.

Example 2.5.10
Licensing or SaaS arrangement (1)

Customer enters into a three-year contract with ABC Corp. to access ABC's software (Product H) in a hosting arrangement. The contract requires an upfront payment of $500,000, and includes a stated monthly fee of $25,000 for the hosting services provided by ABC.

The following additional facts are relevant.

- Customer has the enforceable right under the contract to take possession of Product H at any time for no additional fee. If it does so, it will no longer be required to pay the $25,000 monthly fee for the hosting services.
- If Customer takes possession of Product H, it loses the right to future unspecified updates, upgrades and enhancements. However, Product H is a mature product that ABC updates infrequently and updates are typically minor and not integral to maintaining the utility of Product H.
- Customer has significant and established IT capacity and resources such that the incremental costs of electing to take possession of the software from ABC would not be significant in comparison to the hosting service fees it would avoid.
- The stated hosting fees are equal to the observable stand-alone (selling) price for those services.

Customer and ABC both conclude that the contract includes a license to the Product H software and hosting services. In accordance with paragraphs 350-40-15-4A and 985-20-15-5:

- Customer has the contractual right to take possession of Product H at any time and can do so without incurring a significant penalty. Customer will not incur a significant penalty if it takes possession of the software because:
  - there is no fee or penalty for terminating the hosting services;
  - Customer does not have to continue to pay for the hosting services after they are terminated; and
  - despite the fact that Customer will lose the right to obtain future updates, upgrades and enhancements, those items are not integral to maintaining the utility of Product H outside of the hosting environment because Product H is a mature software product. As such, Customer will not experience a significant diminution in utility or value of Product H from taking possession of Product H.

— It is feasible for Customer to run (i.e. host) Product H on its own because of its significant and established IT capacity and resources. Because Customer has a significant and established IT capacity, it will incur no (or minimal) incremental costs to host the Product H software.

Example 2.5.20
Licensing or SaaS arrangement (2)

Customer enters into a three-year contract with DEF Corp. to access DEF's software (Product Q) in a hosting arrangement. The contract requires an upfront payment of $400,000, and includes a stated monthly fee of $30,000 for the hosting services provided by DEF. In addition, the following facts are relevant.

— While Customer does not have significant IT capacity on its own, there are third-party hosting service providers that can host Product Q for Customer for a monthly fee that is comparable to the fee for the hosting services provided by DEF.

— All the core functionality of Product Q will remain available to Customer if it chooses to take possession. However, significant search and data reporting functionalities are available to Customer only when Product Q is connected to DEF's proprietary, hosted database, which is only accessible to customers using Product Q within DEF's hosting environment.

Customer and DEF both conclude the contract does *not* include a license to the Product Q software, and therefore is a SaaS arrangement. While Customer could have a third-party host Product Q in place of DEF, and Customer has the contractual right to take possession of Product Q at any time, it cannot take possession without incurring a 'significant penalty' because:

— it will lose access to DEF's proprietary, hosted database, without which significant functionalities will not be available; and

— there are no other readily available resources it could use to replace those functionalities because DEF's database is proprietary and only available to Customer while using Product Q within DEF's hosting environment.

Example 2.5.30
Licensing or SaaS arrangement (3)

Assume the same facts outlined in Example 2.5.10, except the contract requires Customer to provide six months' notice to terminate the hosting services. That is, on Day 1, if Customer takes possession of Product H and terminates the hosting services, it will still have to pay $150,000 in hosting service fees ($25,000 × 6 months).

Customer and ABC conclude the contract does not include a license to Product H, and therefore it is a SaaS arrangement. This is because, while Customer has the contractual right to take possession of Product H at any time, it cannot do so without incurring a 'significant penalty'.

The six-month notice requirement constitutes a significant penalty because Customer must pay this amount without receiving benefit for the fees paid (i.e. it will receive no services in return for those fees) and because that amount of $150,000 exceeds 10% of the fees under the contract ($500,000 upfront fee + $900,000 [$25,000 × 36 months] in hosting service fees).

Question 2.5.50

Does a software license exist if the software will be hosted on servers that are leased to the customer by the software vendor?

Interpretive response: Yes. If the vendor's software is hosted on servers the customer is leasing from the vendor, the customer *has* possession of the software; this is the same as if the software were hosted on customer-owned servers or servers the customer was leasing from a third party. As such, there is no 'hosting arrangement', which is defined as an arrangement in which the customer does not currently have possession of the software. [350-40 Glossary, 985-20 Glossary]

Importantly, it does not matter *why* a lease is determined to exist. For example, in some cases, a customer may explicitly lease equipment from the software vendor, such as servers to host the vendor's software. In those cases, there is typically an explicit software license between the software vendor and the customer – i.e. the intent of the arrangement between those two parties is for the customer to license the vendor's software.

However, in other cases, an 'embedded lease' may exist, even if there is no explicit lease agreement or any lease mentioned in the contract between the parties. An embedded lease *may* exist if, for example, a server is dedicated to the customer – i.e. the server is not used to host software or provide services for any other customer; this is the case even if there is no mention of a lease in the contract and that server is housed in the vendor's data center – e.g. the server may be viewed by the entity as merely part of the data center.

An entity's analysis of whether a lease exists occurs at contract inception. See chapter 3 of KPMG Handbook, Leases, for guidance on evaluating whether a lease exists under Topic 842.

Because an entity's accounting for the contract may differ significantly depending on whether there is a software license in the arrangement, it is important for entities (i.e. customers and vendors) that enter into hosting arrangements (including those that may be characterized solely as SaaS arrangements) to consider whether the customer is leasing the equipment used to host the software. And if so, the entity should account for the arrangement as one that includes a software license (i.e. rather than as a SaaS arrangement).

Question 2.5.60

Is the conclusion about whether a software license exists affected by the customer's or vendor's use of a third-party hosting service?

Interpretive response: No. Determining whether a contract includes a software license is not affected by whether the customer or vendor uses a third party to host the vendor's software.

The fact the customer uses a third-party hosting service provider – or would be required to use a third-party provider if it were to exercise its right to take possession of the software in a hosting arrangement – rather than hosting the software on its own IT equipment, does not affect the conclusion that would otherwise be reached about whether the contract includes a software license.

Similarly, the fact the vendor uses a third-party hosting service provider to host its software, rather than hosting the software on its owned or leased equipment, does not change the conclusion that would otherwise be reached as to whether the contract includes a software license; this includes the possibility the customer *could* be deemed to be leasing (or sub-leasing from the vendor) the third party provider's equipment.

Example 2.5.40

Licensing or SaaS arrangement (4)

ABC Corp.'s typical customer contract provides customers with the right to use its software (Product J) on a SaaS basis. ABC hosts Product J using a third-party hosting service provider (XYZ public cloud provider), rather than hosting Product J in its own data center.

ABC's customers are not permitted to take possession of Product J. ABC manages and controls the hosting services from XYZ associated with Product J – i.e. ABC has the contract with XYZ for the hosting services.

ABC bills customers on a monthly or quarterly basis, which includes proportional reimbursement of ABC's actual costs for the XYZ hosting services related to Product J.

Customer (an existing customer of XYZ) has expressed an interest in deploying Product J in its own XYZ hosting environment, rather than ABC's, to take advantage of Customer's favorable contract terms and pricing arrangement with XYZ – i.e. Customer will realize savings in actual hosting costs by structuring the arrangement in this manner.

Notwithstanding Customer's desire to achieve these cost savings, it is the intent of both ABC and Customer to have ABC manage and control the hosting of Product J in the same manner as ABC manages and controls its typical arrangements; this includes the provision that Product J cannot be removed from XYZ's hosting environment.

To permit this arrangement, a provision has been added to Customer's agreement with XYZ to give ABC access, billing, control and management

rights/responsibilities for a separate Customer account with XYZ that is dedicated to hosting Product J. Customer is not permitted to take possession of Product J or transfer the software to another hosting service provider or another Customer account with XYZ.

Notwithstanding the specifics of the new contractual provision, Customer and ABC conclude the arrangement includes a license to Product J (i.e. the contract is not a SaaS arrangement) and does *not* include hosting services. This is because ABC transferred control of the license to Customer A when ABC delivered the license to *Customer's* hosting agent (XYZ).

2.6 Combining contracts

Chapter B of KPMG Handbook, Revenue for software and SaaS, addresses combining contracts with customers for software vendors and cloud service providers.

Question 2.6.10

Do customers need to combine vendor contracts entered into at or near the same time with the same counterparty?

Background: Entities (customers) frequently engage a single service provider to perform multiple services.

For example, in a CCA, the entity and a cloud service provider may separately paper and execute contracts for access to the cloud-based solution (i.e. hosting services) and implementation services. Similarly, an entity and a software vendor may separately paper and execute the software license and a contract (e.g. a statement of work) for implementation services.

In other scenarios, implementation services (for internal-use software or a cloud-based solution) contracted for by the customer with a third party (e.g. a consultant) that is not the cloud service provider or the software vendor include more than one legal contract.

Subtopic 350-40 does not include explicit contract combination guidance. Therefore, the question arises about whether the entity must combine two or more contracts with the same counterparty when applying Subtopic 350-40.

If an entity does not combine two or more related contracts with the same counterparty and the prices in each contract do not reflect the stand-alone prices of the license and/or services contracted, the accounting will differ from what would result if the contracts were combined.

For example, an entity contracts with a third party consultant to perform configuration and data migration services for a new CCA in two separate statements of work. The costs of the configuration services will generally be capitalized under Subtopic 350-40 (see section 3.2.10), while the data migration service costs will be expensed as incurred (see section 3.2.20).

If the two statements of work are treated as a single contract, the entity will allocate the combined fees to the configuration and data migration services on a relative stand-alone price basis. In contrast, if the two contracts are not combined, the entity will follow the stated prices in the contract when determining the costs that should be capitalized and those that should be expensed as incurred.

Interpretive response: Yes. Although Subtopic 350-40 does not contain contract combination guidance, we believe it is appropriate to combine two or more contracts entered into at or near the same time that are in substance part of the same commercial arrangement – e.g. a CCA (or licensing arrangement) with related implementation services provided by the cloud service provider (or software vendor). An entity's accounting for the elements of an arrangement should not differ based solely on how the arrangement is papered – i.e. in one contract or multiple contracts.

2.7 Multiple-element arrangements

2.7.10 Subtopic 350-40

Excerpt from ASC 350-40

20 Glossary

Standalone Price

The price at which a customer would purchase a component of a **contract** separately.

30 Initial Measurement

General

> Multiple-Element Arrangements Included in Purchase Price

30-4 Entities may purchase internal-use computer software from a third party or may enter into a **hosting arrangement**. In some cases, the price includes multiple elements, such as the license or hosting, training for the software, maintenance fees for routine maintenance work to be performed by the third party, data conversion costs, reengineering costs, and rights to future upgrades and enhancements. Entities shall allocate the cost among all individual elements. The allocation shall be based on the relative **standalone price** of the elements in the contract, not necessarily separate prices stated within the contract for each element. Those elements included in the scope of this Subtopic shall be accounted for in accordance with the provisions of this Subtopic.

Implementation Costs of a Hosting Arrangement That Is a Service Contract

30-5 An entity shall apply the General Subsection of this Section as though the **hosting arrangement** that is a service contract were an internal-use computer

software project to determine when implementation costs of a hosting arrangement that is a service contract are and are not capitalized.

When an entity purchases internal-use software from a third party or enters into a hosting arrangement with multiple elements (e.g. a license and PCS, or SaaS and implementation services), the consideration in the contract is allocated to the separate elements on a relative stand-alone price basis. The stand-alone price of an element is the price at which a customer would purchase that component separately. [350-40 Glossary, 350-40-30-4]

Question 2.7.10

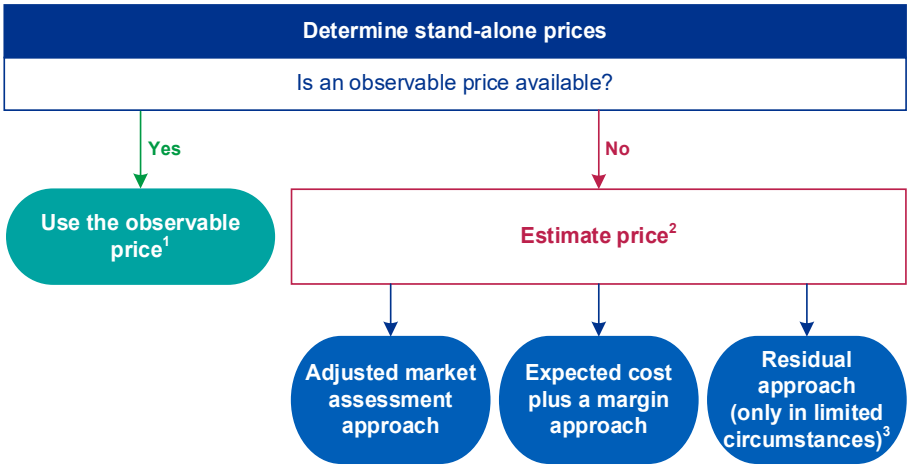
How does an entity determine the stand-alone price of an element in an internal-use software or cloud computing arrangement?

Background: While Subtopic 350-40 requires entities to allocate costs in internal-use software licensing arrangements and CCAs based on relative stand-alone prices, it does not provide guidance about how to determine stand-alone prices.

Interpretive response: We believe the FASB and the EITF intended for stand-alone prices to be determined by customers in internal-use software licensing arrangements and CCAs in substantially the same manner that lessees do under Topic 842 (leases).

This is based on our observations of the EITF deliberations of ASU 2018-15 and other parallels drawn between the then-recently issued Topic 842 and the amendments in ASU 2018-15, such as between the lease term and the 'term of the hosting arrangement' (i.e. the amortization period for CCA implementation costs; see section 6.2.20).

The following diagram reflects the process for determining a stand-alone price.



Notes:

1. An observable stand-alone price is the price charged by the software vendor or similar vendors for a similar element – i.e. a similar license or service under similar terms and conditions, for example with respect to duration and payment terms – on a stand-alone basis.
2. An entity considers all information that is reasonably available when estimating a stand-alone price – e.g. market conditions and entity-specific factors. An entity also maximizes the use of observable inputs and applies consistent methods to estimate the stand-alone price of elements with similar characteristics.
3. See Question 2.7.30.

Question 2.7.20

Is an entity always required to account for contract elements separately?

Interpretive response: Yes. When an internal-use software licensing arrangement or a CCA includes multiple elements, they must be separated.

Unlike the lessee guidance in Topic 842 (see section 4.4.1 of KPMG Handbook, Leases), there is not a practical expedient in Subtopic 350-40 that permits an entity not to separate elements in an arrangement. [842-10-15-37]

Question 2.7.30

When can an entity use the residual approach to estimate stand-alone price?

Interpretive response: We believe four criteria need to be met for an entity to use the residual approach to estimate the stand-alone price of a component of a contract. They are consistent with those that need to be met for a lessee to use the residual approach when applying Topic 842 (see Question 2.7.10).

Criterion 1: Highly variable or uncertain stand-alone price

Using the residual approach to estimate the stand-alone price of an element is appropriate only if the stand-alone price of the element to which the approach would be applied is highly variable or uncertain. [842-10-15-33(a), ASU 2016-02.BC155(a)]

Stand-alone price is …	If …
Highly variable	The price at which the entity could purchase the same or a substantially similar good or service in the same timeframe is widely varied.
Uncertain	The same or a substantially similar good or service is not, and has not previously been, sold on a stand-alone basis such that its stand-alone price has not been established.

Criterion 2: Other observable data is considered first

The method used to estimate a stand-alone price should maximize the use of observable inputs (or information). Therefore, before using a residual estimation approach, the entity must consider whether another estimation approach that maximizes the use of observable information/inputs, such as observable cost and/or margin information, is more appropriate. [842-10-15-33(a), ASU 2016-02.BC155(a)]

Criterion 3: Residual approach does not produce zero or de minimis stand-alone price

We believe a residual estimation approach is not appropriate if it results in zero or very little consideration being allocated to an element, or to a bundle of elements. It is inconsistent with the view that the element transfers a good or service (i.e. provides benefit to the entity) to conclude that it has no stand-alone value. [606-10-55-269, ASU 2014-09.BC273]

Criterion 4: Observable stand-alone prices for other components

To apply the residual approach, the entity needs to have observable stand-alone prices for the elements of the contract for which the residual approach will not be used to estimate their stand-alone prices.

Residual bundles

If two or more goods or services in a contract have highly variable or uncertain stand-alone prices, an entity may need to use a combination of methods to estimate the stand-alone prices of the elements in the contract. For example, the entity may use:

— the residual approach to estimate the aggregate stand-alone prices for all the elements with highly variable or uncertain stand-alone prices; and then
— another technique to estimate the stand-alone prices of the elements in the residual bundle.

Example 2.7.10
Customer in a multiple-element software licensing arrangement

Customer enters into a contract with ABC Corp. for a perpetual license to ABC's software (Product H), three years of PCS on Product H and implementation services. The implementation services include software installation, configuration, and data conversion. Total contract consideration is $680,000.

The elements of the contract and their stated prices in the contract are as follows:

— Perpetual license to Product H, $300,000 payable upfront.
— PCS, 20% of stated license fee ($60,000) per year for three years, each year paid in advance.
— Data conversion services, $50,000, payable as the services are performed.
— Installation and configuration services, $150,000, payable as the services are performed.

Customer can observe stand-alone prices for the individual implementation services based on (1) observable third-party consulting rates for comparable services and (2) Customer's estimate of the hours each service will take provided by its qualified IT engineers.

Customer cannot observe stand-alone prices for the perpetual license to ABC's proprietary software or related PCS – prices of other software vendors' licenses and PCS would not reflect prices of a substantially similar software license or PCS services.

Therefore, Customer applies a residual approach to the license and PCS as a bundle on the basis that:

— ABC's license pricing is unknown from Customer's perspective, and also understood by Customer to be highly variable based on Customer's knowledge of the marketplace and its negotiations with ABC for this license (i.e. it is clearly subject to negotiation case-by-case); and

— the PCS pricing is based on a percentage of the contractual license fee.

Consequently, Customer first allocates the total contract consideration as follows.

Elements	Stand-alone price	Allocation	Basis
Data conversion	$ 50,000	$ 50,000	Observable
Installation and configuration	150,000	150,000	Observable
License/PCS	480,000	480,000	Residual
	$680,000	**$680,000**	

Customer then allocates the $480,000 as follows:

— $300,000 to the perpetual license; and
— $180,000 to the three years of PCS.

In doing so, Customer considers an annual PCS fee equal to 20% of the stated license fee in a perpetual software license is generally consistent with both (1) its other internal-use software arrangements and (2) pricing quoted from ABC's competitors during the bid process for this new software. Therefore, this allocation reflects an appropriate value relationship between the license and the PCS.

Customer accounts for the elements as follows.

— Customer capitalizes a Product H internal-use software license asset (see section 3.3). Assuming there are no other capitalizable costs (e.g. internal costs that qualify for capitalization), the asset is recorded at $450,000, equal to the allocated license fees of $300,000 plus the $150,000 in capitalizable installation and configuration costs.

— Customer expenses the $50,000 in data conversion service costs as incurred (see section 3.2.20).

— Customer recognizes the $180,000 allocated to the PCS to expense ratably over the three-year PCS period (see Question 3.2.120).

Example 2.7.20

Multiple-element cloud computing arrangement – implementation by cloud service provider

Customer enters into a contract with Cloud Service Provider (CSP) for a three-year subscription to its human resources SaaS offering (HRX). Together with the three-year subscription, Customer and CSP contract for CSP to provide implementation services related to HRX. The contract does not include any renewal or extension options.

The elements of the contract and their contractually stated prices, totaling to $945,000, are as follows.

- Three-year subscription to HRX, with an annual fee of $150,000 each year paid in advance.
- Data conversion services, $100,000 payable as the services are performed.
- Configuration services, $350,000 payable as the services are performed.
- Training of Customer personnel, $45,000 payable after the multiday training sessions are completed.

Customer can observe stand-alone prices for similar services from other cloud vendors and consulting firms. HRX, while proprietary software, has competitor offerings in the marketplace for which Customer obtained pricing quotes during its competitive bid process. There are third-party consulting firms that offer all the implementation services CSP is providing in this arrangement.

The following reflects the stand-alone prices for each element and Customer's relative allocation of the contract consideration to each.

Elements	Stand-alone price	Allocation	Calculation
HRX subscription	$ 450,000	$425,250	(450,000 ÷ 1,000,000) × 945,000
Data conversion	120,000	113,400	(120,000 ÷ 1,000,000) × 945,000
Configuration	375,000	354,375	(375,000 ÷ 1,000,000) × 945,000
Training	55,000	51,975	(55,000 ÷ 1,000,000) × 945,000
	$1,000,000	**$945,000**	

Customer accounts for the elements as follows.

- Customer expenses the HRX subscription costs of $425,250 ratably over the three-year subscription period (see section 3.4).
- Customer expenses the $113,400 in data conversion costs and $51,975 in training costs as incurred (see section 3.2.20).
- Customer capitalizes the configuration costs of $354,375 (see section 3.2.10) and will amortize them to expense over the term of the hosting arrangement (see section 6.2.20). The term of the hosting arrangement is three years because there are no renewal or extension options in Customer's contract with CSP (see Question 6.2.90).

Example 2.7.30

Multiple-element cloud computing arrangement – implementation by third-party consultant

Assume the same HRX subscription with CSP as in Example 2.7.20 – i.e. same contractual terms and conditions, including price. However, Customer contracts with XYZ Consulting Firm, instead of CSP, for the implementation services. Therefore, Customer's contract with CSP is a *single* element arrangement – i.e. the only element of that contract is the HRX SaaS subscription.

Customer's contract with XYZ is a multiple-element arrangement. The elements of the contract between Customer and XYZ and their contractually stated prices are as follows.

- Data conversion services, $110,000 payable as the services are performed.
- Configuration services, $380,000 payable as the services are performed.
- Training of Customer personnel, $50,000 payable after the multiday training sessions are completed.

Customer can observe stand-alone prices for similar implementation services from other consulting firms because there are other third-party consulting firms that offer all the implementation services XYZ is providing in this arrangement.

The following reflects the stand-alone prices determined for each element in Customer's contract with XYZ and Customer's relative allocation of the contract consideration to each (numbers in the table are rounded).

Elements	Stand-alone price	Allocation	Calculation
Data conversion	$120,000	$117,818	(120,000 ÷ 550,000) × 540,000
Configuration	375,000	368,182	(375,000 ÷ 550,000) × 540,000
Training	55,000	54,000	(55,000 ÷ 550,000) × 540,000
	$550,000	**$540,000**	

Customer accounts for the contracts with CSP and XYZ separately as follows.

- Customer expenses the HRX subscription costs of $450,000, which are equal to the stated subscription fees in the single-element CSP contract, over the three-year subscription period (section 3.4).
- Customer expenses the $117,818 and $54,000 in allocated data conversion costs and training costs, respectively, as incurred (see section 3.2.20).
- Customer capitalizes the allocated configuration costs of $368,182 (see section 3.2.10) and will amortize them to expense over the term of the hosting arrangement (see section 6.2.20). The term of the hosting arrangement is three years because there are no renewal or extension options in Customer's contract with CSP (see Question 6.2.90).

2.7.20 Subtopic 985-20

Entities selling licenses to their software or providing cloud computing services (e.g. SaaS) using their software, follow the guidance in Topic 606 (revenue from contracts with customers) in allocating the transaction price of a contract with a customer to multiple elements (i.e. performance obligations). See chapter E in KPMG Handbook, Revenue for software and SaaS.

An entity developing software for external use may enter into a multiple-element arrangement as a customer when developing that software. For example, the entity may engage a third party, rather than using internal resources, to develop the software (see Question 5.3.40).

When the entity is the customer (e.g. purchasing third-party developer services), Topic 606 does not apply, and Subtopic 985-20 does not provide guidance on accounting for costs to develop external-use software incurred from a third party in a multiple-element arrangement.

Question 2.7.40

How does an entity allocate third-party costs in a multiple-element arrangement when applying Subtopic 985-20?

Background: An entity may engage a third party to develop a new external-use software product. The contract with the third party may include fees for development services (e.g. coding and testing) and non-development services. Non-development services could include training of vendor personnel by the third party, other consulting activities or hosting services for the software during its development.

Interpretive response: In the absence of multiple-element arrangement guidance in Subtopic 985-20, we believe an entity developing external-use software should analogize to the multiple-element arrangement guidance in Subtopic 350-40 (see section 2.7.10) when allocating third-party costs between development and non-development activities.

Question 2.7.50

How does an entity determine if a set of software programs is a single software product or multiple software products?

Background: Under Subtopic 985-20, a software product: [985-20-55-1]

- is complete and has exchange value; and
- as software, it is a set of programs that interact with each other. A program is further defined as a series of instructions or statements that cause a computer to work.

Determining whether an entity's development efforts are for one or multiple software products may affect one or more of the following:

- when technological feasibility is established (see section 5.2) – i.e. the date each product's technological feasibility is established will likely differ from the date technological feasibility would be established for a single, combined product;
- amortization period (see section 6.4.10) – i.e. two or more products may not have the same economic life or the same economic life a single, combined product would have;
- amortization method (see section 6.4.10) – the unit of account (i.e. the software product) may affect which amortization method (ratio or straight-line) applies in a given annual period; and
- impairment (see section 6.4.30) – the unit of account may affect whether an NRV writedown is required – i.e. one product's NRV may mask an impairment of another if they are accounted for as a single product.

Interpretive response: Determining whether software is a 'software product' in accordance with paragraph 985-20-55-1 often involves judgment. In making that judgment, we believe the software's functionality and the entity's marketing plans for the software will significantly influence the conclusion.

For software to be 'complete' and have its own 'exchange value' as described in paragraph 985-20-55-1, we believe it cannot be functionally dependent on other software. If the software being evaluated cannot function without other software, it likely cannot be considered complete and would not have independent exchange value (i.e. independent marketability).

An entity's marketing efforts and plans may also indicate whether software is a software product. Software that is or will be priced and marketed separately may be a software product. Separate pricing and marketing of software, even if it is not always priced or marketed separately in all markets (e.g. in all geographies or with all customer classes) may indicate that the software is a complete product that has exchange value on its own.

2.8 Software data costs**

2.8.10 Accounting for data costs: The basics**

The rise of AI has put particular focus on data's role in the training (and re-training) of AI models. Data is fundamental to AI software development, continuous improvement and maintenance. Large datasets are obtained/collected (e.g. licensed from unrelated data providers) and used to train underlying AI models, enabling them to recognize patterns, make predictions and perform tasks with a high degree of accuracy.

The use of data is also fundamental to nearly all types of software (AI or otherwise). For example, data is a crucial input to the key outputs (e.g. reports, analytics, visualizations) of many software applications.

For the most part, data is either internally generated or acquired via licensing agreement. Entities generally acquire rights to use data from third parties rather than purchase data outright. US GAAP does not contain specific guidance on the accounting for data. The FASB's Emerging Issues Task Force (EITF) discussed the accounting for database content acquired to which others would be sold access in Issue No. 00-20, *Accounting for Costs Incurred to Acquire or Originate Information for Database Content and Other Collections of Information*. However, no consensus was reached, and the FASB has not subsequently addressed the topic.

In the absence of specific guidance, an entity follows the general guidance that applies to intangible assets more broadly. Thereunder, costs to (1) generate data solely for a particular R&D project or (2) license data solely for an R&D project and for which there is no alternative future use are expensed as incurred under Subtopic 730-10 on R&D. [730-10-25-2(c)]

Subtopic 350-30 applies to any other internally generated or licensed data. Under Subtopic 350-30, internally generated data costs are generally expensed as incurred, while licensed data is capitalizable, initially measured at its cost to the entity (including any transaction costs), because it arises from identifiable, contractual-legal rights granted by the data licensor. [350-30-25-1, 25-3 – 25-4, 30-1]

Once licensed data is recognized, Section 350-30-35 governs its subsequent accounting. The licensed data is amortized over its 'useful life' to the entity, which is the period over which the data is expected to contribute directly or indirectly to the entity's future cash flows, in a manner that reflects how the entity will consume its benefits. Useful life is not the same concept as economic life or productive life; useful life is an entity-specific determination, which may involve judgment, about the period the asset is expected to contribute to the entity's future cash flows. Paragraph 350-30-35-3 provides a non-exhaustive list of factors that should be considered when estimating an intangible asset's useful life. [350-30-35-1 – 35-3, 35-6]

Licensed data, like any other amortizable intangible asset, is subject to the guidance in Section 360-10-35 on the impairment or disposal of long-lived assets – see KPMG Handbook, Impairment of nonfinancial assets, for guidance. [350-30-35-14]

The basics above notwithstanding, the accounting for licensed data is affected by whether it is used in training AI software. In the two sub-sections that follow, we discuss accounting for both (1) data used to train/re-train AI software and (2) data that is *not* used for that purpose. There are different accounting considerations for these two different types of data.

Question 2.8.10**

How are data-related infrastructure costs accounted for?

Background: An entity will typically incur infrastructure costs to store and use data. These costs may include (not exhaustive): purchased or leased computer hardware (e.g. servers, networking equipment); purchased or leased data center space; and third-party infrastructure (e.g. cloud service provider) and

telecommunications costs, which may or may not include leases not explicitly identified as such (i.e. embedded leases). These costs may be significant for some entities given the extensive data requirements often necessary for AI software development and maintenance.

Interpretive response: Such costs are accounted for under Topic 360 (for any purchased facilities or equipment), Topic 842 (for any identified leases), Subtopic 350-40 (for any cloud-based infrastructure service implementation costs) and other US GAAP applicable to service arrangements in general.

Question 3.2.57 and Question 3A.2.230 specifically address whether data-related infrastructure costs (e.g. depreciation of equipment under Topic 360, lease cost recorded under Topic 842, IaaS service fees) are direct or indirect costs of internal-use software development and, therefore, whether those costs *may* be capitalizable as part of an internal-use software asset rather than expensed as incurred. As stated in section 3.2.30, direct costs of eligible internal-use software and cloud computing arrangement development and implementation activities are capitalizable costs, while indirect costs are not.

2.8.20 Data not used to train or retrain AI software**

Software (AI or otherwise) may access and use data, such as medical or trade data, to produce its outputs (e.g. reports that display or analyze that data), that has no role in training or maintaining (e.g. fixing bugs in) the software's functionality

Question 2.8.20**

What accounting guidance applies to data that software uses to produce outputs but not to train or re-train itself?

Interpretive response: We believe an entity follows the *general* guidance that applies to intangible assets broadly (see section 2.8.10), and that the US GAAP software development guidance does not apply to the costs of such data.

2.8.30 Data used to train or retrain external-use AI software**

Section 2.4 addresses whether software is internal- or external-use software. The considerations about whether software is internal- or external-use do not differ for AI software.

Question 2.8.30**

What accounting guidance applies to *external-use* AI software training data?

Background: Chapter 5 discusses the accounting for external-use software development costs in detail. Various references to guidance in that chapter are included in the interpretive response that follows.

Interpretive response: We believe it depends on whether the data has alternative future use. An intangible asset has an alternative future use when (adapted from Question 2.3.80 in KPMG Handbook, Research and development):

— the entity reasonably expects to use it in an alternative manner (e.g. currently identifiable future external-use software development or future maintenance of a different software product) and expects to derive an economic benefit from that alternative use; and

— this alternative use is not contingent on further development of the intangible asset (i.e. the licensed data can be used in the alternative manner in its present condition at its acquisition date).

Data with no alternative future use

Under Subtopic 730-10, the cost of an acquired intangible asset without alternative future use is expensed as incurred. Therefore:

— the costs of licensed data used in developing the AI software *before* technological feasibility is established are expensed as incurred, even though such costs may otherwise qualify for capitalization under Subtopic 350-30; and [730-10-25-2(c)]

— the costs of licensed data used in the further development of AI software *after* technological feasibility is established (see section 5.2) would, we believe, generally qualify as 'production costs' of the AI software under Subtopic 985-20 and therefore be capitalized (see section 5.3.10). [985-20-25-1, 25-3]

As discussed in section 5.2 and Question 5.3.20, for some entities, their software development process may establish technological feasibility only very late in the development cycle, which may lead to very limited, if any, production cost capitalization related to their external-use software.

Data that has an alternative future use

Licensed data *with* an alternative future use is capitalized and subsequently accounted for under Subtopic 350-30 (see section 2.8.10). Given that the licensed data has alternative future use, it may be that the data is being used for multiple purposes concurrently (e.g. to train new AI software and to retrain an existing AI model, to train both an external- and internal-use AI software application, or to train new AI software and serve as content for a non-AI software application). Therefore, amortization may be allocated between the multiple projects/uses, with each allocation accounted for differently (e.g. some allocated amortization capitalized into the cost of a capitalized software asset, another portion recorded as an expense).

— Amortization of the licensed data asset attributable to AI software for which technological feasibility has not been established is recorded as R&D expense. [730-10-25-2(c), 985-20-25-1]

— Amortization of the licensed data asset attributable to a software product for which technological feasibility has been established is generally capitalizable as a production cost (see section 5.3) of the AI software, even though an allocation of such amortization may be considered an indirect, rather than direct, production cost. As outlined in section 5.3, indirect production costs (e.g. allocations of hardware, third-party infrastructure or developer facility costs) are capitalizable under Subtopic 985-20. [985-20-25-1, 25-3, 25-5]

Product enhancements and maintenance

Once external-use software has been developed and released, an entity may undertake efforts to enhance the software and will almost always undertake efforts to maintain it. As further discussed in section 5.2.20, 'Product enhancements' extend the life or significantly improve an existing software product's marketability. Other updates (i.e. that do not meet the definition of a product enhancement) and activities to correct errors or keep the software updated with current information are considered maintenance. [985-20 Glossary]

Distinguishing product enhancements from maintenance frequently requires judgment. In the context of AI software, this includes assessing whether additional training or retraining of the software creates new functionality (i.e. new abilities) or solely maintains the software's relevancy (e.g. merely permits the software to continue to perform its existing tasks with current data). Involvement of the software development team will often be necessary to make this assessment; involving them early in the process will help to ensure appropriate conclusions are reached.

Data costs incurred for product enhancements

Under Subtopic 985-20, the accounting for product enhancement development is the same as the accounting for new software product development (see section 5.2.20 and Question 5.5.20). Therefore, the same considerations as outlined earlier in this question for new software development apply to costs of licensed data to be used in developing software product enhancements. [985-20-55-18]

Data costs incurred for maintenance

Software maintenance costs, including data costs incurred to maintain AI software, are expensed as incurred (see section 5.3.40). [985-20-25-6]

2.8.40 Data used to train or retrain internal-use AI software**

Section 2.4 addresses whether software is internal- or external-use software. The considerations about whether software is internal- or external-use do not differ for AI software.

Question 2.8.40**

What accounting guidance applies to *internal-use* AI software training data?

Background: Chapter 3 (pre-ASU 2025-06) and chapter 3A (post-ASU 2025-06) discuss the accounting for internal-use software development costs in detail. Various references to guidance in those chapters are included in the interpretive response that follows.

Interpretive response: We believe it depends on whether the data has other uses. 'Other uses' in this context means the data can be used for more than one purpose by the entity; it is not equivalent to 'alternative future use' as that term is used in Subtopic 730-10 on R&D. This is because Subtopic 350-40 does not define internal-use software development as an R&D activity.

Data with other uses

Licensed data may have one or more uses to the entity other than in training a specific AI internal-use software application. For example, the entity may be developing, or have a clear intention to develop, multiple applications in which it will use the data, possibly in varying stages of development and with differing likelihoods of successful deployment, or the entity may have existing, already developed AI software for which the data is necessary to its maintenance.

If licensed data has other uses to the entity, it is capitalized and subsequently accounted for under Subtopic 350-30 (see section 2.8.10). No portion of the licensed data asset amortization is capitalized as part of an AI internal-use software asset under Subtopic 350-40 because any such allocation would be an *indirect* cost, ineligible for capitalization.

Section 3.2.30 (before adopting ASU 2025-06) or section 3A.2.50 (after adopting ASU 2025-06) discuss the differential accounting for direct versus indirect costs under Subtopic 350-40 and provide guidance on distinguishing between the two. Question 3.2.57 and Question 3A.2.230 discuss our view that data infrastructure costs (see Question 2.8.10) are typically *indirect* costs.

For clarity, we observe here that neither the treatment, nor the determination, of direct versus indirect costs is changed by ASU 2025-06.

Data with no other uses

The accounting for costs of licensed data with no use to the entity other than developing or maintaining specific internal-use AI software depends on where the software resides in the development lifecycle when those costs are incurred.

— Costs incurred before the Subtopic 350-40 capitalization criteria[1] are met are expensed as incurred. [350-40-25-1, 25-12]

— Costs of licensed data to be used in software development (e.g. to train the AI model's core functionalities) incurred *after* the Subtopic 350-40 capitalization criteria[1] are met, but before the software is 'substantially complete and ready for its intended use' (see section 3.2.40 or section 3A.2.30), are capitalized as part of the AI internal-use software asset. [350-40-25-2, 25-12, 25-14]

— Costs incurred after the AI software is 'substantially complete and ready for its intended use' are expensed as incurred (unless they are capitalizable upgrade or enhancement costs – see discussion below). [350-40-25-6]

Note:

1. The first two bullets above refer to the 'Subtopic 350-40 capitalization criteria'; this wording is intentional because whether pre- or post-adoption of ASU 2025-06, there are criteria that must be met before capitalization can begin (see section 3.2.40 before adopting ASU 2025-06 and section 3A.2.30 after adopting ASU 2025-06). And while those criteria change pre- and post-adoption, the accounting outlined before and after meeting the applicable capitalization criteria does not.

Data licensing arrangements can span many years, with new/changed data made available throughout the data license period; therefore, data costs may be incurred under the arrangement at different times during the software's development lifecycle. This could lead to costs incurred under a single data licensing arrangement being accounted for differently (i.e. some costs expensed as incurred, and others capitalized into the AI internal-use software asset).

Upgrades and enhancements

Upgrades and enhancements are defined in Subtopic 350-40 as changes to existing internal-use software that result in additional software functionality (i.e. the ability to perform additional tasks). As discussed in sections 3.2.50 and 3A.2.60, modifications that merely extend the useful or economic life of internal-use software, or increase how efficiently the software operates, but do not add to the tasks the software is able to perform are not upgrades or enhancements. [350-40-25-7]

Section 3.2.50 (or section 3A.2.60 after adopting ASU 2025-06) further discusses identifying upgrades and enhancements.

It will often require judgment to determine whether training or retraining AI software creates new functionality or merely maintains existing functionality (e.g. keeps the software current but does not add to the tasks the software can perform).

— Costs of licensed data used to upgrade or enhance internal-use AI software follow the same capitalization requirements as costs of licensed data used in new software development.

— Costs of licensed data used solely to maintain internal-use software are expensed as incurred.

3. Initial recognition and measurement: Internal-use software and CCA implementation costs (pre-ASU 2025-06)

Detailed contents

New item added in this edition: **
Item significantly updated in this edition: #

Examples

3.4 Hosting service fees in a CCA

Question

3.1 How the standard works

Subtopic 350-40 addresses the accounting for the following:

- internally developed internal-use software – i.e. the entity owns the software IP;
- acquired internal-use software licenses; and
- cloud computing arrangement (CCA) implementation costs.

Subtopic 350-40 does *not* address the accounting for components of an internal-use software or CCA outside its scope. While it addresses allocating contract consideration between in-scope and out of scope components (see chapter 2), it generally does not prescribe the recognition and measurement for out-of-scope components. There is often little guidance in US GAAP on the accounting for these other components.

Software development and implementation

The following steps apply to determine what software development and implementation costs are capitalized.

- **Step 1**: Determine the stage of software development during which the cost is being incurred
- **Step 2**: Determine if the activity to which the cost relates is specifically prohibited from capitalization
- **Step 3**: Determine which costs of the activity qualify for capitalization
- **Step 4**: Determine when to start and stop capitalization of eligible costs

The same steps apply regardless of whether an entity is accounting for:

- internally developed internal-use software;
- the development (e.g. customization, modification) or implementation of licensed internal-use software; or
- implementation of a CCA.

This chapter uses the term 'capitalized' (and not 'deferral') in the context of both internal-use software and CCA implementation costs. This terminology distinction is explained in section 3.2.

*AI software development***

While AI software is a rapidly growing and emerging space, AI internal-use software development is not subject to different guidance than non-AI software development.

Certain questions in the sections that follow address AI-specific considerations, but there is no different US GAAP accounting guidance.

Software license fees

When an entity acquires an internal-use software license, it capitalizes:

— an intangible asset for the cost of the acquired license (see chapter 7 on presentation); and
— a liability for any fixed and unpaid license fees as of the beginning of the license period.

Capitalized development and implementation costs become part of the cost-basis of the license intangible asset.

Hosting service fees

The CCA itself is an executory service contract. The hosting service fees (i.e. the ongoing subscription fees) – see 'About this publication' – are accounted for in the same manner as the entity would account for the fees for other services it receives. See section 3.4.

3.2 Software development and implementation

US GAAP does not define or provide a finite list of development and implementation activities. However, the following is a list (not exhaustive) of activities that entities will frequently undertake during the development and implementation of internal-use software (internally developed or licensed) or a CCA.

- integration (developing interfaces between the hosted software and the entity's other systems);
- customization of the entity's other systems or the hosted software;
- configuration, of the entity's other systems or the hosted software;
- installation;
- architecture and design;
- coding;
- testing;
- data conversion or migration;
- training; and
- business process reengineering.

This section addresses the accounting for software development and implementation costs incurred in the following scenarios.

- The development and implementation of new internal-use software – e.g. a new application, a new module/component, an upgrade or enhancement to existing software or development of a software interface that is itself internal-use software.
- Implementation or further development (i.e. customization or modification) of acquired or licensed internal-use software.
- Implementation of a cloud-based solution subject to a CCA.

The section is organized into the following sub-sections as follows.

Section	Description
3.2.10	Stages of software development and implementation
3.2.20	Development and implementation activities for which Subtopic 350-40 prescribes the cost accounting regardless of stage of development or implementation
3.2.30	Which costs of a software development or implementation activity are capitalizable
3.2.40	When eligible capitalization begins and ceases during development or implementation
3.2.50	Costs to develop and implement software upgrades and enhancements
3.2.60	Specific considerations about 'agile' internal-use software development projects

‘Capitalized’ vs ‘deferred’ CCA implementation costs

Many entities refer to the ‘deferral’, rather than ‘capitalization’, of CCA implementation costs. This is generally because the classification and presentation requirements for these costs are more akin to those of other deferred costs (see chapter 7), and also to differentiate those costs from internal-use software costs that may commonly be excluded from non-GAAP depreciation and amortization measures like EBITDA (see Observation ‘CCA implementation cost amortization and EBITDA’ in section 7.2.20).

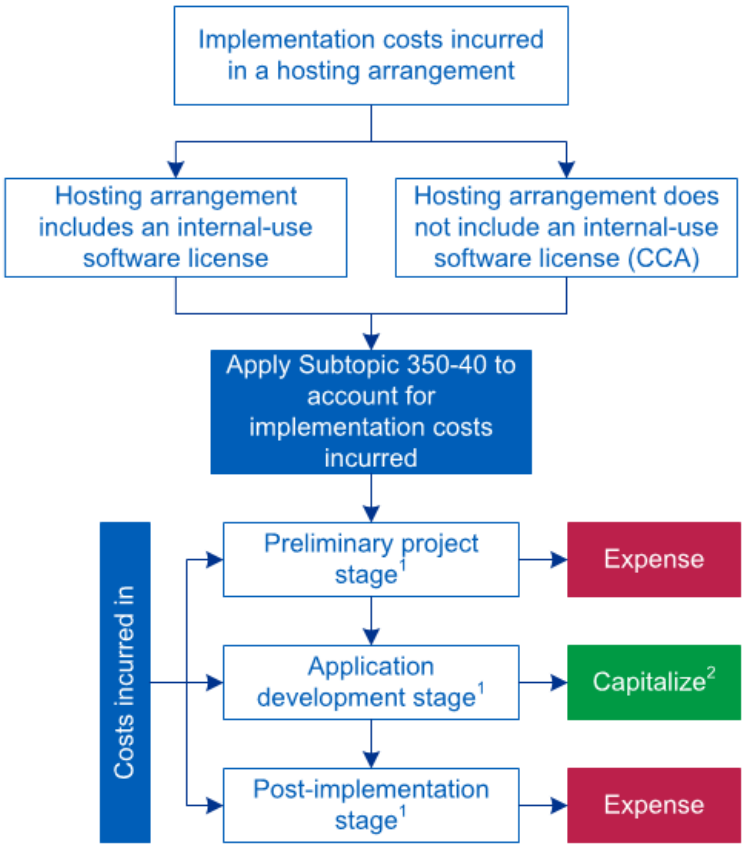
This Handbook uses capitalization, and derivatives thereof, principally to align with the wording in Subtopic 350-40.

Implementation costs incurred in a cloud computing arrangement

Costs incurred to implement a CCA (e.g. configuring the software to the customer’s needs) are capitalized or expensed as incurred using the internal-use software guidance. This section addresses how to apply that guidance. This guidance, enacted by ASU 2018-15, does not affect the accounting by cloud service providers, other software vendors or customers’ accounting for software licensing arrangements.

Significant implementation costs (e.g. configuration costs) *not in the scope of other Topics* are frequently capitalized under the CCA implementation costs guidance, regardless of who undertakes the implementation activities – i.e. internal resources, the cloud service provider or an unrelated third party.

Costs in the scope of other Topics, such as business process re-engineering costs in the scope of Subtopic 720-45, are accounted for under those Topics. The Subtopic 350-40 CCA implementation cost guidance does not permit entities to capitalize costs that are required to be expensed under another Topic or Subtopic.



[1]See section 3.2.10

[2]Amortize capitalized implementation costs – see section 6.2

Question 3.2.10

What is the unit of account for the initial recognition and measurement of internal-use software and CCA development and implementation costs?

Interpretive response: It depends. In some aspects of initial recognition and measurement, the unit of account is the software 'project'. It is the project for which an entity determines the stage of development (see section 3.2.10). If an update to existing software, it is also generally at the project level that the entity determines whether (see section 3.2.50):

— the update is an upgrade or enhancement; and
— if so, whether it is minor or more than minor (affecting whether the guidance in paragraph 350-40-25-10 applies to the project).

In addition, cost capitalization ceases when the project either is (see section 3.2.40): [350-40-25-13 – 25-14]

— no longer probable of being completed and placed into service; or
— substantially complete and ready for its intended use.

Identifying the project may require judgment. We have observed this may especially be the case in some agile development scenarios. Question 3.2.170 addresses considerations relative to identifying the internal-use software 'project' in that context, but we believe are also relevant to non-agile development scenarios.

However, in certain other respects of the Subtopic 350-40 accounting model, each individual software development or implementation cost – whether incurred in connection with a software development project or an internal-use software licensing arrangement – is evaluated to determine whether it should be capitalized or expensed as incurred. In other words, each cost incurred is its own unit of account for purposes of determining whether it is capitalized or must be expensed as incurred. [350-40-25, 30]

For example, even if one concludes the software development *project* is in the application development stage (see section 3.2.10), some costs incurred during that stage are capitalizable (e.g. coding and testing of the software), while others must be expensed as incurred (e.g. data conversion and training costs – see section 3.2.20).

CCA implementation costs

An entity treats the CCA as if it were an internal-use software project. This means that an entity looks at the implementation of the cloud-based solution (or module/component thereof) as the 'project' when determining: [350-40-25-18]

— the project stage (see section 3.2.10); and
— when cost capitalization should begin and cease (see section 3.2.40).

Like internal-use software project costs, each individual CCA implementation cost is its own unit of account for purposes of determining whether it is capitalizable or must be expensed as incurred.

Question 3.2.100 addresses accounting for costs of CCA implementation activities undertaken after go-live.

Multiple modules or components

If an internal-use software project or a cloud-based solution includes multiple modules or components, entities will need to identify the module or component to which development or implementation costs relate.

An entity that inappropriately treats a software application or a cloud-based solution as having a single module/component may not:

— appropriately identify and allocate implementation costs on a relative stand-alone price basis to modules or components that are substantially complete and ready for their intended use on different dates, or appropriately capitalize or expense development or implementation costs;

— begin amortizing capitalized costs at the right time or properly calculate the generally straight-line expense (see Question 6.2.30); or

— properly accelerate cost amortization when a software or cloud-based module or component is planned for abandonment (see section 6.2.30).

3.2.10 Step 1: Determine software development stage

Excerpt from ASC 350-40

05 Overview and Background

General

05-1D The General Subsections of this Subtopic provide guidance on accounting for the cost of computer software developed or obtained for internal use and for determining whether the software is for internal use. Certain costs incurred for computer software developed or obtained for internal use should be capitalized depending on the nature of the costs and the project stage during which they were incurred in accordance with the guidance in Section 350-40-25. Computer software to be sold, leased, or otherwise marketed externally is not considered to be for internal use.

25 Recognition

General

> Preliminary Project Stage

25-1 Internal and external costs incurred during the **preliminary project stage** shall be expensed as they are incurred.

> Application Development Stage

25-2 Internal and external costs incurred to develop internal-use computer software during the application development stage shall be capitalized.

> Postimplementation-Operation Stage

25-6 Internal and external training costs and maintenance costs during the postimplementation-operation stage shall be expensed as incurred.

Implementation Costs of a Hosting Arrangement That Is a Service Contract

25-18 An entity shall apply the General Subsection of this Section as though the **hosting arrangement** that is a service contract were an internal-use computer software project to determine when implementation costs of a hosting arrangement that is a service contract are and are not capitalized.

55 Implementation Guidance and Illustrations

General

> Implementation Guidance

55-4 This Subtopic recognizes that the development of internal-use computer software may not follow the order shown in the preceding list. For example, coding and testing are often performed simultaneously. Regardless, for costs incurred subsequent to completion of the preliminary project stage, the guidance shall be applied based on the nature of the costs incurred, not the timing of their incurrence. For example, while some training may occur in the

application development stage, it should be expensed as incurred as required in paragraphs 350-40-25-2 through 25-6.

Note: The above excerpts have been either amended or superseded by ASU 2025-06 and therefore do not apply once this ASU is adopted (see chapter 3A).

Subtopic 350-40 describes software development and implementation activities as occurring in three stages: [350-40-25-1 – 25-6, 55-3]

- preliminary project stage;
- application development stage; and
- postimplementation-operation stage.

The first step in determining whether the costs of a software development or implementation activity should be capitalized is to identify the stage in which the activity is occurring. This is because Subtopic 350-40 requires costs incurred during the preliminary project and postimplementation-operation stages to be expensed as incurred, while requiring eligible costs (see sections 3.2.20 and 3.2.30) incurred during the application development stage to be capitalized. [350-40-25-1 – 25-2, 25-6]

Question 3.2.20

Is it always necessary to assess the software development stage to determine if an activity cost should be capitalized?

Interpretive response: No. If the costs related to an activity will be accounted for the same way regardless of the software development stage, determining the software development stage is unnecessary. For example, data conversion and training costs are always expensed as incurred (see section 3.2.20), regardless of when they occur during the software development lifecycle. [350-40-25-4]

Preliminary project stage

Excerpt from ASC 350-40

20 Glossary

Preliminary Project Stage

When a computer software project is in the preliminary project stage, entities will likely do the following:

a. Make strategic decisions to allocate resources between alternative projects at a given point in time. For example, should programmers develop a new payroll system or direct their efforts toward correcting existing problems in an operating payroll system?

b. Determine the performance requirements (that is, what it is that they need the software to do) and systems requirements for the computer software project it has proposed to undertake.
c. Invite vendors to perform demonstrations of how their software will fulfill an entity's needs.
d. Explore alternative means of achieving specified performance requirements. For example, should an entity make or buy the software? Should the software run on a mainframe or a client server system?
e. Determine that the technology needed to achieve performance requirements exists.
f. Select a vendor if an entity chooses to obtain software.
g. Select a consultant to assist in the development or installation of the software.

55 Implementation Guidance and Illustrations

General

> Implementation Guidance

55-3 The following list illustrates the various stages and related processes of computer software development:

a. Preliminary project stage:

1. Conceptual formulation of alternatives
2. Evaluation of alternatives
3. Determination of existence of needed technology
4. Final selection of alternatives.

Note: The above excerpts have been superseded by ASU 2025-06 and therefore do not apply once this ASU is adopted (see chapter 3A).

The preliminary project stage generally reflects the entity's activities to consider and define its software needs and possible solutions, regardless of whether it pursues a project to develop or acquire internal-use software, license internal-use software or enter into a CCA. Once the entity has reached decisions on these matters and begins to take concrete actions toward effecting its chosen path, the preliminary project stage has typically ended.

Observation

Preliminary project stage costs are like R&D costs

AcSEC concluded that activities performed during the preliminary project stage of development for internal-use software are analogous to R&D activities. Therefore, costs incurred during this stage should be expensed as they are incurred, consistent with other R&D costs. [SOP 98-1.68]

Specifically, Subtopic 730-10 includes the following examples of R&D activities that AcSEC concluded were consistent with activities normally undertaken during the preliminary project stage: [SOP 98-1.55, 730-10-55-1]

— conceptual formulation and design of possible product or process alternatives; and
— testing in search for or evaluation of product or process alternatives as examples of activities that are R&D.

Application development stage

Excerpt from ASC 350-40

55 Implementation Guidance and Illustrations

General

> Implementation Guidance

55-3 The following list illustrates the various stages and related processes of computer software development:…

b. Application development stage:

1. Design of chosen path, including software configuration and software interfaces
2. Coding
3. Installation to hardware
4. Testing, including parallel processing phase.

Note: The above excerpt has been superseded by ASU 2025-06 and therefore does not apply once this ASU is adopted (see chapter 3A).

The application development stage for a software development project or license/CCA implementation takes place:

— after the preliminary project stage is complete; (see section 3.2.60 for complexities arising from agile software development); and
— before the software or cloud-based solution is ready for its intended use.

Software is ready for its intended use when all substantial testing is completed. [350-40-25-14]

Examples of activities that take place during the application development stage, include: [350-40-55-3(b)]

— designing of the software configuration and software interfaces;
— coding;
— installation to hardware; and
— testing, including that during a parallel processing phase.

Question 3.2.30

When is a cloud-based solution ready for its intended use?

Background: Subtopic 350-40 specifies that internal-use software is ready for its intended use when all substantial testing has been completed. [350-40-25-14]

Interpretive response: Subtopic 350-40 specifies that customers in CCAs should capitalize implementation costs based on the guidance for such costs incurred when developing or implementing internal-use software. Therefore, consistent with developed or licensed internal-use software, a cloud-based solution (or module/component thereof) is ready for its intended use when all substantial testing of the solution is complete. [350-40-25-18]

Postimplementation-operation stage

Excerpt from ASC 350-40

25 Recognition

General

> Capitalization of Cost

25-14 Capitalization shall cease no later than the point at which a computer software project is substantially complete and ready for its intended use, that is, after all substantial testing is completed.

55 Implementation Guidance and Illustrations

General

> Implementation Guidance

55-3 The following list illustrates the various stages and related processes of computer software development:…

c. Postimplementation-operation stage:

1. Training
2. Application maintenance.

Note: The above excerpt from paragraph 350-40-55-3 has been superseded by ASU 2025-06 and therefore does not apply once this ASU is adopted (see chapter 3A).

When internal-use software or a cloud-based solution subject to a CCA is substantially complete and ready for its intended use (see Question 3.2.30), the postimplementation-operation stage begins.

Costs not attributable to a new internal-use software project (including an upgrade or enhancement – see section 3.2.50) or new CCA are expensed as incurred during this stage. [350-40-25-6]

The following are typical postimplementation-operation stage activities (not exhaustive): [350-40-55-3(c)]

— training; and
— application maintenance.

Training on an internal-use software application or a cloud-based solution may not always occur only during the postimplementation-operation stage. However, training costs are expensed as incurred regardless of the project stage during which they are incurred (see section 3.2.20). [350-40-25-4]

3.2.20 Step 2: Does a specific requirement apply to the activity?

Excerpt from ASC 350-40

25 Recognition

General

> Application Development Stage

25-3 Costs to develop or obtain software that allows for access to or conversion of old data by new systems shall also be capitalized.

25-4 Training costs are not internal-use software development costs and, if incurred during this stage, shall be expensed as incurred.

25-5 Data conversion costs, except as noted in paragraph 350-40-25-3, shall be expensed as incurred. The process of data conversion from old to new systems may include purging or cleansing of existing data, reconciliation or balancing of the old data and the data in the new system, creation of new or additional data, and conversion of old data to the new system.

Implementation Costs of a Hosting Arrangement That Is a Service Contract

25-18 An entity shall apply the General Subsection of this Section as though the **hosting arrangement** that is a service contract were an internal-use computer software project to determine when implementation costs of a hosting arrangement that is a service contract are and are not capitalized.

Note: The general section excerpts above have been either amended or moved within Subtopic 350-40 by ASU 2025-06 (see chapter 3A). The amendments are conforming edits, and do not change the accounting requirements.

Although a software development or implementation activity occurs during the application development stage, it may still be required to be expensed as incurred. Subtopic 350-40 specifically prohibits costs of the following activities from being capitalized regardless of when they are incurred: [350-40-25-4 – 25-5, 25-18]

— training; and
— data conversion (or migration – see Question 3.2.40).

The requirement to expense costs of data conversion or migration as incurred does not extend to costs to develop or acquire (including obtaining a license to) software to assist in the activity. [350-40-25-3]

Observation

Expensing training and data conversion costs

AcSEC decided that training costs should be expensed as incurred because: [SOP 98-1.71]

— they are not software development costs; and
— entities are not able to identify the specific future period benefited because they do not control the continued employment of the trained employees.

AcSEC decided that data conversion costs, other than those permitted by paragraph 350-40-25-3, should be expensed as incurred because converting existing data is inherent to the continuing business, rather than creating a new asset for the entity. [SOP 98-1.70]

Question 3.2.40

Should data 'migration' costs be accounted for in the same manner as data 'conversion' costs?

Interpretive response: Yes. Data conversion and data migration are not, strictly speaking, synonyms. The former refers to the process of converting computer data from one format to another, while the latter can refer to the process of transferring data from one system or technology to another without necessarily requiring a format change.

However, the terms are often used interchangeably, including by the FASB staff in its written issue summaries during the development of ASU 2018-15. [EITF Issue Summary No. 1, Supplement No. 1, September 28, 2017]

In addition, we believe AcSEC's basis for deciding that data conversion costs should be expensed as incurred (see Observation 'Expensing training and data conversion costs') is equally relevant for data migration costs.

Therefore, although Subtopic 350-40 only specifically refers to expensing 'data conversion' costs as incurred, we believe the same requirement applies to data migration costs.

Question 3.2.50

Are hosting service fees paid to the cloud service provider in a CCA before completion of implementation activities an implementation cost?

Background: The cloud service provider may initiate the hosting service (i.e. complete the user interface and activate the service) before the customer (or another party on its behalf) completes implementation activities (e.g. specific configurations, desired interfaces with other on-premise or cloud-based applications) that are necessary for the customer's intended use of the cloud-based solution.

In these scenarios, the question arises about whether the customer should capitalize the hosting service fees incurred before the cloud-based solution is ready for its intended use – i.e. during the application development stage of the CCA – as a CCA implementation cost.

Interpretive response: No. The hosting service fees are outside the scope of the 'Implementation Costs of a Hosting Arrangement That Is a Service Contract' subsections of Subtopic 350-40, and therefore are not an implementation cost. The CCA itself is a service contract and the hosting service fees are expensed as the hosting services are provided, consistent with other service contracts (unless the fees can be capitalized as part of the cost of another asset). [350-40-15-4C, ASU 2018-15.BC7]

Question 3.4.10 addresses the commencement of expense recognition for hosting service fees.

3.2.30 Step 3: What activity costs qualify for capitalization?

Excerpt from ASC 350-40

30 Initial Measurement

General

> Capitalizable Cost

30-1 Costs of computer software developed or obtained for internal use that shall be capitalized include only the following:

a. External direct costs of materials and services consumed in developing or obtaining internal-use computer software. Examples of those costs include but are not limited to the following:

    1. Fees paid to third parties for services provided to develop the software during the application development stage
    2. Costs incurred to obtain computer software from third parties
    3. Travel expenses incurred by employees in their duties directly associated with developing software.

b. Payroll and payroll-related costs (for example, costs of employee benefits) for employees who are directly associated with and who devote time to the internal-use computer software project, to the extent of the time spent directly on the project. Examples of employee activities include but are not limited to coding and testing during the application development stage.
c. Interest costs incurred while developing internal-use computer software. Interest shall be capitalized in accordance with the provisions of Subtopic 835-20.

Pending Content

Transition Date: (P) December 16, 2027; (N) December 16, 2027 | Transition Guidance: 350-40-65-4

> Capitalizable Costs

30-1 Costs of computer software developed or obtained for internal use that shall be capitalized include only the following:…

d. Costs to develop or obtain software that allows for access to or conversion of old data by new systems.

30-2 If the entity suspends substantially all activities related to the software developed or obtained for internal use, interest capitalization shall cease until activities are resumed.

30-3 General and administrative costs and overhead costs shall not be capitalized as costs of internal-use software.

> Multiple-Element Arrangements Included in Purchase Price

30-4 Entities may purchase internal-use computer software from a third party or may enter into a **hosting arrangement**. In some cases, the price includes multiple elements, such as the license or hosting, training for the software, maintenance fees for routine maintenance work to be performed by the third party, data conversion costs, reengineering costs, and rights to future upgrades and enhancements. Entities shall allocate the cost among all individual elements. The allocation shall be based on the relative **standalone price** of the elements in the contract, not necessarily separate prices stated within the contract for each element. Those elements included in the scope of this Subtopic shall be accounted for in accordance with the provisions of this Subtopic.

Implementation Costs of a Hosting Arrangement That Is a Service Contract

30-5 An entity shall apply the General Subsection of this Section as though the **hosting arrangement** that is a service contract were an internal-use computer software project to determine when implementation costs of a hosting arrangement that is a service contract are and are not capitalized.

Although a software activity occurs after completion of the preliminary project stage (see section 3.2.10) and the activity is not specifically non-capitalizable (see section 3.2.20), not all costs of, or related to, the activity are necessarily capitalizable.

Direct vs indirect costs

In general, *direct* costs of eligible internal-use software and cloud computing arrangement development and implementation activities are capitalized, while *indirect* costs (i.e. general and administrative and overhead costs) are not. Examples of indirect costs that would generally not be eligible for capitalization include an allocation of lease cost for the space used by the software developers or depreciation of computer servers not dedicated to the development of the internal-use software. [350-40-30-3, 30-5]

Questions 3.2.55 to 3.2.57 address issues we have fielded in practice around whether certain types of costs are direct or indirect costs.

Observation

Direct vs indirect costs

AcSEC excluded indirect costs from the capitalizable cost pool for internal-use software because it concluded a full-costing approach would be complex to apply in an internal-use software context. In doing so, AcSEC acknowledged such costs may be part of the cost of the internal-use software asset and that excluding such costs from the cost basis of an internal-use software asset means its basis will differ in that respect from inventory and property, plant and equipment. [SOP 98-1.80]

Software data costs

With the rise of AI, questions have arisen about the proper accounting treatment of costs to license or otherwise acquire data. Section 2.8 discusses accounting for software data and data infrastructure costs.

Interest costs

Interest costs incurred to develop and/or implement internal-use software or a CCA are capitalized based on the guidance in Subtopic 835-20 (interest capitalization). This includes suspending interest cost capitalization if the entity suspends substantially all its software development or implementation activities for reasons other than those permitted under the Subtopic. [350-40-30-2, 835-20-25-4]

Observation

Interest capitalization rationale

The following contrasting scenarios illustrate the economic rationale underlying the Subtopic 350-40 requirement to capitalize interest costs.

Scenario 1: Third-party software development

An entity engages a third party to develop internal-use software. The development period lasts 18 months and payment is made at completion of development. The fees paid to the third party reflect the implicit financing of the development costs by the third party.

Capitalizing the fees paid to the third party as the cost basis of the internal-use software therefore capitalizes the implied interest element.

Scenario 2: Internal software development

The entity undertakes the same software development project as in Scenario 1 internally. The entity manages the project internally and makes regular payments for payroll and payroll-related costs incurred during development. The entity incurs financing costs (either by borrowing additional funds or using funds that could otherwise be used to repay outstanding debt obligations) during the development phase.

For the internal-use software to reflect a comparable cost basis to that in Scenario 1, the entity must capitalize its interest costs as part of the cost basis of the internal-use software asset.

However, if the entity did not incur interest costs then it follows that none would be capitalized, and the cost basis of the software would differ between the two scenarios.

Questions and answers

Question 3.2.55

Are fees paid to a vendor under an IaaS arrangement capitalizable to an internal-use software project?

Background: Entities may use an IaaS instance to host internal-use software under development. When this is the case, the question arises about whether the IaaS hosting service fees paid to the cloud service provider incurred during the software's application development are capitalizable.

This question does not address the accounting for any implementation costs incurred related to the IaaS arrangement; those costs are subject to the guidance on CCA implementation costs discussed elsewhere in this section 3.2.

Interpretive response: It depends. Determining when IaaS fees are capitalizable direct costs of an internal-use software project, especially as IaaS models continue to evolve, may involve judgment and depend heavily on the specific facts and circumstances. Whether the IaaS fees incurred can be directly attributed to a particular software project such that the IaaS fees are direct costs instead of indirect costs may be particularly important to this analysis. For example:

— If an entity incurs incremental IaaS fees (e.g. under an 'on-demand' pricing structure) for additional, identifiable compute capacity specifically to host internal-use software during application development, we believe those costs would generally qualify as capitalizable, external direct costs under paragraph 350-40-30-1(a)(1).

— By contrast, IaaS fees incurred under a fixed-term (e.g. one or three years), reserved capacity arrangement would likely not qualify for capitalization unless that reserved capacity was obtained specifically for an identified software project and could not be repurposed when the project was completed or abandoned (i.e. before completion). Absent that, any fees allocated to a software project(s) undertaken during the reservation period would, in our view, be an indirect cost akin to the lease cost and depreciation examples in the paragraph that precedes this Question.

Question 3.2.56**

Are fees paid to a third party for access to a Large Language AI Model capitalizable to an internal-use software project?

Background: Entities may use large language AI models (LLMs) to assist in developing internal-use software subject to Subtopic 350-40. These models may support development activities such as code generation, testing, documentation or optimization. LLMs are often accessed via the cloud only, under consumption-based or subscription pricing arrangements. When this is the case, the question arises about whether the fees paid to access the LLM during the application development stage are capitalizable costs of the software project the LLM is being used to complete.

This question does not address the accounting for any implementation costs incurred related to the LLM arrangement (assuming it is a CCA – see section 2.5); those costs are subject to the guidance on CCA implementation costs discussed elsewhere in this section 3.2.

Interpretive response: It depends. Determining whether LLM access fees are capitalizable direct costs of an internal-use software project requires careful consideration of the nature of the arrangement and the specific facts and circumstances. The key considerations are whether the LLM access fees (1) can be directly attributed to a particular software project and (2) represent incremental and direct external costs incurred during the application development stage.

— If an entity incurs incremental LLM access fees (e.g. under a usage or consumption-based pricing model) specifically to support development activities for a particular internal-use software project, those fees may qualify as capitalizable external direct costs. In this circumstance, the usage is analogous to paying for incremental IaaS compute capacity obtained exclusively to develop a particular software project (see Question 3.2.55).

— Conversely, LLM access fees incurred under a subscription model would likely not qualify for capitalization unless the LLM subscription was obtained

specifically for an identified software project and cannot be repurposed when the project was completed or abandoned before completion. Absent that, any fees allocated to a software project undertaken during the subscription period would, in our view, be an indirect cost akin to (1) the lease cost and depreciation examples earlier in this section and (2) the noncapitalizable IaaS fees in Question 3.2.55.

Question 3.2.57**

Are data-related infrastructure costs direct or indirect costs?

Background: See Question 2.8.10 on accounting for data-related infrastructure costs.

Interpretive response: We believe data-related infrastructure costs are generally indirect costs unless they relate to infrastructure obtained specifically for and dedicated to data for a *specific* internal-use software project – e.g. servers or compute capacity obtained specifically and exclusively to host data that has no use to the entity other than training a specific AI internal-use software application.

Question 3.2.60

Is share-based compensation capitalizable?

Background: It may frequently be the case that employees involved in software development or implementation activities earn share-based compensation.

Interpretive response: Yes. While not explicitly stated, we believe it is clear that payroll and payroll-related costs eligible for capitalization under Subtopic 350-40 includes share-based compensation.

Question 3.2.70

Are performance-based share award costs incurred after the application development stage ends capitalizable?

Background: Consider a scenario in which a software developer employee of the entity has performance-based share awards. The employee worked on a significant internal-use software development project for which all substantial testing is complete when it becomes probable the performance-based award target will be met. Assume it does not become probable the target will be reached until shortly before the performance measurement date. The awards

were granted before the project entered the postimplementation-operation stage.

Under Topic 718, the entity did not recognize any compensation cost for the performance-based share awards before achievement of the award target became probable. See paragraph 4.092 in KPMG Handbook, Share-based payment.

In this scenario, the question arises about whether any of the share-based compensation cost should be capitalized to the internal-use software asset for the software development project the employee worked on, assuming the employee spent time working on capitalizable activities (see sections 3.2.10 and 3.2.20) during the application development stage of the project.

Interpretive response: In general, we believe if the compensation cost is not yet incurred under Topic 718 because the performance target is not probable of achievement before the internal-use software development project is substantially complete and ready for its intended use, it should be expensed when it is incurred. [350-40-25-14]

However, if the performance target becomes probable of achievement before the software project is substantially complete, those costs should be capitalized to the extent they relate to capitalizable application development stage activities.

Example 3.2.10

Capitalization of performance-based share award costs

Scenario 1: Performance target not probable of achievement before software is substantially complete and ready for its intended use

ABC Corp. developed internal-use Software Application using internal resources.

On January 1, Year 1, ABC granted Employee restricted stock units (RSUs) that vest if ABC successfully completes an IPO within five years at a specified enterprise valuation. On January 1, Year 1, it was not probable the performance condition would be met.

The following additional facts are relevant.

- ABC completed all substantial testing of Software Application, and it went live in ABC's production environment on January 1, Year 3.
- Between January 1, Year 1 and January 1, Year 3, Employee spent 50% of their work time on application development stage activities that qualify for cost capitalization.
- ABC successfully completed an IPO at a valuation above that required for the RSUs to vest on October 1, Year 4.

— Given the uncertainty associated with successfully completing the IPO and the valuation that would ultimately result, it was not probable that the RSUs performance condition would be met until the IPO was completed.

In this scenario, ABC concludes none of the compensation cost for Employee's RSUs is capitalizable as part of the cost of Software Application. This is because the cost was not incurred under Topic 718 until after all substantial testing of Software Application was completed.

Scenario 2: Performance target probable of achievement before software is substantially complete and ready for its intended use

ABC Corp. developed internal-use Software Application using internal resources.

On January 1, Year 1, ABC granted Employee restricted stock units (RSUs) that vest if Software Application goes live and successfully achieves a specified efficiency target for ABC from its use by the end of its first year in production. On January 1, Year 1, it was not probable the performance condition would be met.

The following additional facts are relevant.

— ABC completed all substantial testing of Software Application, and it went live in ABC's production environment on January 1, Year 3.

— At January 1, Year 3 ABC assesses, based on its testing to-date, that Software Application will meet its design requirements; and therefore, will meet the performance condition – i.e. achieve the specified efficiency target.

— The performance condition is officially achieved on December 31, Year 3.

— Between January 1, Year 1 and January 1, Year 3, Employee spent 50% of their work time on application development stage activities that qualify for cost capitalization.

— Total estimated compensation cost of the award at the grant date was $120,000.

Until January 1, Year 3, ABC recognized no compensation cost for this award because achievement of the performance condition was not probable.

On January 1, Year 3 when the performance condition becomes probable of achievement, $80,000 of compensation cost is recognized.

$120,000 total cost of the award × (24 months since grant date ÷ 36 months total implied service period) = $80,000

Of the $80,000, $40,000 is allocable to the Software Application internal-use software asset. $40,000 is based on Employee's 50% dedication to capitalizable development activities during the period from January 1, Year 1 to January 1, Year 3.

$120,000 total cost of the award × (24 months working on Software Application ÷ 36 months total implied service period) × 50% = $40,000

3.2.40 Step 4: When to begin and cease capitalization

Excerpt from ASC 350-40

25 Recognition

General

> Capitalization of Cost

25-12 Capitalization of costs shall begin when both of the following occur:

a. Preliminary project stage is completed.
b. Management, with the relevant authority, implicitly or explicitly authorizes and commits to funding a computer software project and it is probable that the project will be completed and the software will be used to perform the function intended.

Examples of authorization include the execution of a contract with a third party to develop the software, approval of expenditures related to internal development, or a commitment to obtain the software from a third party.

25-13 When it is no longer probable that the computer software project will be completed and placed in service, no further costs shall be capitalized, and guidance in paragraphs 350-40-35-1 through 35-3 on impairment shall be applied to existing balances.

25-14 Capitalization shall cease no later than the point at which a computer software project is substantially complete and ready for its intended use, that is, after all substantial testing is completed.

Implementation Costs of a Hosting Arrangement That Is a Service Contract

25-18 An entity shall apply the General Subsection of this Section as though the **hosting arrangement** that is a service contract were an internal-use computer software project to determine when implementation costs of a hosting arrangement that is a service contract are and are not capitalized.

Note: Some of the above excerpts have been amended by ASU 2025-06 (see chapter 3A).

Capitalization of eligible costs of a software development project or CCA implementation does not begin before: [350-40-25-12, 25-18]

- the preliminary project stage is complete;
- properly authorized entity management has:
  - authorized the project (explicitly or implicitly); and
  - committed requisite funding for the project; and
- it is probable that both:
  - the project will be completed; and
  - the software will be used for its intended purpose.

Cost capitalization ceases at the earlier of: [350-40-25-13 – 25-14, 25-18]

— concluding it is no longer probable that the software will be completed and placed in service (or that the cloud-based solution will go live); and

— when the software is substantially complete and ready for its intended use (or all substantial testing of the cloud-based solution necessary for it to go live is complete).

Question 3.2.80#

What does 'probable' mean?

Interpretive response: The amendments in ASU 2025-06 explicitly link the term probable in Subtopic 350-40 to the ASC Master Glossary definition (i.e. "the future event or events are likely to occur") – see Question 3A.2.50. Before those amendments, Subtopic 350-40 did not do so. Despite this, many entities have applied 'probable' in a manner consistent with the Master Glossary definition. [ASC Master Glossary]

By contrast, others have interpreted probable consistent with its definition in SOP 98-1 (which is the source of most of the guidance in Subtopic 350-40). We observe in this regard that many entities applied SOP 98-1 for many years before Subtopic 350-40 was created. SOP 98-1 stated that 'probable' has the same meaning as in FASB Concepts Statement No. 6, which stated that "probable is used with its general meaning, rather than in a specific accounting or technical sense…and refers to that which can reasonably be expected or believed on the basis of available evidence or logic but is neither certain nor proved..." [SOP 98-1.62, 75]

Because Subtopic 350-40 (pre-ASU 2025-06) neither (1) explicitly linked probable to the ASC Master Glossary definition, nor (2) codified the SOP 98-1 definition, we believe either interpretation of 'probable' is acceptable in the context of applying the Subtopic before adopting ASU 2025-06, provided it is applied consistently.

Question 3.2.90

Does technological feasibility need to be established before software development costs are capitalized?

Background: Subtopic 985-20 requires all costs incurred to establish the technological feasibility of computer software to be sold, leased or otherwise marketed be charged to expense when incurred. Only once technological feasibility has been established, should subsequent costs be capitalized. [985-20-25-1]

Technological feasibility is established when the entity has completed all planning, designing, coding and testing activities that are necessary to establish

the product can be produced to meet its design specifications, including functions, features and technical performance requirements. [985-20-25-2]

See chapter 5 for guidance on applying Subtopic 985-20.

Interpretive response: No. There is no technological feasibility threshold in Subtopic 350-40. While considered during the development of SOP 98-1, it was ultimately concluded such a threshold was not appropriate for internal-use software. [SOP 98-1.51]

Question 3.2.100

Are costs of CCA implementation activities incurred after go-live capitalizable?

Background: Implementation activities are not necessarily undertaken only at or before go-live. Consider the following examples (not exhaustive).

- Customer implements cloud-based solution T in Year 1. Customer's implementation activities include significant configuration activities before go-live. In Year 2, Customer decides it wants to change the configuration of T, effectively overwriting the configuration effected in Year 1.
- Customer implements cloud-based solution Z in Year 1. Customer's implementation activities include configuration of Z, but not to a significant degree. In Year 2, Customer decides it can incrementally benefit from Z if it more significantly and specifically configures Z.
- Customer implements cloud-based solution X in Year 1. Customer's implementation activities include configuring and interfacing X to work together with its on-premise HR application. In Year 3, Customer sunsets the on-premise HR application for cloud-based HR solution Y. As part of implementing Y, Customer reconfigures X and implements a new interface between X and Y.
- The same facts as the previous example except that Y is a new on-premise HR application.

In these (and similar) examples, the question arises about whether Customer should capitalize the costs to reconfigure T, Z and X and implement a new interface between X and Y under Subtopic 350-40.

Interpretive response: It depends. In general, consistent with the Subtopic 350-40 guidance applicable to internal-use software costs, we believe CCA implementation costs should not be capitalized after go-live. [350-40-25-14]

However, we believe exceptions arise if the related implementation activities:

- relate to a new CCA or new internal-use software application – e.g. Y in the last two background examples;

— create a new asset for which the incurred costs should be capitalized under Subtopic 350-40 or other Topics – e.g. a new interface that meets the definition of internal-use software; or

— increase the functionality of the cloud-based solution.

Activities relate to a new CCA or internal-use software application

If the activities relate to a new CCA that is not yet live or an internal-use software application that is not yet 'substantially complete and ready for its intended use', the related costs that qualify are capitalized.

Judgment may be required to determine whether new activities relate to (1) an existing CCA or (2) a new CCA or on-premise application. However, just because the activities are being undertaken because of implementing a new CCA or internal-use software application does not mean the new activities relate to that new CCA or application.

In general, we would not view changes specifically to the cloud-based software subject to the existing CCA (e.g. changes to its configuration) as related to another CCA or application – even if those changes are only being made because the entity is implementing the new CCA or application.

By contrast, it may require judgment to determine to which CCA or application a new interface (that is not itself an internal-use software asset – see 'Create a new asset' below), implemented to connect the existing CCA with the new CCA or application, relates. It may be relevant in those circumstances to consider where the interface will reside; if it will reside in the entity's IT environment (or its own hosting environment – e.g. its own AWS or Azure instance), it may be a separate internal-use software asset.

— If the new interface will reside in the hosting environment of the existing CCA cloud service provider, this may indicate it relates to the existing CCA.

— Alternatively, if the new interface will reside in the hosting environment of the cloud service provider for the new CCA, this may indicate it relates to the new CCA.

Create a new asset

Some implementation activities create independent internal-use software or PP&E assets. For example, an interface may itself meet the definition of internal-use software, or implementation activities may include the acquisition or construction of assets that meet the definition of PP&E. In those cases, the entity should follow the Topic applicable to the type of asset created (e.g. Subtopic 350-40 or Topic 360).

Additional functionality or utility

Costs of upgrades and enhancements to internal-use software – i.e. costs that result in additional functionality in the software – are generally capitalized if those same costs would be capitalized for new software. [350-40-25-7 – 25-11]

While the 'Implementation Costs of a Hosting Arrangement That Is a Service Contract' subsections of Subtopic 350-40 do not contain equivalent guidance, Subtopic 350-40 explicitly directs entities to refer to the 'General' subsection of

Subtopic 350-40-25 to determine when CCA implementation costs should be capitalized.

We believe implementation costs incurred to substantively increase the functionality or utility of the cloud-based solution – i.e. implementation costs incurred (e.g. configuration changes) to enable the entity to undertake additional tasks or perform additional functions using the hosted software – are analogous to the costs of upgrades and enhancements to internal-use software, and therefore should be capitalized. [350-40-25-18]

In contrast, costs incurred that do *not* result in the entity being able to undertake additional tasks or perform additional functions using the hosted software – e.g. costs incurred that merely change how an existing task or function is performed – should not be capitalized.

Consideration of previous activities

Regardless of whether the costs of new implementation-type activities should be capitalized, we believe consideration should be given to whether those new activities indicate:

- a plan to abandon a module or component of a cloud-based solution (see section 6.2.30);
- a need to reassess the term of the hosting arrangement (see Question 6.2.100); and/or
- the asset group that includes the capitalized implementation costs is impaired (see section 6.2.40).

Question 3.2.110

Do CCA implementation costs incurred by the acquiree give rise to an asset in a business combination or asset acquisition?

Background: Some implementation activities in CCAs give rise to assets that were recognized before the issuance of ASU 2018-15 – e.g. interfaces developed for use in the customer's IT environment generally meet the definition of internal-use software. Such internal-use software assets will be recognized at fair value in acquisition accounting.

However, the question arises about whether costs previously incurred by an acquiree to implement a CCA give rise to an asset that should be recognized in acquisition accounting. Further, some question whether implementation costs that are required to be expensed as incurred under Subtopic 350-40 (e.g. data migration/conversion and training costs) can still give rise to assets in acquisition accounting.

Interpretive response: In general, yes to both.

Business combinations

The acquiree's pre-acquisition implementation activities will typically permit a market participant to avoid incurring both:

— similar implementation costs to derive value from the CCA; and

— hosting service fees, stemming from the acquiree's contract, during the period it would take to implement the cloud-based solution had the acquiree not already undertaken its implementation activities.

In this way, we believe the acquirer obtains a contractually based in-place CCA asset that is like an in-place lease asset, which reflects the value an acquirer lessor obtains from an in-place lease at the acquisition date. An in-place lease generally permits the acquirer lessor to avoid new lease origination and underlying asset holding costs; see Question 11.1.110 in KPMG Handbook, Leases.

The fair value of an in-place CCA, which will depend on the facts and circumstances, should be recognized in the business combination on a CCA-by-CCA basis.

Consistent with our view about in-place lease assets, we believe in-place CCA assets recognized under Topic 805 generally should be reported separately (as an intangible asset) from:

— acquired technology assets (e.g. acquired internal-use software); and
— any favorable contract asset or unfavorable contract liability associated with the CCA arising from off-market hosting service fees.

Asset acquisitions

We believe the above related to business combinations applies equally to asset acquisitions. However, because an entity does not recognize goodwill or a bargain purchase gain in an asset acquisition, the amounts recognized for the implementation costs intangible asset may be adjusted from what would have been recognized under Topic 805.

3.2.50 Upgrades and enhancements#

Excerpt from ASC 350-40

25 Recognition

General

> Upgrades and enhancements

25-7 Upgrades and enhancements are defined as modifications to existing internal-use software that result in additional functionality—that is, modifications to enable the software to perform tasks that it was previously incapable of performing. Upgrades and enhancements normally require new software specifications and may also require a change to all or part of the existing software specifications. In order for costs of specified upgrades and enhancements to internal-use computer software to be capitalized in accordance with paragraphs 350-40-25-8 through 25-10, it must be probable that those expenditures will result in additional functionality.

25-8 Internal costs incurred for upgrades and enhancements shall be expensed or capitalized in accordance with paragraphs 350-40-25-1 through 25-6.

25-9 Internal costs incurred for maintenance shall be expensed as incurred.

25-10 Entities that cannot separate internal costs on a reasonably cost-effective basis between maintenance and relatively minor upgrades and enhancements shall expense such costs as incurred.

25-11 External costs incurred under agreements related to specified upgrades and enhancements shall be expensed or capitalized in accordance with paragraphs 350-40-25-1 through 25-6. If maintenance is combined with specified upgrades and enhancements in a single contract, the cost shall be allocated between the elements as discussed in paragraph 350-40-30-4 and the maintenance costs shall be expensed over the contract period. However, external costs related to maintenance, unspecified upgrades and enhancements, and costs under agreements that combine the costs of maintenance and unspecified upgrades and enhancements shall be recognized in expense over the contract period on a straight-line basis unless another systematic and rational basis is more representative of the services received.

Implementation Costs of a Hosting Arrangement That Is a Service Contract

25-18 An entity shall apply the General Subsection of this Section as though the **hosting arrangement** that is a service contract were an internal-use computer software project to determine when implementation costs of a hosting arrangement that is a service contract are and are not capitalized.

Note: The general excerpts above have been amended and moved to new locations within Subtopic 350-40 by ASU 2025-06 (see chapter 3A). The amendments are minor such that they are not expected to meaningfully change entities' accounting practices.

Upgrades and enhancements are defined as changes to existing internal-use software that result in additional software functionality. Functionality refers to the software's ability to perform a task. [350-40-25-7]

Software development and implementation costs for upgrades and enhancements, including specified upgrades and enhancements to licensed internal-use software, are capitalized or expensed on the same basis as if those costs were incurred to develop and implement new software. [350-40-25-8]

— Costs that would be capitalized when developing or implementing new software are capitalized when developing or implementing an upgrade or enhancement.

— Costs that would be expensed as incurred when developing or implementing new software are expensed as incurred when developing or implementing an upgrade or enhancement.

Specified upgrades

A specified upgrade right is generally a software vendor's explicit commitment to deliver, or agreement to deliver on a when-and-if available basis, a specific version of the software or an upgrade with specific features

and functionality. *Any* discussion in the contract with the customer of possible features and functionality of future versions of the software may give rise to a specified upgrade right.

A software vendor may implicitly grant its customer a specified upgrade right, without there being explicit discussion in the contract. This is the case when the software vendor provides assurance to the customer that a future product release will contain specific features and/or functionality. This might occur, for example, through communication of a detailed software product roadmap (i.e. marketing materials) or in a noncontractual response to a customer's request for proposal.

As a practical matter, claims made in marketing materials available to customers and commitments by sales personnel may be considered by the customer to be part of the arrangement and therefore represent a specified upgrade right. The specific facts and circumstances should be evaluated on a case-by-case basis, and the entity may need to consult with its financial and legal advisers to determine if a specified upgrade right has been granted implicitly to a customer.

Factors to consider when evaluating whether an implicit specified upgrade right has been granted to a customer include the following.

- ***Upgrade/enhancement detail*** – the level of detail of the features, functionality and general release timeframe of the future product that has been provided to the customer. The greater the level of detail and the closer the release date of the enhancement to the initial contract, the greater the likelihood that the software vendor has created an expectation by the customer of the release of the future identifiable upgrade/enhancement that may have affected the customer's purchase decision.
- ***Caveat language*** – whether the software vendor's use of caveat language detailing the product roadmap and future development efforts in a license arrangement gives rise to uncertainty about whether the customer will receive the product upgrades/enhancements. The greater the level of uncertainty about the future delivery of an upgrade/enhancement, the less likely that the software vendor has created an expectation by the customer of the release of the future identifiable upgrade/enhancement.
- ***Customer communication*** – whether the software vendor communicates the features, functionality and timeframe for general release of the future product, or communicates a release to only certain identifiable customers. The broader the intended distribution of the product and the more specific the communication is about functionality and features of the program, the greater the likelihood that customers would expect to receive the specified product upgrades/enhancements.
- ***Software vendor's history*** – a software vendor's history of charging a substantive amount for product upgrades/enhancements would indicate that the product may not be a specified upgrade.
- ***Customer request*** – whether the customer requests or requires a roadmap to specific features and/or functionality that are not available

currently in the marketed product. Such a request or requirement would indicate that the product roadmap may be a specified upgrade.

Example 3.2.20**

Implicit specified upgrade

ABC Corp. issues a request for proposal (RFP) to a software vendor that indicates ABC desires incremental XYZ functionality in the general ledger software it plans to purchase. In a written response to ABC's RFP, the software vendor states that although XYZ functionality currently is not available in Version 4.0 of Product A (software vendor's currently available general ledger software), the software vendor anticipates that XYZ functionality will be available in Version 4.1 of Product A, which is expected to be released in approximately six months.

Subsequent to the RFP process, the software vendor enters into a contract with ABC to transfer a license to Version 4.0 of Product A and to provide ABC with a right to any future updates, upgrades and enhancements to Product A, when-and-if available, for a period of one year. The arrangement does not explicitly discuss the XYZ functionality.

The software vendor has implicitly promised to provide XYZ functionality to ABC through its communication in the RFP. This created a reasonable expectation on ABC's part that the software vendor will deliver the functionality through a future update. Therefore, the XYZ functionality constitutes a specified upgrade.

Maintenance costs

Internal costs to maintain internal-use software – i.e. to maintain its existing functionality and ensure proper operation – are expensed as incurred and are accounted for separately from costs to develop and implement upgrades and enhancements. [350-40-25-9]

If an entity cannot separate internal costs of maintenance from internal costs of relatively minor upgrades and enhancements on a reasonably cost-effective basis, it expenses all such costs as incurred. When applied, this guidance ensures the entity does not capitalize maintenance costs. Subtopic 350-40 does not define 'relatively minor' or 'reasonably cost-effective'. [350-40-25-10]

External costs to maintain internal-use software – e.g. costs paid to a software vendor to maintain licensed software – are expensed as incurred. [350-40-25-11]

Post-contract customer support

Software licensees typically contract for PCS. PCS generally includes not only software maintenance (e.g. bug fixes and other updates that maintain existing functionality), but also technical support and the right to receive upgrades and enhancements on a when-and-if available (i.e. unspecified) basis.

PCS cost is generally recognized on a straight-line basis over the PCS term, unless another systematic and rational basis is more representative of the pattern in which the services are received. [350-40-25-11]

The right to receive unspecified upgrades and enhancements is accounted for as a service even if the entity has a valid expectation it will receive upgrades and enhancements that will provide additional software functionality. [350-40-25-11]

Specified upgrades and enhancements are not part of PCS and follow the requirements for upgrades and enhancements outlined above.

Question 3.2.120

When is it appropriate to recognize PCS expense on a basis other than straight-line?

Interpretive response: In general, we believe a straight-line basis is appropriate; we would expect recognition on a basis other than straight-line to be infrequent.

This is because the typical PCS elements normally qualify as 'stand-ready' services; the software vendor stands ready to:

- provide technical support when-and-as needed; and
- transfer maintenance updates (including bug fixes), upgrades and enhancements when-and-if developed.

When recognizing expense for a stand-ready service, it is generally appropriate to consider the entity's consistent and equal access to the service throughout the service period. This supports a straight-line expense recognition pattern and would make an other than straight-line pattern – e.g. on the basis of the entity's expected usage of technical support or timing/significance of upgrades and enhancements – generally inappropriate. [350-40-35-13]

The components of PCS may not always be stand-ready services, and an expense recognition pattern other than straight-line expense may be appropriate when they are not.

- If the software vendor promises a defined number of upgrades or enhancements, that would typically suggest the entity has a right to that specified number of upgrades and enhancements, rather than the right to receive a stand-ready service.
- If the entity's right to technical support is for a defined number of support calls or events (or up to a specified number of support events that is substantive – i.e. it is not in excess of any realistic expectation of the entity's use of those services), that would typically suggest the software vendor's technical support obligation is not a stand-ready service.

Combined fees

The components of PCS are typically not priced separately; an entity typically pays one fee for PCS (e.g. as a percentage of the software license fee). If that

is the case, and one (but not all) of the components is not a stand-ready service, we believe it is reasonable to either: [350-40-25-11, 30-4]

- separate the components on a relative stand-alone price basis (see section 2.7) and recognize expense related to each separately; or
- select a single attribution pattern for the combined expense that is reflective of the pattern in which the combined PCS services are received.

An entity should apply its approach consistently to similar circumstances.

Question 3.2.130

Does an upgrade or enhancement include software changes that extend the software's life but do not add functionality?

Interpretive response: No. Under Subtopic 350-40, an upgrade or enhancement must result in additional software functionality (i.e. the ability of the software to perform additional tasks). Activities undertaken to extend the useful or economic life of internal-use software are maintenance activities, the costs of which must be expensed as incurred. [350-40-25-7, 25-9, SOP 98-1.73]

The conclusion that an extension of the useful or economic life of the software is not an upgrade or enhancement differs from that in Subtopic 985-20, which defines a 'product enhancement', in part, as an improvement that extends the life of the software (see chapter 5). AcSEC acknowledged this difference, assessing there to be different considerations for a user of internal-use software versus a software vendor. [985-20 Glossary, SOP 98-1.73]

Question 3.2.140

Are modifications that increase only the efficiency of internal-use software upgrades or enhancements?

Interpretive response: No. Under Subtopic 350-40, an upgrade or enhancement must result in additional software functionality (i.e. the ability of the software to perform additional tasks). Modifications that increase how efficiently the software system operates but do not add to the tasks the software is able to perform are not upgrades or enhancements. [350-40-25-7]

Question 3.2.150

Does modifying software so that it can be hosted and operated in a public cloud or on an additional hardware platform or operating system create 'additional functionality'?

Interpretive response: In general, yes. We believe the ability for software to operate in a public cloud, on an additional hardware platform or with a new operating system is added functionality.

Modifications of this nature differ from merely extending the useful or economic life of the software and differ from maintenance activities. This is because the added ability to be hosted in the public cloud or to operate on a different hardware platform or operating system permits additional uses of the software and typically exists together with – i.e. is incremental to – the software's original ability to be hosted other than in the public cloud or to operate on the initial hardware platform or operating system.

Question 3.2.160

Does the ability to use software in new geographies or with new languages qualify as 'additional functionality'?

Interpretive response: In general, yes. Expanding the software's ability to be used in additional geographies or with additional languages would generally be viewed as additional functionality. The entity now can perform additional tasks using the software – e.g. process transactions in the expansion territories or in the added languages.

3.2.60 Agile software development

Internal-use software developed using an agile software development process (or method) is subject to Subtopic 350-40 even though SOP 98-1 (later codified as Subtopic 350-40) was developed and written before agile became widely used.

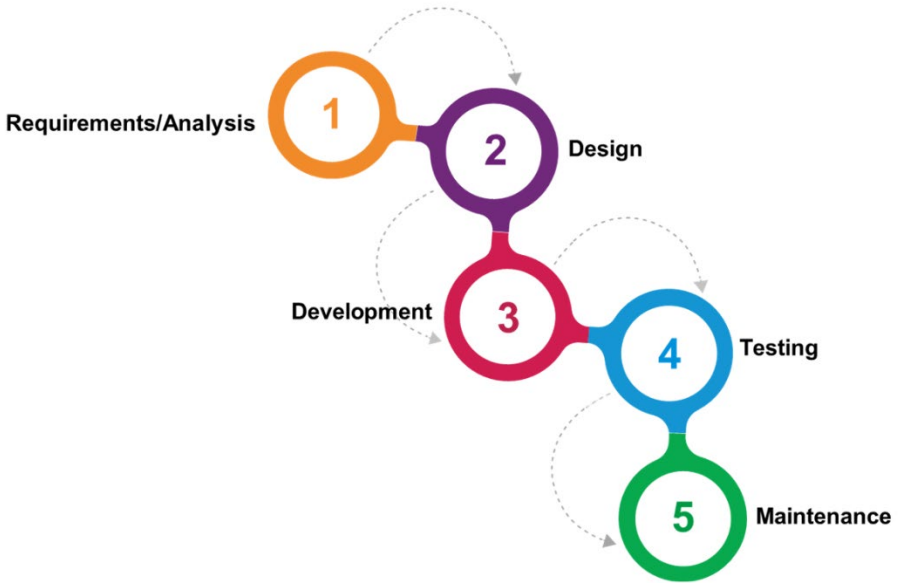
The agile method generally does not align with the Subtopic 350-40 model predicated on identifying discrete software development stages. Therefore, questions arise about how to apply Subtopic 350-40 to agile software development.

This section compares the agile software development method to the waterfall method that more closely aligns to the framework in Subtopic 350-40, and then examines how to apply Subtopic 350-40 to agile software development.

Waterfall method and its influence on ASC 350-40

The two words most used to describe the waterfall method of software development are the synonyms 'sequential' and 'linear'.

The following is a representative illustration of the waterfall software development cycle; sometimes, in alternative representations, the phases are given different, but generally synonymous titles (e.g. verification instead of testing, or implementation versus development), or there may be more than five phases illustrated (e.g. by separating requirements and analysis).



Under the waterfall method, in general, each phase of software development occurs after the one preceding it is completed. A hallmark of the waterfall method is its rigidity in terms of generally not being able to easily revisit a step or activity performed in an earlier phase. In addition, proceeding to the next step or phase typically requires successful completion of the current step or phase.

Subtopic 350-40 (originally, SOP 98-1) was written in the late 1990s when the waterfall method was still predominant in software development. Consequently, it is widely accepted it was written in a waterfall method context. Subtopic 350-40 assumes a sequential (or linear) progression through its three stages (see section 3.2.10). [350-40-25-1 – 25-6, 55-3]

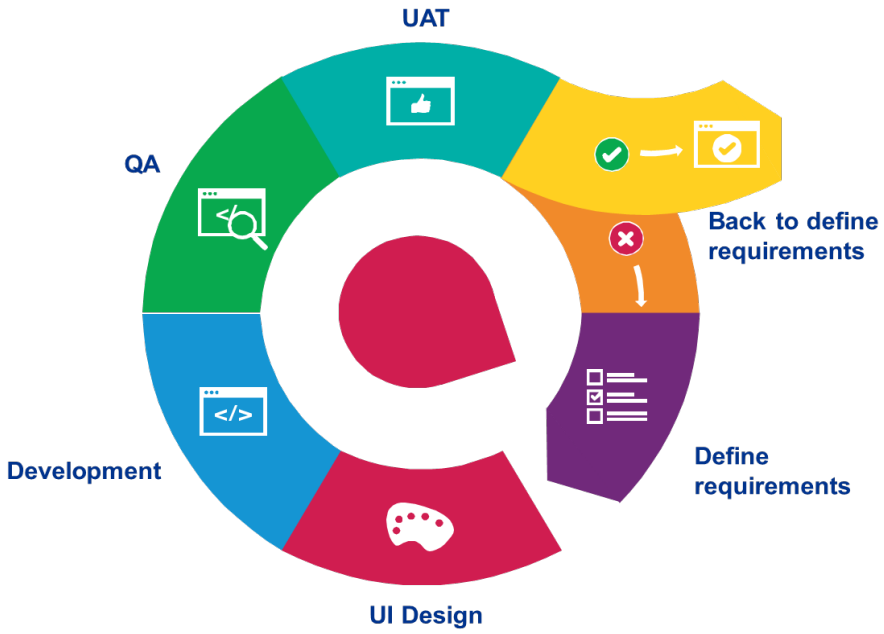
1. **Preliminary project stage.** Formulation and evaluation of alternatives and their viability, and then selection thereof.
2. **Application development stage.** Designing the software requirements (including its configuration and interfaces), coding, testing and installing software to the hardware used to run it.
3. **Postimplementation-operation stage.** Training and application maintenance.

Each of these three stages can be linked to phases of the waterfall method shown in the illustration above: Stage 1 to Requirements/Analysis; Stage 2 to Design, Development and Testing; Stage 3 to Maintenance.

Agile method and its relationship with Subtopic 350-40

Because of the waterfall method presumption in Subtopic 350-40, questions have arisen over time about how to apply Subtopic 350-40 to software development under the agile method, which differs substantially from the waterfall method.

The agile method is designed to be flexible and iterative, rather than (1) heavily pre-planned or rigid and (2) sequential or linear. While there is an overall development objective, agile projects are typically much more lightly planned, and completed through a series of shorter time-frame development 'sprints'. It is understood and accepted that later sprints will frequently drive re-work or revision of tasks completed in previous sprints to arrive at the completed project. One way the agile method can be illustrated follows.



This intended ability to make changes (flexibility) and revisit and/or reperform earlier activities (iteration) often gives rise to the sense that discrete (or distinct) software development phases (or stages) – i.e. that each begin only after the one preceding it ends – do not exist in agile projects. The frequently smaller scale and shorter timeframe of agile projects further contribute to this sense. Finally, the entity's release/update cycle can add to this if it involves the entity concurrently undertaking multiple projects, of different duration and levels of effort or change, related to the same software.

It is this absence of clearly identifiable development phases or stages, which the Subtopic 350-40 model generally envisions there to be, that drives the key questions in practice about applying Subtopic 350-40 to agile software development projects.

Question 3.2.170

How does agile development affect the application of Subtopic 350-40?

Excerpt from ASC 350-40

25 Recognition

General

> Capitalization of Cost

25-12 Capitalization of costs shall begin when both of the following occur:

a. Preliminary project stage is completed.
b. Management, with the relevant authority, implicitly or explicitly authorizes and commits to funding a computer software project and it is probable that the project will be completed and the software will be used to perform the function intended.

Examples of authorization include the execution of a contract with a third party to develop the software, approval of expenditures related to internal development, or a commitment to obtain the software from a third party.

55 Implementation Guidance and Illustrations

General

> Implementation Guidance

55-4 This Subtopic recognizes that the development of internal-use computer software may not follow the order shown in the preceding list. For example, coding and testing are often performed simultaneously. Regardless, for costs incurred subsequent to completion of the preliminary project stage, the guidance shall be applied based on the nature of the costs incurred, not the timing of their incurrence. For example, while some training may occur in the application development stage, it should be expensed as incurred as required in paragraphs 350-40-25-2 through 25-6.

Note: The above excerpts have been amended by ASU 2025-06. Chapter 3A addresses the effect of these amendments on applying Subtopic 350-40.

Excerpt from SOP 98-1

.69 *Application Development Stage*. AcSEC believes that software development activities performed during the application development stage create probable future economic benefits. Therefore, software development costs incurred during this stage should be capitalized.

Interpretive response: While not written for the agile method of software development, Subtopic 350-40 applies to agile projects. Adopting an agile software development methodology does not permit an entity to bypass Subtopic 350-40's requirements or automatically mean that no development costs should be capitalized.

The following reflects a three-step application framework for applying Subtopic 350-40 to agile software development. It may not be the only such framework one could apply; further, judgment and an entity's own facts and circumstances will affect the results from applying it.

Step 1: Identify the unit of account

The first step is to determine the unit of account. As outlined in Question 3.2.10, the primary unit of account for initial recognition and measurement is the software development 'project'. A 'project' can be large (e.g. creating an entirely new software application or module or adding significant new functionality to an existing application) or small (e.g. introducing a single new feature to the software such as a 'Log in with [LinkedIn/Facebook/Google]' option).

The size of the development project does not determine whether or which development costs can or should be capitalized. This is because even a small, single feature development project such as the 'log in with' example above, which could potentially be developed in a single sprint, will generally include capitalizable costs (e.g. coding and testing costs) and non-capitalizable costs (e.g. preliminary project stage costs). Subtopic 350-40 requires capitalization of the eligible costs of even 'minor upgrades and enhancements' (see section 3.2.50) unless those costs cannot be differentiated 'on a reasonably cost-effective basis' from software maintenance costs (see 'Minor upgrades and enhancements' discussion below). [350-40-25-8, 25-10]

Defining the project requires judgment. However, in general, we believe a project is defined by its functional independence from other development efforts – e.g. whether its successful release or deployment depends on the success of other ongoing efforts. When two or more development efforts depend on each other to meet one or more design requirements/objectives, that would typically indicate those efforts are (or are part of, together with other development efforts) a single project, regardless of whether the entity's agile process breaks the development efforts down into smaller milestones (each potentially accomplished through separately defined sprints).

In contrast, we do *not* believe a project should be defined by its size in terms of either (1) development effort (e.g. timeline, labor hours or number of development sprints) or (2) significance (i.e. of the added functionality or overall extent of the changes to the software code).

Multiple units of account

Multiple agile projects related to the same software application may be ongoing concurrently and at different stages of development. While one project related to a single application (or module/component) may be in the preliminary project stage, another project may be past that stage and undertaking capitalizable application development stage activities. In that case, the latter project's eligible

costs are capitalized just as if it were the only ongoing project related to the application.

Minor upgrades and enhancements

We have observed many entities applying an agile development process produce frequent software releases that are often comprised of both maintenance updates and only minor upgrades and enhancements of software features. Those entities may conclude they cannot, consistent with the discussion above, differentiate on a reasonably cost-effective basis the costs of the maintenance updates from the costs to develop the minor upgrades and enhancements with which they are bundled. They may, therefore, expense all the development costs for a particular release of this nature. As noted in section 3.2.50, Subtopic 350-40 does not define 'relatively minor' or 'reasonably cost-effective'; therefore, applying those notions involves judgment. [350-40-25-10]

Step 2: Account for project costs

As outlined in the background preceding this question, it may be difficult to identify discrete software development stages for an agile internal-use software project. When discrete project development stages are not clearly identifiable, we believe entities should generally look to the nature of the development and implementation activities – i.e. whether Subtopic 350-40 characterizes them as capitalizable application development stage activities – when deciding whether the costs of those activities should be capitalized or expensed as incurred. [350-40-55-3]

For example, the entity would capitalize its eligible coding and testing costs for a software project, even when, because of the iterative nature of its agile design process, it cannot identify a distinct beginning or end to the application development or preliminary project stages, respectively, provided the criteria in paragraph 350-40-25-12(b), unrelated to software project development stage, are met.

We believe looking to the nature of the development and implementation activities giving rise to the costs in these scenarios, rather than, for example, expensing all the costs as incurred, is both:

— consistent with AcSEC's intent and conclusion that application development stage activities create probable future economic benefits (and should be capitalized), regardless of whether those activities occur in the sequential order envisioned by SOP 98-1 and its linkage to the then-prevalent waterfall method of software development; and [SOP 98-1.69]

— Subtopic 350-40's reference to applying its capitalization guidance 'based on the nature of the costs incurred, not the timing of their incurrence'. [350-40-55-4]

Is the preliminary project stage ever 'completed'?

The preceding notwithstanding, we have observed entities conclude and disclose that because of the nature of their software applications and their agile development process, the preliminary project stage for new software and significant feature development remains ongoing until just before their software is completed (e.g. made available to the entity's customers on a SaaS basis).

And this contributes – e.g. together with doubts as to the ultimate completion of the software (and therefore, satisfaction of the criteria in paragraph 350-40-25-12(b)) – to a conclusion that they should not capitalize any development costs for these projects.

We believe a preliminary project stage conclusion consistent with that described in the preceding paragraph needs to be supported by relevant evidence – e.g. by a history of substantive preliminary project stage activities, such as the selection of the software's significant features, being revised or finalized for development projects only shortly before that project's completion. It would typically not be appropriate to treat the preliminary project stage as still 'open' or incomplete for most projects merely because of the possibility, due to the iterative nature of the agile method, that preliminary project stage activities will be revisited.

For most entities, we would expect that coding, testing and other application development stage activities will occur after the substantial preliminary project stage activities have been completed.

Step 3: Consider iterative activities

In addition to unit of account and project cost questions, the question arises in the context of agile software development about how to account for those iterative development activities that, in effect, change or override earlier, similar project activities. For example, if after going through quality assurance (QA) or user acceptance testing (UAT) the development team decides to change some of the project design features, does the entity capitalize the eligible application development costs (e.g. coding and testing) to effect the design changes and continue to recognize the previously capitalized application development costs?

We believe this depends on the facts and circumstances. However, in general, all software development has aspects of trial and error to it. Therefore, just because there are design changes or coding must be revised or added for development flaws or design changes to effect the same additional functionality does not mean the entity must expense the previously incurred costs (e.g. those incurred before the QA or UAT that led to the coding change or changed design). Instead, we would more typically expect all the eligible application development stage costs (see sections 3.2.20 and 3.2.30) to be capitalized as costs necessary to complete the project.

Undertaking iterative activities, inherent to the agile method, within a development project differs from a scenario of expecting a subsequent development project to override or replace the feature(s) being developed in the current project. In the latter scenario, the expectation of override or replacement should affect the useful life assigned to the current project software asset. [350-40-25-15, 35-5]

3.3 Internal-use software licenses

Excerpt from ASC 350-40

15 Scope and Scope Exceptions

General

> Transactions

15-4A The guidance in the General Subsections of this Subtopic applies only to internal-use software that a customer obtains access to in a **hosting arrangement** if both of the following criteria are met:

a. The customer has the contractual right to take possession of the software at any time during the hosting period without significant penalty.
b. It is feasible for the customer to either run the software on its own hardware or contract with another party unrelated to the vendor to host the software.

15-4C Hosting arrangements that do not meet both criteria in paragraph 350-40-15-4A are service contracts and do not constitute a purchase of, or convey a license to, software.

20 Glossary

Hosting Arrangement

In connection with accessing and using software products, an arrangement in which the customer of the software does not currently have possession of the software; rather, the customer accesses and uses the software on an as-needed basis.

25 Recognition

General

> Capitalization of Cost

25-17 Entities often license internal-use software from third parties. A software license within the scope of this Subtopic (see paragraphs 350-40-15-1 through 15-4C) shall be accounted for as the acquisition of an intangible asset and the incurrence of a liability (that is, to the extent that all or a portion of the software licensing fees are not paid on or before the acquisition date of the license) by the licensee. The intangible asset acquired shall be recognized and measured in accordance with paragraphs 350-30-25-1 and 350-30-30-1, respectively.

Excerpt from ASC 350-30

25 Recognition

General

25-1 An intangible asset that is acquired either individually or with a group of other assets shall be recognized.

30 Initial Measurement

General

30-1 An intangible asset that is acquired either individually or with a group of other assets (but not those acquired in a business combination) shall be initially measured based on the guidance included in paragraphs 805-50-15-3 and 805-50-30-1 through 30-4.

Excerpt from ASC 805-50

30 Initial Measurement

Acquisition of Assets Rather than a Business

> Determining Cost

30-1 Paragraph 805-50-25-1 discusses exchange transactions that trigger the initial recognition of assets acquired and liabilities assumed. Assets are recognized based on their cost to the acquiring entity, which generally includes the transaction costs of the asset acquisition, and no gain or loss is recognized unless the fair value of noncash assets given as consideration differs from the assets' carrying amounts on the acquiring entity's books. For transactions involving nonmonetary consideration within the scope of Topic 845, an acquirer must first determine if any of the conditions in paragraph 845-10-30-3 apply. If the consideration given is nonfinancial assets or in substance nonfinancial assets within the scope of Subtopic 610-20 on gains and losses from the derecognition of nonfinancial assets, the assets acquired shall be treated as noncash consideration and any gain or loss shall be recognized in accordance with Subtopic 610-20.

30-2 Asset acquisitions in which the consideration given is cash are measured by the amount of cash paid, which generally includes the transaction costs of the asset acquisition. However, if the consideration given is not in the form of cash (that is, in the form of noncash assets, liabilities incurred, or equity interests issued) and no other generally accepted accounting principles (GAAP) apply (for example, Topic 845 on nonmonetary transactions or Subtopic 610-20), measurement is based on either the cost which shall be measured based on the fair value of the consideration given or the fair value of the assets (or net assets) acquired, whichever is more clearly evident and, thus, more reliably measurable. For transactions involving nonmonetary consideration within the scope of Topic 845, an acquirer must first determine if any of the conditions in paragraph 845-10-30-3 apply. If the consideration given is nonfinancial assets or in substance nonfinancial assets within the scope of Subtopic 610-20, the assets acquired shall be treated as noncash consideration and any gain or loss shall be recognized in accordance with Subtopic 610-20.

Entities frequently do not develop internal-use software; they license it from third parties. For example, most entities license their internal-use enterprise resource planning (ERP) software.

Entities may obtain a license to internal-use software through a 'hosting arrangement' if that arrangement is determined to grant the entity a license to the hosted software. If the hosting arrangement does not include a software license, it is a service arrangement. Section 2.5 addresses how to determine if a hosting arrangement includes a software license. [350-40-15-4A – 15-4C]

When licensing internal-use software, the entity recognizes an intangible asset for the software license. Its cost basis includes the sum of the following: [350-40-25-17, 30-4, 805-50-30-1 – 30-2]

- The license fees, which may be an allocated number if there are multiple elements in the arrangement – e.g. license and other services, such as PCS or hosting services in a hosting arrangement that includes a license (see section 2.7).
- Capitalized development – e.g. customization or modification of the software, and implementation costs (see section 3.2).
- Transaction costs, if any.

If all or a portion of the software license fees are unpaid when the license is obtained, a liability is recognized for the unpaid fees. In that case, the initial measurement of the license asset is either: [350-40-25-17, 805-50-30-2]

- the cost basis of the license asset, measured based on the fair value of the license fees liability incurred; or
- the fair value of the acquired license.

The initial measurement basis of the license asset is not a choice; an entity must use the former if the fair value of the license fees liability is more reliably measurable than the fair value of the acquired license (see Question 3.3.30). [805-50-30-2]

Section 6.2.10 addresses the subsequent measurement of intangible software license assets and unpaid license fee liabilities.

Question 3.3.10#

At what date is a new internal-use software license asset and any associated liability recognized?

Interpretive response: Subtopic 350-40 does not provide guidance in this regard. Therefore, we believe it is appropriate to consider the guidance in Subtopic 350-30 on recognizing identifiable intangible assets and the CON 8 definition of an asset. [350-30-25-4, CON 8.E16 – E17]

We believe the license should be recognized when it qualifies as an identifiable asset and the entity (customer) has the present right to the economic benefits of the license.

Identifiable assets include those that meet either the contractual-legal criterion or separability criterion for being an asset. Software licenses will generally meet the contractual-legal criterion; that is, the software license gives rise to contractual rights for the entity stemming from the applicable software license agreement. [350-30-20]

We believe this asset should be recognized *when* it meets the CON 8 definition of an asset for the entity, meaning it constitutes a present right of the entity to economic benefits. [CON 8.E16 – E17]

Any license liability should be recognized at the same time because the entity's financial obligation arises from the obligating event of the software vendor making the software available for the entity's use.

We believe the software license constitutes a present right to economic benefits for the entity when the entity:

a. takes possession, or has the right to take possession, of a copy of the software; and
b. has the right to begin to use and benefit from the license.

We believe (a) occurs when:

— a copy of the software has been physically delivered to the entity;
— the entity has taken possession of the software via download;
— the entity has been provided with the access code (or key) that allows it to take immediate possession of the software; or
— the entity has the present, enforceable right to request an access code (or key) at any time and transfer of such code (or key) is effectively administrative or perfunctory.

We believe (b) occurs at the earlier of:

— the beginning of the contractual license period; and

— when the software is made available for the entity (or any other party on its behalf, including the software vendor) to undertake the development or implementation activities (e.g. entity-specific customizations, installation to the entity's hardware or third-party hosting environment or user acceptance testing) necessary to make it ready for the entity's intended use. Concluding the entity has the right to use and benefit from the software at this point in time is analogous to the conclusion that a lease commences when the underlying asset has been made available to be modified or customized for the lessee's use (e.g. leasehold improvements installed). See section 5.1 of KPMG Handbook, Leases. [842 Glossary, 842-10-55-19 – 55-21]

The contractual license period usually begins before the software is available for the entity to customize or implement it. An entity usually needs to have an in-force license to the software before it can begin either custom development or implementation activities, even if those activities will be undertaken by the software vendor. However, this may not always be the case.

It is possible capitalizable activities will occur before the license qualifies for recognition by the entity. We believe the capitalizable costs of such activities should be capitalized as part of an in-process intangible asset that will include the costs of the software license once it is recognized. This is provided the activities do not give rise to their own asset (e.g. a software interface that qualifies as internal-use software on its own) or constitute activities to ready a *different* asset for its intended use (e.g. costs to ready a computer server for its intended use with the internal-use software). This approach is similar to how an entity constructing a building accounts for costs to clear or grade land before construction of the building structure commences.

Availability of software when the software vendor performs certain activities

For the entity, or a third party (including the software vendor) on its behalf, to install the licensed software or undertake other implementation activities such as commencing data migration or user acceptance testing, it must have a copy of the software.

However, it may be the case that if the software vendor is undertaking customization or configuration activities, it will not provide a copy of the software (or make the software available to the entity for download) before it completes those activities; the vendor provides a copy of the customized or configured software only.

In that case, we believe it is generally appropriate to recognize the software license when those activities commence, just as the entity would if it or a third party were undertaking those activities. This approach is consistent with both of the following.

— The commencement date for a lease does not change if the lessor constructs lessee-owned leasehold improvements, rather than the lessee (or a third party on its behalf). See Example 5.1.10 in KPMG Handbook, Leases.

— Software vendors often recognize software customization services over the customization period on the basis that the customization enhances an asset the customer controls. See Question F200 in KPMG Handbook, Revenue for software and SaaS).

However, because there is no explicit licensee guidance for this scenario, we believe there may be diversity in practice. Some entities may not recognize the software license until the customized/configured software is made available for installation. In the absence of further guidance from the FASB or the SEC staff, and assuming the entity does not have a present enforceable right to take possession of the unmodified/non-configured software, we believe this alternative approach is also acceptable.

Example 3.3.10

Initial recognition and measurement of an internal-use software license asset – license fees prepaid

Customer and Vendor execute a five-year term license on December 1, Year 1. The contractual license term commences on January 1, Year 2, which coincides with Vendor providing the key necessary for Customer to download the software.

Customer prepays the $300,000 license fee and $60,000 for Year 2 PCS on December 31, Year 1.

Customer incurs the following implementation costs related to the software between January 1 and May 31, Year 2. On May 31, Year 2, Customer concludes that the software is ready for its intended use.

This example assumes that all amounts for the license and services (including PCS) from Vendor reflect their stand-alone prices.

Cost	Amount
Installation, configuration and testing[1]	$40,000
Data conversion/migration[2]	15,000
Training[2]	3,000
Software interfacing (Vendor software to other Customer on-premise and hosted applications)[1]	20,000
Total	**$78,000**

Notes:
1. Implementation activities performed by Vendor.
2. Implementation activities performed by Customer or a third-party consultant.

At December 31, Year 1, Customer has recognized a $300,000 prepaid asset and $60,000 in prepaid PCS.

On January 1, Year 2 Customer recognizes the software license on the basis that it has obtained the key necessary to download the software and begin implementation efforts and reclassifies the $300,000 prepaid asset to an in-process intangible software license asset.

Customer capitalizes the $40,000 in installation, configuration and testing costs as part of the cost basis of the software license asset as those costs are incurred between January 1 and May 31.

Customer also capitalizes the $20,000 cost of the software interfaces; those interfaces are capitalized as internal-use software assets separate from the software license asset.

The $15,000 and $3,000 in data conversion/migration and training costs are expensed as incurred during the implementation period.

On May 31, Year 2 (when the internal-use software acquired from Vendor is ready for its intended use), Customer begins amortizing the $340,000 software license asset.

The substantial testing of the new interfaces developed to work with Vendor software is completed at the same time as the testing of Vendor software itself. Therefore, Customer also begins amortizing the software interface assets on that date.

Question 3.3.20

At what date is an internal-use software license asset recognized for a license renewal?

Background: Assume an entity enters into a three-year software license with a software vendor. The term of that license is January 1, Year 1 – December 31,

Year 3. On July 1, Year 3, the parties agree to a three-year extension of that license – i.e. for the period January 1, Year 4 – December 31, Year 6. No renewal option existed in the original license agreement. This is **Scenario 1**.

Consider a variation on this example whereby on July 1, Year 3 the entity formally executes a three-year renewal option that was available to it in the original license agreement (**Scenario 2**).

The question arises about whether the renewal license in these scenarios is a separate intangible asset, and, if so, at what date it (and any related license fees liability) should be recognized.

Interpretive response: Subtopic 350-40 does not provide guidance specific to either background scenario.

Scenario 1: License renewal was not an option in original contract

We believe there is likely diversity in practice, and in the absence of further guidance from the FASB or the SEC staff, we believe either of the following approaches, applied consistently, is acceptable.

Approach 1: Recognize separate renewal license asset only at start of renewal period

Topic 606 treats renewal software licenses as separate and distinct from the initial software license as long as they are not required to be combined under the contract combination guidance. Renewal licenses are not transferred to the customer – i.e. the customer does not obtain control of the renewal license – before the start of the renewal period. Chapter F of KPMG Handbook, Revenue for software and SaaS, discusses this further. [606-10-55-58C(b), 55-392A – 55-392D, ASU 2016-10.BC50(a)]

Under this approach, the entity analogizes to that software vendor guidance. It concludes that the renewal license is not yet a present right of the entity to economic benefits before the start of the renewal period. Therefore, it recognizes the renewal software license only at the start of the renewal period (e.g. January 1, Year 4 in the background scenario). If the entity prepays for the renewal license, it would recognize a prepaid asset, rather than an intangible license asset, until it obtains control of the renewal license at the start of the renewal period.

Approach 2: Recognize modification of the license asset when renewal is agreed

Topic 606 does not apply to the customers in revenue arrangements. Therefore, we believe it is reasonable to not consider the software vendor's accounting when considering that of the entity (customer).

Under this approach, the renewal is accounted for as a modification of the software license (i.e. an extension of the time attribute thereof). The renewal is not treated as a separate or distinct license because the entity already has the rights being renewed and already has a copy of the software.

The cost of the renewal is added to the existing carrying amount of the intangible software license asset on the date the renewal is agreed to by the vendor and the entity (July 1, Year 3 in the background scenario). The useful life

of the license asset is extended by the term of the renewal. The change in useful life is accounted for prospectively (see Question 6.2.20).

Scenario 2: License renewal option included in original license agreement

We believe either approach to Scenario 1 is also acceptable for Scenario 2 as an accounting policy election.

Question 3.3.30

Which is typically more reliably measurable, the fair value of the consideration given or the fair value of the license asset received?

Background: If all or a portion of the software licensing fees are unpaid when the entity recognizes the license (see Questions 3.3.10 and 3.3.20), a liability is recognized for those unpaid fees. In that case, the initial measurement of the license asset is either: [350-40-25-17, 805-50-30-2]

- the cost basis of the license asset, measured based on the fair value of the license fees liability incurred; or
- the fair value of the acquired license.

The initial measurement basis of the license asset is not a choice; an entity must use the former if the fair value of the license fees liability is more reliably measurable than the fair value of the acquired license. [805-50-30-2]

This question assumes none of the following apply (consider the guidance in Chapter A of KPMG Handbook, Revenue for software and SaaS):

- Topic 606 (revenue from contracts with customers) – e.g. the entity is a software vendor and receives the internal-use software license as full or partial payment for granting a license to its software;
- Subtopic 610-20 (gains and losses from the derecognition of nonfinancial assets) – e.g. the entity receives the internal-use software license as payment for the sale of an asset to a non-customer; or
- Topic 845 (nonmonetary transactions).

The question arises about whether the fair value of the consideration given (e.g. the liability to pay the license fees over the license term) is more reliably measurable than the fair value of the acquired license.

The response to this question assumes the unpaid license fees are payable in cash. For guidance that addresses the acquisition of an asset, including an internal-use software license, for consideration other than cash, see section 3.3 of KPMG Handbook, Asset acquisitions.

Interpretive response: We believe in most cases the fair value of the unpaid license fees liability will be more reliably measurable than the fair value of the software license acquired. The generally proprietary nature of a vendor's software and the typically unobservable information about the prices at which it sells licenses to the software contribute to this conclusion.

Question 3.3.40

Should an unpaid software license fees liability be discounted?

Interpretive response: Yes. Ignoring the possibility the effect of discounting may be immaterial, an unpaid license fees liability should be discounted to reflect fair value. [805-50-30-2]

In our experience, the license fees liability will typically be measured at the present value of the amounts yet to be paid under the license agreement using an appropriate discount rate from the perspective of a market participant. [820-10-55-4, 835-30-25-12 – 25-13]

Example 3.3.20

Initial recognition and measurement of an internal-use software license asset – license fees paid over license term

Customer and Vendor execute a five-year term license on December 1, Year 1 for total license fees of $300,000, payable in five upfront annual installments of $60,000. The contractual license term commences on January 1, Year 2, which coincides with Vendor providing the key necessary for Customer to download the software.

Customer prepays the Year 1 $60,000 license fee and $60,000 for Year 1 post-contract customer support (PCS) on December 31, Year 1.

Customer incurs the following implementation costs related to the software between January 1 and May 31, Year 2. On May 31, Year 2, Customer concludes the software is ready for its intended use.

This example assumes all amounts for the license and services (including PCS) from Vendor reflect their stand-alone prices.

Cost	Amount
Installation, configuration and testing[1]	$40,000
Data conversion/migration[2]	15,000
Training[2]	3,000
Software interfacing (Vendor software to other Customer on-premise and hosted applications)[1]	20,000
Total	**$78,000**

Notes:

1. Implementation activities performed by Vendor.
2. Implementation activities performed by Customer or a third-party consultant engaged by Customer.

At December 31, Year 1 Customer has recognized a $60,000 prepaid asset and $60,000 in prepaid PCS.

On January 1, Year 2 Customer recognizes the software license and recognizes an in-process intangible software license asset of $267,906, comprising:

— $207,906 for the unpaid license fees liability (which is also recognized at this date) – present value of the four remaining payments of $60,000 due under the license contract, discounted at 6%; plus

— the $60,000 prepayment made on December 31, Year 1.

Customer capitalizes the $40,000 in installation, configuration and testing costs as part of the cost basis of the software license asset as those costs are incurred between January 1 and May 31.

Customer also capitalizes the $20,000 cost of the software interfaces; those interfaces are capitalized as internal-use software assets separate from the software license asset.

The $15,000 and $3,000 in data conversion/migration and training costs are expensed as incurred during the implementation period.

On May 31, Year 2 (when the internal-use software acquired from Vendor is ready for its intended use), Customer begins amortizing the $307,906 ($267,906 + $40,000) software license asset.

The substantial testing of the new interfaces developed to work with Vendor software is completed at the same time as the testing of Vendor software itself. Therefore, Customer also begins amortizing the software interface assets on that date.

Question 3.3.50

Does the measurement of the software license fees liability include usage-based fees?

Background: In addition to fixed license fees, an entity licensing internal-use software may be required to pay usage- or transaction-based fees. The following are examples:

— An entity licenses software it will use to route customer service tickets to appropriate departments and locations. In addition to a fixed license fee, the entity is also required to pay a fixed fee for each customer service ticket routed using the software.

— An entity licenses research software for use by its employees. In addition to a fixed license fee, the entity pays a usage-based fee for each user that logs into the application each month. If 100 employees log into the application in a given month and the usage-based fee is $100 per user, the entity will owe a usage-based fee of $10,000 for that month.

— An entity licenses accounting software for a fee based on estimated annual revenues of the entity of $100 million. If the entity's annual revenues

exceed $100 million, it will owe an incremental fee based on the excess revenues.

These examples are derived from Examples C390.1 – C390.3 in KPMG Handbook, Revenue for software and SaaS, which contain additional detail.

Interpretive response: In general, no. Subtopic 350-40 requires entities to measure the license fees liability on the basis of the asset acquisition guidance in Subtopic 805-50 (see section 3.5 in KPMG Handbook, Asset acquisitions). [350-40-25-17, 350-30-30-1, 805-50-30-2]

Consistent therewith, we believe usage-based fees should generally be excluded from the initial measurement of the license fees liability. Determining when to recognize those fees as an expense involves judgment. In general, we believe:

— Usage-based fees that are expected to be recurring – e.g. as the entity uses the software throughout the license period – should be recognized as a cost of the period in which they are incurred.

— In contrast, a usage-based fee that is not expected to recur – e.g. a one-time milestone payment – should adjust the cost basis of the acquired license asset on a cumulative effect basis.

An exception may arise in more unusual circumstances, such as if the usage-based fees are either (1) payable in the form of the entity's equity instruments or (2) meet the requirements in Topic 815 (derivatives and hedging). Section 3.5 of KPMG Handbook, Asset acquisitions, discusses this further.

Interim reporting considerations

Entities reporting financial information on an interim basis under Topic 270 (interim reporting) frequently make estimates in assigning costs and expenses to interim periods so that interim period results more closely reflect anticipated annual results. [270-10-45-4(b)]

In the case of usage-based fees based on an annual benchmark, such as in the third background example, an entity may estimate the usage-based fees it will incur for the entire annual period, and accrue a proportion during interim periods. This applies even though the entity will not owe the usage-based fee to the software vendor if the annual benchmark or target is not ultimately met.

Question 3.3.60

Is the interest on the unpaid license fees liability incurred during the application development stage capitalized?

Background: Interest costs incurred while developing and implementing internal-use software are capitalized under Subtopic 835-20. Therefore, if an entity has outstanding borrowings, the historical cost of internal-use software (i.e. its carrying amount) will include financing costs. [350-40-30-1(c)]

In the case of an internal-use software license that will be paid for over time, the license is directly financed through the software vendor.

Interpretive response: It depends. Like software developed for internal use, interest is capitalized during the application development stage for acquired software licenses that require substantial implementation or customization. This includes interest on any unpaid license fees liability. [835-20-05-01, 15-2 – 15-3, 15-5]

However, an entity is not required to capitalize the interest cost for an acquired license if the customization or implementation activities are not substantial. [835-20-15-3, 15-6]

Consistent with other assets subject to interest capitalization under Subtopic 835-20, interest capitalization (if any) should cease if the implementation or customization project is intentionally delayed or is suspended other than for reasons permitted under Subtopic 835-20, even if the project is still in the application development stage. [835-20-25-3 – 25-4, 25-6]

3.4 Hosting service fees in a CCA

Excerpt from ASC 350-40

15 Scope and Scope Exceptions

General

> Transactions

15-4C Hosting arrangements that do not meet both criteria in paragraph 350-40-15-4A are service contracts and do not constitute a purchase of, or convey a license to, software.

20 Glossary

Hosting Arrangement

In connection with accessing and using software products, an arrangement in which the customer of the software does not currently have possession of the software; rather, the customer accesses and uses the software on an as-needed basis.

A CCA is a service contract. Consequently, the hosting service fees – i.e. the subscription fees paid to the cloud service provider for the right to access the hosted software – are accounted for in the same manner as fees for any other stand-ready service the entity receives (e.g. most equipment maintenance or internet access contracts). [350-40-15-4C]

Section 3.2 addresses accounting for implementation costs incurred in a CCA, and section 2.7 addresses allocating arrangement consideration between hosting service fees and implementation costs of a CCA.

Observation

Accounting for hosting service fees

In the summary to ASU 2018-15, the FASB expressed the view that accounting for a CCA as a service contract generally means the hosting service fees should be expensed as incurred. [ASU 2018-15.Summary]

Question 3.4.10

Should hosting service fees in a CCA begin to be recognized as expense if the arrangement term begins before go-live?

Background: Completion of implementation activities frequently requires access to the cloud-based solution – i.e. the implementation activities cannot occur before the customer can access the hosted software. For example, a customer cannot begin to migrate data from its existing system before it has access to the cloud-based solution. Access is also typically required to begin to implement and test interfaces. Therefore, go-live frequently occurs after the contractual CCA term commences and the customer has been provided access to the cloud-based solution.

If the CCA term begins before completion of implementation activities integral to going live with the cloud-based solution, the question arises about whether the customer should begin recognizing the costs of the CCA as expense before go-live.

Question 3.2.50 explains that hosting service fees due under the CCA are not 'implementation costs'. Instead, they are service fees.

Interpretive response: Yes. CCA hosting service fees should begin to be recognized as an expense when the customer (including third-party consultants working on the customer's behalf) obtains access to the cloud-based solution such that it can begin to undertake its implementation activities. The customer begins to consume and receive benefit from the CCA at that time.

For example, the cloud service provider initiates the CCA on January 1, Year 1, such that the customer has access to the hosted software to commence its implementation activities from that date. The customer should begin recognizing expense attributable to the CCA as of that date, even if it does not plan to go live until January 1, Year 2.

This is consistent with how lessees recognize lease cost when they install leasehold improvements (when they are the accounting owner of the improvements). In general, a lease commences (and lease cost begins to be recognized) when the lessee obtains access to the underlying asset to begin installing leasehold improvements to make the asset (e.g. retail space) ready for its intended use. Section 5.1 of KPMG Handbook, Leases, discusses this in further detail.

In addition, we believe expensing the hosting service fees incurred during the implementation phase of a CCA is also supported by the accounting treatment for analogous website hosting fees incurred by a website developer during the application and infrastructure development stage of a website. Under Subtopic 350-50, website hosting fees are generally expensed over the period the hosting services are provided (see section 4.2.20). [350-50-25-5]

Cloud service provider performs the implementation activities

The timing of expense recognition for hosting service fees should not differ based on whether it is the customer (or a third party on its behalf) or the cloud service provider (or a third party on its behalf) that undertakes the implementation activities. While the customer may not be given access to the hosted software during the implementation period if the cloud service provider undertakes the activities, the customer is still consuming and receiving benefit from the CCA as the implementation activities specific to its needs (e.g. customer-requested configurations, customer data migration) are performed, just as if it had obtained access to undertake the activities itself.

This is consistent with the view a lease commences (and lease cost begins to be recognized) when lessee-owned leasehold improvements begin to be constructed even if it is the lessor that is constructing them (rather than the lessee or a third party engaged by the lessee). Example 5.1.10 in KPMG Handbook, Leases, illustrates this.

3A. Initial recognition and measurement: Internal-use software and CCA implementation costs (post-ASU 2025-06)**

Detailed contents

Note: Because the amendments in ASU 2025-06 left most of Subtopic 350-40 unchanged, much of the guidance in this new chapter duplicates guidance in chapter 3. To assist readers that have adopted ASU 2025-06 (or are preparing to) in applying the guidance in this chapter, we have marked *only* questions, examples and observations that are new to this Handbook or significantly modified from those in chapter 3 with the new item and significant update symbols.

New item added in this edition: **
Item significantly updated in this edition: #

3A.2.240 When is it appropriate to recognize PCS expense on a basis other than straight-line?

3A.2.250 Does an upgrade or enhancement include software changes that extend the software's life but do not add functionality?

3A.2.260 Are modifications that increase only the efficiency of internal-use software upgrades or enhancements?

3A.2.270 Does modifying software so that it can be hosted and operated in a public cloud or on an additional hardware platform or operating system create 'additional functionality'?

3A.2.280 Does the ability to use software in new geographies or with new languages qualify as 'additional functionality'?

Examples

3A.2.10 Determining when to begin cost capitalization – new AI data management and analytics software **

3A.2.20 Determining when to begin cost capitalization – new analytics features for existing AI data management software **

3A.2.30 Capitalization of performance-based share award costs

3A.2.40 Implicit specified upgrade **

3A.3 Internal-use software licenses

Questions

3A.3.10 At what date is a new internal-use software license asset and any associated liability recognized? #

3A.3.20 At what date is an internal-use software license asset recognized for a license renewal?

3A.3.30 Which is typically more reliably measurable, the fair value of the consideration given or the fair value of the license asset received?

3A.3.40 Should an unpaid software license fees liability be discounted?

3A.3.50 Does the measurement of the software license fees liability include usage-based fees?

3A.3.60 Is the interest on the unpaid license fees liability incurred during application development capitalized?

Examples

3A.3.10 Initial recognition and measurement of an internal-use software license asset – license fees prepaid #

3A.3.20 Initial recognition and measurement of an internal-use software license asset – license fees paid over license term

3A.4 Hosting service fees in a CCA

Question

3A.4.10 Should hosting service fees in a CCA begin to be recognized as expense if the arrangement term begins before go-live?

3A.1 How the standard works

This chapter is written as if ASU 2025-06, Intangibles—Goodwill and Other—Internal-Use Software (Subtopic 350-40): Targeted Improvements to the Accounting for Internal-Use Software, has been adopted. For guidance on applying Subtopic 350-40 before adopting ASU 2025-06, see chapter 3.

Given the newness of ASU 2025-06, this chapter may be updated in future editions of the Handbook as new questions and/or interpretations emerge, or if the FASB, its staff or the SEC staff provide views or guidance.

Subtopic 350-40 addresses the accounting for the following:

- internally developed internal-use software – i.e. the entity owns the software IP;
- acquired internal-use software licenses;
- cloud computing arrangement (CCA) implementation costs; and
- website development costs.

Subtopic 350-40 does *not* address the accounting for components of an internal-use software or CCA outside its scope. While it addresses allocating contract consideration between in-scope and out of scope components (see chapter 2), it generally does not prescribe the recognition and measurement for out-of-scope components. There is often little guidance in US GAAP on the accounting for these other components.

Software development and implementation

The following steps apply to determine what internal-use software development and implementation costs are capitalized.

- **Step 1:** Determine the unit of account
- **Step 2:** Determine when to start and stop capitalization of eligible costs
- **Step 3:** Determine if the activity to which the cost relates is specifically prohibited from capitalization
- **Step 4:** Determine which costs of the activity qualify for capitalization

The same steps apply regardless of whether an entity is accounting for:

- internally developed internal-use software;
- the development (e.g. customization, modification) or implementation of licensed internal-use software;
- implementation of a CCA; or
- website development costs.

This chapter uses the term ‘capitalized’ (and not ‘deferral’) in the context of both internal-use software and CCA implementation costs. This terminology distinction is explained in section 3A.2.

AI software development**

AI internal-use software development is not subject to different guidance than non-AI software development.

Certain questions in the sections that follow address AI-specific considerations, but there is no different US GAAP accounting guidance.

Software license fees

When an entity acquires an internal-use software license, it capitalizes:

— an intangible asset for the cost of the acquired license (see chapter 7 on presentation); and
— a liability for any fixed and unpaid license fees as of the beginning of the license period.

Capitalized development and implementation costs become part of the cost-basis of the license intangible asset.

Hosting service fees

The CCA itself is an executory service contract. The hosting service fees (i.e. the ongoing subscription fees) are accounted for in the same manner as the entity would account for the fees for other services it receives. See section 3A.4.

3A.2 Software development and implementation

US GAAP does not define or provide a finite list of development and implementation activities. However, the following is a list (not exhaustive) of activities that entities will frequently undertake during the development and implementation of internal-use software (internally developed or licensed) or a CCA.

- integration (developing interfaces between the hosted software and the entity's other systems);
- customization of the entity's other systems or the hosted software;
- configuration of the entity's other systems or the hosted software;
- installation;
- architecture and design;
- coding;
- testing;
- data conversion or migration;
- training; and
- business process reengineering.

This section addresses the accounting for software development and implementation costs incurred in the following scenarios.

- The development and implementation of new internal-use software – e.g. a new application, a new module/component, an upgrade or enhancement to existing software or development of a software interface that is itself internal-use software.
- Implementation or further development (i.e. customization or modification) of acquired or licensed internal-use software.
- Implementation of a cloud-based solution subject to a CCA.

The section is organized as follows.

Section	Description
3A.2.10	Common software development methods
3A.2.20	Unit of account
3A.2.30	When eligible capitalization should begin and cease during development or implementation
3A.2.40	Development and implementation activities for which Subtopic 350-40 prescribes the cost accounting
3A.2.50	Which costs of a software development or implementation activity are capitalizable
3A.2.60	Costs to develop and implement software upgrades and enhancements
3A.2.70	The FASB's implementation guidance and illustrations in ASC section 350-40-55 related to internal-use software and website costs

‘Capitalized’ vs ‘deferred’ CCA implementation costs

Many entities refer to the ‘deferral’, rather than ‘capitalization’, of CCA implementation costs. This is generally because the classification and presentation requirements for these costs are more akin to those of other deferred costs (see chapter 7), and also to differentiate those costs from internal-use software costs that may commonly be excluded from non-GAAP depreciation and amortization measures like EBITDA (see Observation ‘CCA implementation cost amortization and EBITDA’ in section 7.2.20).

This Handbook uses capitalization, and derivatives thereof, principally to align with the wording in Subtopic 350-40.

3A.2.10 Software development methods**

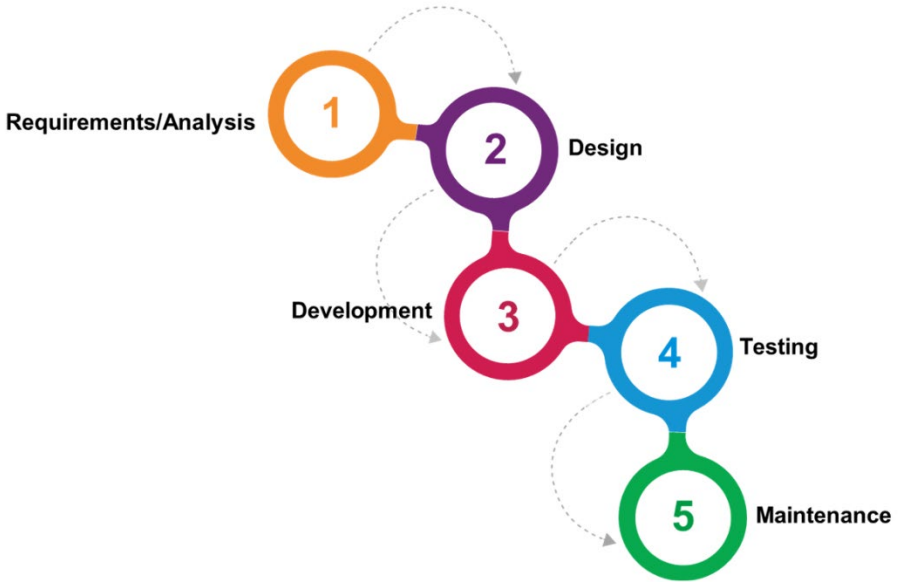
This section provides context to internal-use software development by comparing the agile software development method to the waterfall method. We observe that some entities apply hybrid models that incorporate elements of both models in their software projects, particularly in larger projects.

The discussion that follows is reproduced, without substantial change, from section 3.2.60; it has been relocated to this general discussion section because ASU 2025-06 aims to eliminate the need for distinct agile software development accounting considerations.

Waterfall method and its historical influence on Subtopic 350-40

The two words most used to describe the waterfall method of software development are the synonyms ‘sequential’ and ‘linear’.

The following is a representative illustration of the waterfall software development cycle; sometimes, in alternative representations, the phases are given different but generally synonymous titles (e.g. verification instead of testing, or implementation instead of development), or there may be more than five phases illustrated (e.g. by separating requirements and analysis).

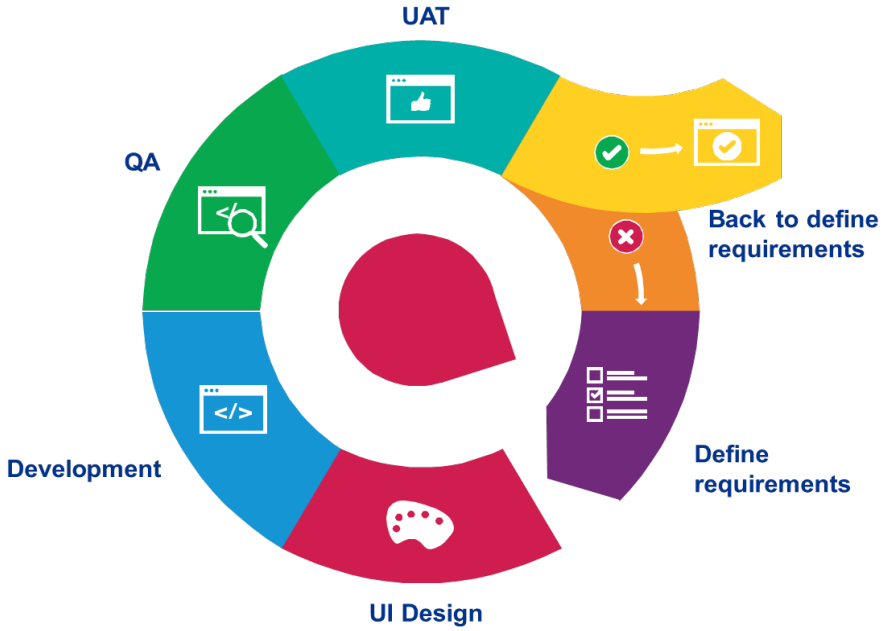


Under the waterfall method, in general, each phase of software development occurs after the one preceding it is completed. A hallmark of the waterfall method is its rigidity in terms of generally not being able to easily revisit a step or activity performed in an earlier phase. In addition, proceeding to the next step or phase typically requires successful completion of the current step or phase.

Prior to being amended by ASU 2025-06, Subtopic 350-40 (originally, SOP 98-1) was written in the late 1990s when the waterfall method was still predominant in software development. Consequently, practice widely accepted the notion that it was originally written in a waterfall method context.

Agile method and its relationship with Subtopic 350-40

The agile method differs substantially from the waterfall method and is designed to be flexible and iterative, rather than (1) heavily pre-planned or rigid and (2) sequential or linear. While there is an overall development objective, agile projects are typically much more lightly planned and completed through a series of shorter time-frame development 'sprints'. It is understood and accepted that later sprints will frequently drive re-work or revision of tasks completed in previous sprints to arrive at the completed project. One way the agile method can be illustrated follows.



This intended ability to make changes (flexibility) and revisit and/or reperform earlier activities (iteration) often gives rise to the sense that discrete (or distinct) software development phases (or stages) – i.e. that each begin only after the one preceding it ends – do not exist in agile projects. The frequently smaller scale and shorter timeframe of agile projects further contribute to this sense. Finally, the entity's release/update cycle can add to this sense if it involves the entity concurrently undertaking multiple projects, of different duration and levels of effort or change, related to the same software.

The agile method generally did not align with the legacy (pre-ASU 2025-06) Subtopic 350-40 model that was predicated on identifying discrete software development stages. Therefore, questions frequently arose about how to apply Subtopic 350-40 to agile software development (see section 3.2.60).

ASU 2025-06

One of the principal objectives of ASU 2025-06 was to make the guidance in Subtopic 350-40 more suitable (intuitive) for agile software development.

The principal amendments designed to achieve this objective were those that removed all references in Subtopic 350-40 to software project development stages (i.e. the project staging guidance).

Observation**

Removing the project staging guidance

Eliminating the legacy project staging guidance in Subtopic 350-40 seems likely to substantially alleviate the challenges that exist in applying the legacy guidance to agile (or similar non-linear, non-sequential) software development.

However, as discussed below, some of the judgments eliminated by removing the project staging guidance will likely be replaced with new judgments around the 'probable-to-complete' threshold. [ASU 2025-06.BC35]

Question 3A.2.10**

What special considerations often apply to agile software development projects?

Interpretive response: The following are two areas (not exhaustive) we observe entities have questions about when applying Subtopic 350-40 to agile software development projects.

Multiple units of account

Multiple agile projects related to the same software application may be ongoing concurrently and be at earlier or later stages of project completion. For example, while one project related to a single application (or module/component) may be in the early stages of determining performance requirements, another project may be past that stage and undertaking capitalizable development activities. In that case, the latter project's eligible costs are capitalized just as if it were the only ongoing project related to the application.

Minor upgrades and enhancements

We have observed many entities applying an agile development process produce frequent software releases that often comprise both maintenance updates and only minor upgrades and enhancements of software features. Those entities may conclude they cannot differentiate on a reasonably cost-effective basis the costs of the maintenance updates from the costs to develop the minor upgrades and enhancements with which they are bundled (see Question 3A.2.40). Therefore, they may expense all the development costs for a particular release of this nature. As noted in section 3A.2.60, Subtopic 350-40 does not define 'relatively minor' or 'reasonably cost-effective'; therefore, applying those notions involves judgment. [350-40-25-10]

3A.2.20 Step 1: Determine the unit of account

The unit of account for internal-use software and CCA implementation costs is not consistent within Subtopic 350-40. At various points, Subtopic 350-40 refers to the 'software project', to the individual cost and to the 'software module' or 'software component'. Question 3A.2.20 aims to resolve this inconsistency. [350-40-25-3 – 25-4, 25-12, 25-14, 35-6, 35-17, ASU 2025-06.BC61]

Question 3A.2.20

What is the unit of account for internal-use software and CCA development and implementation costs?

Interpretive response: It depends. In some aspects of initial recognition and measurement, the unit of account is the software 'project'. It is the project for which an entity determines whether costs can be capitalized (see section 3A.2.30). For an update to existing software, it is also generally at the project level that the entity determines whether (see section 3A.2.60):

— the update is an upgrade or enhancement; and
— if so, whether it is minor or more than minor (affecting whether the guidance in paragraph 350-40-25-10 applies to the project).

In addition, cost capitalization ceases when the project either is (see section 3A.2.30): [350-40-25-13 – 25-14]

— no longer probable of being completed and placed into service; or
— substantially complete and ready for its intended use.

Identifying the project may require judgment. We have observed this may especially be the case in some agile development scenarios. Question 3A.2.30 addresses considerations relative to identifying internal-use software 'projects'.

However, in certain other aspects of the Subtopic 350-40 accounting model, each individual software development or implementation cost – whether incurred in connection with a software development project or an internal-use software licensing arrangement – is evaluated to determine whether it should be capitalized or expensed as incurred. In other words, each cost incurred is its own unit of account for purposes of determining whether it is capitalized or must be expensed as incurred. [350-40-25, 350-40-30]

For example, even if one concludes the capitalization threshold has been met (see section 3A.2.30), some costs incurred thereafter will be capitalizable (e.g. coding and testing of the software), while others must be expensed as incurred (e.g. data conversion and training costs – section 3A.2.40 and section 3A.2.50 explain which costs can and cannot be capitalized.

CCA implementation costs

An entity treats the CCA as if it were an internal-use software project. This means that an entity looks at the implementation of the cloud-based solution (or module/component thereof) as the 'project' when determining when cost capitalization should begin and cease (see section 3A.2.30). [350-40-25-18]

Like internal-use software project costs, each individual CCA implementation cost is its own unit of account for purposes of determining whether it is capitalizable or must be expensed as incurred.

Question 3A.2.140 addresses accounting for costs of CCA implementation activities undertaken after go-live.

Multiple modules or components

If an internal-use software project or a cloud-based solution includes multiple modules or components, entities will need to identify the module or component to which development or implementation costs relate.

An entity that inappropriately treats a software application or a cloud-based solution as having a single module/component may not:

- appropriately identify and allocate implementation costs on a relative stand-alone price basis to modules or components that are substantially complete and ready for their intended use on different dates, or appropriately capitalize or expense development or implementation costs;
- begin amortizing capitalized costs at the right time or properly calculate the generally straight-line expense (see Question 6.2.30); or
- properly accelerate cost amortization when a software or cloud-based module or component is planned for abandonment (see section 6.2.30).

Question 3A.2.30

How does an entity determine what constitutes a 'project'?

Interpretive response: A 'project' can be large (e.g. creating an entirely new software application or module or adding significant new functionality to an existing application) or small (e.g. introducing a single new feature to the software such as a 'Log in with [LinkedIn/Facebook/Google]' option).

Defining the project requires judgment. However, in general, we believe a project is defined by its functional independence from other development efforts – e.g. whether its successful release or deployment depends on the success of other ongoing efforts. When two or more development efforts depend on each other to meet one or more design requirements/objectives, that would typically indicate those efforts are (or are part of, together with other development efforts) a single project, regardless of whether the entity's agile process breaks the development efforts down into smaller milestones (each potentially accomplished through separately defined sprints).

In contrast, we do *not* believe a project should be defined by its size in terms of either (1) development effort (e.g. timeline, labor hours or number of development sprints) or (2) significance (i.e. of the added functionality or overall extent of the changes to the software code).

Question 3A.2.40

Does the size and scope of a project affect whether or which costs get capitalized?

Interpretive response: No. This is because even a small, single-feature development project such as the 'log in with' example in Question 3A.2.30, which could potentially be developed in a single sprint, will generally include capitalizable costs (e.g. coding and testing costs incurred after the capitalization criteria in paragraph 350-40-25-12 are met) and non-capitalizable costs. Subtopic 350-40 requires capitalization of the eligible costs of even 'minor upgrades and enhancements' (see section 3A.2.60) unless those costs cannot be differentiated 'on a reasonably cost-effective basis' from software maintenance costs (see 'Minor upgrades and enhancements' discussion in Question 3A.2.10. [350-40-25-8, 25-10]

3A.2.30 Step 2: When to begin and cease capitalization**

Excerpt from ASC 350-40

25 Recognition

General

> Costs to Be Expensed as Incurred

25-1 Internal and external costs incurred prior to meeting the capitalization requirements in paragraphs 350-40-25-12 through 25-12A shall be expensed as they are incurred.

> Capitalization of Costs

25-12 Capitalization of costs shall begin when both of the following occur:

a. Paragraph superseded by Accounting Standards Update No. 2025-06.
b. Management, with the relevant authority, implicitly or explicitly authorizes and commits to funding a computer software project. Examples of authorization and commitment to funding a computer software project include the execution of a contract with a third party to develop the software, approval of expenditures related to internal development, or a commitment to obtain the software from a third party.
c. It is **probable** that the project will be completed and the software will be used to perform the function intended (referred to as the probable-to-complete recognition threshold). In evaluating whether the probable-to-complete recognition threshold has been met, an entity shall assess whether there is significant uncertainty associated with the development activities of the software (referred to as significant development uncertainty) in accordance with paragraph 350-40-25-12A.

25-12A If significant development uncertainty exists, the probable-to-complete recognition threshold in paragraph 350-40-25-12(c) is not met until that

significant development uncertainty has been resolved. Significant development uncertainty exists if either of the following factors is present:

a. The software being developed has technological innovations or novel, unique, or unproven functions or features, and the uncertainty related to those technological innovations, functions, or features, if identified, has not been resolved through coding and testing.
b. The significant **performance requirements** of the software have not been identified, or the identified significant performance requirements continue to be substantially revised.

For some types of software projects, the assessment of whether significant development uncertainty exists will be straightforward, such as illustrated in Example 1 (see paragraphs 350-40-55-5 through 55-8). For other types of software projects, the assessment will be more complex, such as illustrated in Example 3 (see paragraphs 350-40-55-13 through 55-17). If significant development uncertainty does not exist or if there was significant development uncertainty that has been resolved, an entity shall evaluate the requirements in paragraph 350-40-25-12 to determine when to begin capitalizing costs.

25-13 If the capitalization requirements in paragraphs 350-40-25-12 through 25-12A are no longer met for software being developed, no further costs shall be capitalized, and guidance in paragraphs 350-40-35-1 through 35-3 on impairment shall be applied to existing balances.

25-14 Capitalization shall cease no later than the point at which a computer software project is substantially complete and ready for its intended use, that is, after all substantial testing is completed.

Implementation Costs of a Hosting Arrangement That Is a Service Contract

25-18 An entity shall apply the General Subsection of this Section as though the **hosting arrangement** that is a service contract were an internal-use computer software project to determine when implementation costs of a hosting arrangement that is a service contract are and are not capitalized.

Cost capitalization begins

Once the unit of account is established, capitalization of eligible costs of a software project or CCA implementation only begins once the following two conditions are met. [350-40-25-12, 25-18]

Properly authorized entity management has: — authorized the project (explicitly or implicitly); and — committed requisite funding for the project	**AND**	It is probable that both: — the project will be completed; and — the software will be used for its intended purpose This is the 'probable-to-complete threshold'.

For all software projects, entities are required to assess whether significant uncertainty exists related to the development of the software ('significant development uncertainty'). The probable-to-complete threshold is not met until

any significant development uncertainty has been resolved. Significant development uncertainty exists if either: [350-40-25-12 – 25-12A]

- the software has novel, unique, or unproven functions and/or features or technological innovations that have not yet been resolved through coding and testing (see Question 3A.2.120); or
- the software's significant performance requirements have not been determined or are subject to substantial revision (see Observation below).

Questions 3A.2.50 to 3A.2.90 and Questions 3A.2.120 and 3A.2.130 address application questions around the probable-to-complete threshold and the concept of 'significant development uncertainty'. As time passes and more companies adopt and apply the amendments in ASU 2025-06, updates to these questions and new questions are likely to arise.

Cost capitalization ceases

Cost capitalization ceases at the **earlier of**: [350-40-25-13 – 25-14, 25-18]

Concluding it is no longer probable that the software will be completed and placed into service (or that the cloud-based solution will go live)	**AND**	When the software is substantially complete and ready for its intended use (or all substantial testing of the cloud-based solution necessary for it to go live is complete)

Observations

Observation**
'Significant' and 'substantial'

The FASB considered whether to use terms other than 'significant' and 'substantial', which appear in paragraphs 350-40-25-12 and 25-12A, and also considered whether it needed to provide additional guidance around those terms.

Ultimately, the FASB concluded that 'significant' is widely used in US GAAP such that how to apply the term in practice is well understood. [ASU 2025-06.BC53]

The FASB did not explicitly discuss 'substantial' either publicly (i.e. in deliberations of ASU 2025-06) or in the basis for conclusions to ASU 2025-06, but we believe the Board's considerations were likely similar to those for 'significant'.

Not defining either term appears to be consistent with the Board's statement in the ASU 2025-06 basis for conclusions that it expects entities to apply judgment when assessing significant development uncertainty. [ASU 2025-06.BC33]

Observation**

Only *significant* performance requirements and *substantial* potential revisions give rise to significant development uncertainty

The FASB emphasizes in the basis for conclusions to ASU 2025-06 that the amendments do not require an entity to identify and resolve all of the software's performance requirements before it begins to capitalize software development costs, only those performance requirements that are 'significant' and substantively unresolved. An entity should not defer eligible cost capitalization for either (1) minor performance requirements that have not yet been determined or (2) significant performance requirements subject only to further minor potential revision. [ASU 2025-06.BC50]

Questions and answers

Question 3A.2.50#

What does 'probable' mean?

Background: Before ASU 2025-06, Subtopic 350-40 did not explicitly link 'probable' in the probable-to-complete threshold to the ASC Master Glossary definition thereof. That, together with how SOP 98-1 (which is the source of most of the guidance in Subtopic 350-40) defined probable (but which was never codified in Subtopic 350-40), gave rise to diversity in practice about whether probable had the meaning described below as per the ASC Master Glossary or another meaning, such as that in SOP 98-1.

ASU 2025-06 resolves this legacy diversity.

Interpretive response: The amendments in ASU 2025-06 explicitly link the term 'probable' used in the probable-to-complete threshold to the ASC Master Glossary definition. 'Probable' is defined in the ASC Master Glossary as "The future event or events are likely to occur." [ASC Master Glossary]

Question 3A.2.60**

What key judgments are involved in evaluating whether significant development uncertainty exists?

Background: The FASB acknowledged that evaluating whether the probable-to-complete threshold has been met requires judgment, including when assessing whether significant development uncertainty exists. An entity's judgment should be based on its evaluation of its specific facts and circumstances. The Board noted that the application of judgment is inherent in US GAAP and concluded that the use of judgment is appropriate in the internal-use software

guidance because an entity's management is best positioned to evaluate its facts and circumstances, considering the diverse and continuously evolving nature of software development. [ASU 2025-06.BC31 – BC33]

The purpose of this question is to highlight key judgments we believe are inherent to assessing whether significant development uncertainty exists. Question 3A.2.80 discusses that a software project may not meet the probable-to-complete threshold even if no significant development uncertainty exists.

Interpretive response: We believe entities may make different judgments, even in similar circumstances, about whether software features or functions are novel, unproven or unique; what constitutes a 'significant' performance requirement; or what level of ongoing revision of a significant performance requirement is 'substantial'. They also may interpret the level of coding and testing necessary to resolve significant development uncertainty stemming from novel, unproven or unique features and functions differently. The probable-to-complete threshold guidance in Subtopic 350-40 is not designed to eliminate, or enforce conforming (i.e. from entity to entity), judgment.

The table that follows indicates key judgments we believe are inherent to assessing when significant development uncertainty exists, and things entities may consider when making those judgments. *Neither the list of judgments, nor the considerations we enumerate, are exhaustive.*

Judgment	Considerations and suggestions for overcoming uncertainty
What constitutes a novel, unique, unproven function or feature or technological innovation?	Question 3A.2.120 addresses resolving uncertainty stemming from novel, unique or unproven features or functions or technological innovations, while Question 3A.2.70 addresses the perspective an entity takes in assessing whether a function or feature is novel, unique or unproven. Judgment also exists in assessing whether features or functions are novel, unique or unproven, or are 'technological innovations'. In making these judgments, we believe entities would likely do the following (not exhaustive). — **Consult with their technical experts.** Engage with their software engineers and developers to understand the nature of the features and functions being developed. — **Consider their experience with previous software projects.** To the extent there *are* similar projects, the extent of that similarity may indicate whether the features and functions in question are novel, unique or unproven or are technological innovations. In addition, past projects may give insight into how accurate the entity's judgments were in this regard; for example, an entity's past failures to successfully develop features or functions it concluded were *not* novel, unique or unproven might suggest its assessments in those cases were not accurate.

Judgment	Considerations and suggestions for overcoming uncertainty
	— **Consider the nature of the project as an upgrade/enhancement versus new software.** Upgrades or enhancements to existing software may be less likely to be unique, novel or unproven than new software development (e.g. an upgrade or enhancement may leverage existing, operable software code).
What are *significant* performance requirements?	'Performance requirements are what the entity needs the software to do (for example, functions or features)'. We generally do not believe there is significant judgment involved in identifying performance requirements; we would generally expect that an entity can identify, at some point during a project, what it intends for the software to do. [350-40 Glossary] Therefore, we believe the principal judgment involved is identifying which performance requirements are 'significant' (see our observation 'Significant' and 'substantial' earlier in this section). How an entity might go about obtaining sufficient and appropriate information to make its judgment in this regard may differ depending on the type of internal-use software in question. — **For software that will be sold to customers (i.e. on a SaaS basis),** the entity may (not exhaustive): – obtain information from the project developers and the business/sales personnel that will be tasked with selling the SaaS offering to customers; and – review draft marketing materials and/or proposals upon which management approved and committed to funding the project. — **For software for the entity's internal use,** the entity may (not exhaustive): – obtain information from the project developers and the internal project sponsor; and – if licensing the software from a vendor, look to the software vendor's marketing materials, the entity's own RFP and the vendor's response thereto, and the license agreement (and any related contracts like any professional services statements of work).
When are significant performance requirements still subject to further *substantial* revision?	Judgment is likely involved in both (1) assessing when significant performance requirements are subject to further revision and (2) when such revisions may be 'substantial', as opposed to only minor (and perhaps inherent to an agile development process). Here again, we believe it is important for entities to work together with, and obtain information from, those involved in developing and using/selling the software, when making these judgments.

Judgment	Considerations and suggestions for overcoming uncertainty
	And consistent with the first judgment area in this table, we believe entities should consider their history with respect to these judgments. For example, does the entity have a history of frequently assessing significant performance requirements as settled only for them to still change substantially, or vice versa? Alternatively, does the entity have a demonstrated history of accurate conclusions in this regard?

Question 3A.2.70**

Is 'novel, unique or unproven' assessed based on internal factors only or assessed more broadly?

Background: We believe an entity *could* reach a different conclusion about whether a software feature or function is novel, unique or unproven (or constitutes a technological innovation) if it assesses that question against its own knowledge and experience instead of against a broader consideration of software successfully developed by other entities.

Interpretive response: Subtopic 350-40 (inclusive of the basis for conclusions to ASU 2025-06) does not address this question. Therefore, reasonable judgment may be acceptable.

That said, we believe 'significant development uncertainty' fundamentally exists to identify when it is not probable that *an entity*, in the context of *its* software project and development, will complete *that* project. Therefore, even if a substantially similar feature or function is known to have been successfully developed by another entity, but the entity in question does not have access to that *other* entity's knowledge or expertise (or other substantially equivalent knowledge or expertise), we believe it may be reasonable to conclude that the feature or function is novel, unique or unproven in the context of *the entity's* software development project and, therefore, gives rise to significant development uncertainty. By contrast, we do not believe the mere fact that the entity has not developed a similar feature or function in the past means the relevant feature or function is novel, unique or unproven.

Question 3A.2.80**

Are there reasons *other than* 'significant development uncertainty' that the probable-to-complete threshold may not be met?

Interpretive response: Yes. Even if no significant development uncertainty exists, a software project may not meet the probable-to-complete threshold. The FASB intentionally did not create a finite list of factors or indicators related

to meeting the overall threshold, observing this threshold already existed under legacy Subtopic 350-40 (pre-ASU 2025-06 paragraph 350-40-25-12(b)). The FASB expects that entities will continue to consider all relevant facts and circumstances when assessing the probable-to-complete threshold, and not just the existence (or non-existence) of significant development uncertainty. [ASU 2025-06.BC45]

Question 3A.2.90**

Does remaining *insignificant* development uncertainty preclude meeting the probable-to-complete threshold?

Background: To give context to this question, assume an entity initially concludes a novel and unproven feature of its planned software creates significant development uncertainty. At some point, it has substantially demonstrated that feature's viability (see Question 3A.2.120), but may conclude that *some* uncertainty exists about, for example, the ability to integrate this feature with another one in the software. The entity considers this to be only a minor risk as it has successfully integrated new features into its software products in the past.

Interpretive response: No. We believe the reference to *significant* development uncertainty means *insignificant* development uncertainty (including *remaining* insignificant development uncertainty) would not, in isolation, preclude the probable-to-complete threshold from being met. Judgment will necessarily be involved in determining whether remaining development uncertainty is only insignificant.

We believe two statements in ASU 2025-06's basis for conclusions support this view. These are that (1) an entity is not required to "identify and cease revising all of the software's performance requirements before it begins to capitalize its software" and (2) "an entity should identify only those performance requirements that are significant and/or are significant and expected to continue to be substantially revised." Taken together, these statements mean that remaining unidentified *insignificant* performance requirements or the remaining possibility of *nonsubstantial* revisions to significant performance requirements do not preclude meeting the probable-to-complete threshold. While these statements only explicitly relate to the significant performance requirements factor, we do not believe the concept suggested by these statements is isolated to this factor alone. [ASU 2025-06.BC50]

Question 3A.2.100

Does technological feasibility need to be established before software development costs are capitalized?

Background: Subtopic 985-20 requires all costs incurred to establish the technological feasibility of computer software to be sold, leased or otherwise marketed be charged to expense when incurred. Subsequent eligible costs are capitalized only after technological feasibility has been established. [985-20-25-1]

Technological feasibility is established when the entity has completed all planning, designing, coding and testing activities necessary to establish that the product can be produced to meet its design specifications, including functions, features and technical performance requirements. [985-20-25-2]

See chapter 5 for guidance on applying Subtopic 985-20.

Interpretive response: No. There is no technological feasibility threshold in Subtopic 350-40. While considered during the development of SOP 98-1 and again during the deliberations of ASU 2025-06, it was ultimately concluded such a threshold was not appropriate for internal-use software. [SOP 98-1.51, ASU 2025-06.BC42]

Question 3A.2.110

How do iterative development activities affect software cost capitalization?

This question previously existed as part of Question 3.2.170. While separately presented in this chapter, our interpretation has not changed.

Background: A question arises, principally in the context of agile software development, about how to account for iterative development activities that in effect change or override earlier, similar project activities. For example, if, after going through quality assurance (QA) or user acceptance testing (UAT), the development team decides to change some of the project design features, does the entity capitalize the eligible development costs (see section 3A.2.50) to effect the design changes *and* continue to recognize the previously capitalized application development costs?

Interpretive response: We believe this depends on the facts and circumstances. However, in general, all software development has aspects of trial and error to it. Therefore, just because there are design changes or coding that must be revised or added for development flaws or design changes to effect the same functionality does not mean the entity must expense the previously incurred costs (e.g. those incurred before the QA or UAT that led to the coding or design change). Instead, we would more typically expect all the eligible application development costs (see sections 3A.2.40 and 3A.2.50) to be capitalized as costs necessary to complete the project.

Undertaking iterative activities, typically most inherent to agile development, within a project differs from a scenario of expecting a subsequent project to override or replace the feature(s) being developed in the current project. In the latter scenario, the expectation of override or replacement should affect the useful life assigned to the current project software asset. [350-40-25-15, 35-5]

Question 3A.2.120**

What does it mean to resolve uncertainties related to novel, unique or unproven functions or features through coding and testing?

Background: Subtopic 350-40 stipulates that when assessing significant development uncertainty, the uncertainty related to the development of novel, unique or unproven software features or functions or technological innovations must be resolved through coding and testing. It does not explain what constitutes coding and testing or what coding and testing must demonstrate to resolve the uncertainty. [350-40-25-12A]

Interpretive response: In general, we believe 'to resolve' significant development uncertainty stemming from novel, unique or unproven software features or functions or technological innovations via coding and testing means the entity proves (or establishes) – via that coding and testing – the viability of those features, functions or technological innovations to the point that they no longer give rise to significant development uncertainty for the overall software project. Judgment will necessarily be involved in determining when this requirement has been satisfied.

As a practical matter, we believe entities would generally achieve an adequate level of resolution if they have produced a 'working model' (drawing on the definition thereof and the guidance in paragraph 985-20-55-9 by analogy – see section 5.2.10). This may be a desirable approach for entities that establish technological feasibility of external-use software through a working model approach because of the consistency it may afford them in their development cost accounting approaches for their internal- and external-use software.

However, we do not believe an entity *must* produce a working model to resolve significant development uncertainty stemming from novel, unique or unproven software features or functions or technological innovations; there may be other ways to meet the requirement given that Subtopic 350-40 does not specify. Additionally, we do not believe the standard permits an entity to *default* to a working model 'threshold' (or any other threshold) if doing so would inappropriately delay resolution.

Lastly, we observe that entities should consider that a novel, unique or unproven function or feature may be only one part of the 'software project' (see section 3A.2.20) such that sufficient viability of that function or feature may be established through appropriate coding and testing activities before a working model of the *complete* software is produced. In that case, the software project may be probable of completion once the uncertainty related to the novel, unique or unproven function or feature is resolved even though the

development of other functions or features that are *not* novel, unique or unproven is not yet complete or has not yet advanced to the point of a working model.

Comparison to Subtopic 985-20

Under Subtopic 985-20, to establish technological feasibility before producing a working model (i.e. under paragraph 985-20-25-2(a)), an entity must conclude that all 'high-risk development issues' have been resolved through coding and testing. Examples of high-risk development issues are 'novel, unique or unproven functions and features or technological innovations' – i.e. consistent with the factor in paragraph 350-40-25-12A(a). [985-20-25-2(a)(3)]

In practice, most entities establish technological feasibility through a working model; a working model of the software product implicitly demonstrates that any high-risk development issues have been resolved. Therefore, most entities do not assess high-risk development issues separately from assessing the technological feasibility of the software product as a whole.

For those entities that have experience assessing the resolution of high-risk development issues under Subtopic 985-20, the FASB has indicated it expects the process to resolve such issues would be similar under Subtopics 350-40 and 985-20. That said, we believe a difference could arise given that the basis for conclusions to FAS 86 (which was codified as Subtopic 985-20) states that "Resolution of *all* uncertainties related to identified high-risk development issues is therefore included as a requirement for establishing technological feasibility." [emphasis added] [ASU 2025-06.BC55, FAS 86.34]

We contrast that with only *significant* development uncertainty definitively precluding meeting the Subtopic 350-40 probable-to-complete threshold (see Question 3A.2.90). Therefore, while it may be 'similar', the analysis *may* not be exactly the same. An entity may be able to conclude it has resolved significant high-risk development issue uncertainty earlier under Subtopic 350-40 than under Subtopic 985-20.

Question 3A.2.130**

What is the accounting for significant development uncertainty that arises after the probable-to-complete threshold has been met?

Background: Significant development uncertainty may arise after it was initially determined that the probable-to-complete threshold was met. For example, an entity may have reasonably concluded (i.e. based on appropriate judgment) that a software project's significant performance requirements were not subject to further substantial revision. However, as it worked to complete the project, external factors (e.g. competitor actions, new regulations the software is intended to address) result in the entity concluding it needs to substantially revise or add a new significant performance requirement.

Interpretive response: The probable-to-complete threshold is not met until significant development uncertainty has been resolved (or mitigated to an insignificant level – see Question 3A.2.90). [350-40-25-12A]

Identifying novel, unique, unproven functions and features or technological innovations, or substantial revisions to the significant performance requirements of the software (i.e. factors that indicate significant development uncertainty), after the probable-to-complete threshold is met should be infrequent.

However, if a new issue arises and is not the result of an error in the previous assessment of the probable-to-complete threshold, we believe the newly identified significant development uncertainty means the software project is no longer probable of completion. Therefore, the guidance in paragraph 350-40-35-3 applies. Thereunder, when it is no longer probable that *uncompleted* internal-use software will be completed, the in-process internal-use software asset is written down to the lower of its carrying amount and fair value less costs to sell. There is a rebuttable presumption that uncompleted internal-use software has a fair value of zero. [350-40-35-3]

Consequently, the entity:

— expenses all previously capitalized costs in the period in which the new issue was identified (except to the extent the presumption of zero fair value can be rebutted); and

— expenses new development costs as incurred until the probable-to-complete threshold is again met.

KPMG Handbook, Accounting changes and error corrections, provides additional guidance on identifying and accounting for errors.

Example 3A.2.10**

Determining when to begin cost capitalization – new AI data management and analytics software

Background

ABC Corp. is a SaaS vendor that specializes in cloud-based data software solutions. As of January 1, Year 2, ABC is developing new AI-powered data management platform software designed for enterprise customers with complex, distributed data environments. ABC's new software will be delivered exclusively via the cloud; therefore, it is internal-use software in the scope of Subtopic 350-40 (see section 2.4).

ABC's intended customers for the new software typically operate across multiple business units and geographies, each with their own data sources and analytics tools, resulting in fragmented data landscapes and challenges in achieving unified insights. The software will provide centralized data ingestion, cleansing, storage and governance, enabling customers to aggregate and standardize data from disparate systems. In addition, it will include advanced AI-powered analytic capabilities. These capabilities are expected to include:

— continuously scanning customer data for anomalies, outliers and shifts in key business drivers;
— alerting users when significant changes in assumptions or trends are detected, supporting financial reporting, risk management and operational forecasting;
— generating automated insights and recommendations based on detected patterns;
— integrating with external data sources (e.g. market feeds, regulatory updates) to enrich analysis and support scenario modeling; and
— providing interactive dashboards for users to visualize data lineage, model results and exception alerts.

When to begin capitalization – analysis

ABC assesses when capitalization should begin for the new software based on the steps and analysis that follow.

Step 1: What is (are) the 'software project(s)'?

ABC first concludes that the new software constitutes a single software project because all of the capabilities described above are necessary to successfully penetrate the data management software market. Customers expect advanced analytic capabilities from any new data management solution, especially considering the expected significant implementation costs (i.e. customers will not incur those costs for software that does not have these capabilities). Therefore, ABC concludes that only the capabilities together constitute a commercially viable offering, and thus, the project.

Step 2: Does 'significant development uncertainty' exist?

Under paragraph 350-40-25-12, capitalization of the new software's development costs cannot begin if significant development uncertainty exists related to the software's development. To determine whether such uncertainty exists, ABC assesses whether either of the factors in paragraph 350-40-25-12A are present. As of January 1, Year 2:

— ABC concludes that the AI-powered data analytics features are novel and unproven, and ABC has not progressed the coding and testing of these new features to the point necessary to prove their viability. In this regard, ABC considers that it has never developed AI-powered features of this nature before and it does not have access to any other software code (e.g. open-source code) or software developers that have successfully developed substantially similar features. The intended features are also not widely available in the marketplace such that ABC's efforts are inherently not novel, unique or unproven.

— ABC determines that because it has not yet established the viability of the new analytics features, it cannot conclude the significant performance requirements of the new software will not be substantially revised (e.g. based on successful development efforts of the new features).

Therefore, ABC concludes that significant development uncertainty exists.

On October 1, Year 2, ABC concludes that significant development uncertainty no longer exists. This is because of the following circumstances.

— ABC has successfully coded and tested the new AI-powered data analytics features to a point where they successfully integrate and interoperate with the non-AI features. ABC's development process in this case was such that coding and testing of the AI data analytics features only were completed to the point of proving the viability of those novel features concurrent with the coding and testing of the new software product as a whole, at which point ABC determined it had completed a 'working model' of the complete software.

— The successful completion of the working model means it is now unlikely that any of the significant performance requirements will substantially change.

Therefore, in accordance with paragraph 350-40-25-12, capitalization of eligible costs begins on October 1, Year 2 if the criteria in 350-40-25-12(b) and (c) have also been met.

Step 3: Has ABC management approved the project and committed funding (criterion (b))?

Criterion (b) in paragraph 350-40-25-12 requires that management with relevant authority has authorized and committed to funding the project. In this instance, ABC management, including the CEO and the CIO (in charge of R&D), approved the project and expected sufficient funding in late Year 1, before development began in earnest.

Step 4: Is it probable the new data management software will be completed and used for the function intended (criterion (c))?

Criterion (c) in paragraph 350-40-25-12 requires it to be probable that the software project will be completed and used for the function intended, and considers facts and circumstances beyond solely the presence of significant development uncertainty. In this instance, as of October 1, Year 2, all significant development uncertainty is resolved (see Step 2). Further, ABC concludes that completion of the working model of the software is evidence that it is probable the software project will now be completed and the new software sold to customers via a cloud-based model as was always intended. No other facts and circumstances exist suggesting otherwise.

Conclusion

ABC will begin capitalizing any remaining, eligible development costs on October 1, Year 2. Before that date, all costs of the new software development are expensed as incurred.

Example 3A.2.20**

Determining when to begin cost capitalization – new analytics features for existing AI data management software

Background

Assume the same facts in Example 3A.2.10, except that the non-AI capabilities of the software already exist. Only relatively minor updates and enhancements are being made to the existing software to keep it current with customer expectations. The new AI-powered analytics features will be available to customers as a separate, add-on module to the core data management software.

While ABC expects many new and existing customers to purchase access to the new analytics module and to integrate it with the existing data management software, it expects other customers to continue to subscribe to just the existing data management software on its own.

When to begin capitalization – analysis

ABC assesses when capitalization should begin for the identified software project(s) based on the steps and analysis that follow.

Step 1: What is (are) the 'software project(s)'?

Because the core data management software can operate independently of the new AI-powered analytics features and ABC expects many customers will continue to do so, ABC concludes that the updates and enhancements to the core data management software and the development of the new analytics features are two separate and distinct software projects.

Step 2: Does 'significant development uncertainty' exist?

AI data analytics features project: The relevant facts and ABC's analysis for the new AI data analytics project are substantially the same as outlined in Step 2 in Example 3A.2.10. Therefore, ABC concludes that significant development uncertainty exists related to that project until October 1, Year 2.

In this example, on October 1, Year 2:

— ABC concludes that the significant development uncertainty stemming from the novel and unproven nature of the AI data analytics features has been resolved via testing that its analytic features can work as intended with a test data set. Even though ABC has not yet tested the features with a data set derived from integration with the core data management software, ABC concludes that the analytics features are viable, and therefore no longer novel and unproven.

— As in Example 3A.2.10, successful testing of the analytics features also means it is now likely that there will be no substantial revisions to the analytics module's significant performance requirements.

Therefore, in accordance with paragraph 350-40-25-12, capitalization of eligible costs for this project begins on October 1, Year 2 if the criteria in 350-40-25-12 (b) and (c) have also been met.

Core data management software upgrades project: Significant development uncertainty does not exist for this project at any point because:

- the planned updates and enhancements are not significant, and ABC has sufficient, relevant experience in developing similar updates and enhancements in the past to conclude that these upgrades are not novel, unique or unproven or technological innovations; and
- the significant performance requirements of the updates and enhancements are known and unlikely to substantively change.

Step 3: Has ABC management approved the project and committed funding (criterion (b))?

Criterion (b) in paragraph 350-40-25-12 requires that management with the relevant authority has authorized and committed to funding the project. In this instance, ABC management, including the CEO and the CIO (in charge of R&D), approved both projects and sufficient expected funding on December 10, Year 1, before development work began on January 1, Year 2.

Step 4: Is it probable the new data management software will be completed and used for the function intended (criterion (c))?

Criterion (c) in paragraph 350-40-25-12 requires it to be probable that the software project will be completed and used for the function intended.

AI data analytics features project: ABC concludes that the probable-to-complete threshold is met for this project as of October 1, Year 2. No significant development uncertainty remains, and even though ABC has not yet tested these features on an integrated basis with the core data management software (see Step 2), it concludes that that integration risk is minor such that the risk does not call into question this project being probable of completion.

Core data management software project: Given Management's commitment to the project and its funding (see Step 3), and the non-novel nature of the planned updates and enhancements, for which ABC has relevant experience in developing, ABC concludes it was probable they would be completed and used for their intended purpose (i.e. to update ABC's core data management software) as soon as development efforts began on January 1, Year 2.

Conclusion

ABC will begin capitalizing any remaining, eligible development costs of the AI data analytics features project on October 1, Year 2. Before that date, all costs of the new module's development are expensed as incurred.

By contrast, ABC will capitalize eligible development costs of the enhancements to the core development management software beginning on January 1, Year 2 if it can, on a reasonable cost-effective basis, separate its development costs for the relatively minor enhancements from those of the other, maintenance updates that will also be included in the release (see section 3A.2.60).

Question 3A.2.140

Are costs of CCA implementation activities incurred after go-live capitalizable?

Background: Implementation activities are not necessarily undertaken only at or before go-live. Consider the following examples (not exhaustive).

- Customer implements cloud-based solution T in Year 1. Customer's implementation activities include significant configuration activities before go-live. In Year 2, Customer decides it wants to change the configuration of T, effectively overwriting the configuration effected in Year 1.
- Customer implements cloud-based solution Z in Year 1. Customer's implementation activities include configuration of Z, but not to a significant degree. In Year 2, Customer decides it can incrementally benefit from Z if it more significantly and specifically configures Z.
- Customer implements cloud-based solution X in Year 1. Customer's implementation activities include configuring and interfacing X to work together with its on-premise HR application. In Year 3, Customer sunsets the on-premise HR application for cloud-based HR solution Y. As part of implementing Y, Customer reconfigures X and implements a new interface between X and Y.
- The same facts as the previous example except that Y is a new on-premise HR application.

In these (and similar) examples, the question arises about whether Customer should capitalize the costs to reconfigure T, Z and X and implement a new interface between X and Y under Subtopic 350-40.

Interpretive response: It depends. In general, consistent with the Subtopic 350-40 guidance applicable to internal-use software costs, we believe CCA implementation costs should not be capitalized after go-live. [350-40-25-14]

However, we believe exceptions arise if the related implementation activities:

- relate to a new CCA or new internal-use software application – e.g. Y in the last two background examples;
- create a new asset for which the incurred costs should be capitalized under Subtopic 350-40 or other Topics – e.g. a new interface that meets the definition of internal-use software; or
- increase the functionality of the cloud-based solution.

Activities relate to a new CCA or internal-use software application

If the activities relate to a new CCA that is not yet live or an internal-use software application that is not yet 'substantially complete and ready for its intended use', the related costs that qualify are capitalized.

Judgment may be required to determine whether new activities relate to (1) an existing CCA or (2) a new CCA or on-premise application. However, just because the activities are being undertaken because of implementing a new

CCA or internal-use software application does not mean the new activities relate to that new CCA or application.

In general, we would not view changes specifically to the cloud-based software subject to the existing CCA (e.g. changes to its configuration) as related to another CCA or application – even if those changes are only being made because the entity is implementing the new CCA or application.

By contrast, it may require judgment to determine to which CCA or application a new interface (that is not itself an internal-use software asset – see 'Create a new asset' below), implemented to connect the existing CCA with the new CCA or application, relates. It may be relevant in those circumstances to consider where the interface will reside; if it will reside in the entity's IT environment (or its own hosting environment – e.g. its own AWS or Azure instance), it may be a separate internal-use software asset.

— If the new interface will reside in the hosting environment of the existing CCA cloud service provider, this may indicate it relates to the existing CCA.

— Alternatively, if the new interface will reside in the hosting environment of the cloud service provider for the new CCA, this may indicate it relates to the new CCA.

Create a new asset

Some implementation activities create independent internal-use software or PP&E assets. For example, an interface may itself meet the definition of internal-use software, or implementation activities may include the acquisition or construction of assets that meet the definition of PP&E. In those cases, the entity should follow the Topic applicable to the type of asset created (e.g. Subtopic 350-40 or Topic 360).

Additional functionality or utility

Costs of upgrades and enhancements to internal-use software – i.e. costs that result in additional functionality in the software – are generally capitalized if those same costs would be capitalized for new software. [350-40-25-7 – 25-11]

While the 'Implementation Costs of a Hosting Arrangement That Is a Service Contract' subsections of Subtopic 350-40 do not contain equivalent guidance, Subtopic 350-40 explicitly directs entities to refer to the 'General' subsection of Section 350-40-25 to determine when CCA implementation costs should be capitalized.

We believe implementation costs incurred to substantively increase the functionality or utility of the cloud-based solution – i.e. implementation costs incurred (e.g. configuration changes) to enable the entity to undertake additional tasks or perform additional functions using the hosted software – are analogous to the costs of upgrades and enhancements to internal-use software, and therefore should be capitalized. [350-40-25-18]

In contrast, costs incurred that do *not* result in the entity being able to undertake additional tasks or perform additional functions using the hosted software – e.g. costs incurred that merely change how an existing task or function is performed – should not be capitalized.

Consideration of previous activities

Regardless of whether the costs of new implementation-type activities should be capitalized, we believe consideration should be given to whether those new activities indicate:

— a plan to abandon a module or component of a cloud-based solution (see section 6.2.30);
— a need to reassess the term of the hosting arrangement (see Question 6.2.100); and/or
— the asset group that includes the capitalized implementation costs is impaired (see section 6.2.40).

Question 3A.2.150

Do CCA implementation costs incurred by the acquiree give rise to an asset in a business combination or asset acquisition?

Background: Some implementation activities in CCAs give rise to assets that were recognized before the issuance of ASU 2018-15 – e.g. interfaces developed for use in the customer's IT environment generally meet the definition of internal-use software. Such internal-use software assets will be recognized at fair value in acquisition accounting.

However, the question arises about whether costs previously incurred by an acquiree to implement a CCA give rise to an asset that should be recognized in acquisition accounting. Further, some question whether implementation costs that are required to be expensed as incurred under Subtopic 350-40 (e.g. data migration/conversion and training costs) can still give rise to assets in acquisition accounting.

Interpretive response: In general, yes to both.

Business combinations

The acquiree's pre-acquisition implementation activities will typically permit a market participant to avoid incurring both:

— similar implementation costs to derive value from the CCA; and

— hosting service fees, stemming from the acquiree's contract, during the period it would take to implement the cloud-based solution had the acquiree not already undertaken its implementation activities.

In this way, we believe the acquirer obtains a contractually based in-place CCA asset that is like an in-place lease asset, which reflects the value an acquirer-lessor obtains from an in-place lease at the acquisition date. An in-place lease generally permits the acquirer-lessor to avoid new lease origination and underlying asset holding costs; see Question 11.1.110 in KPMG Handbook, Leases.

The fair value of an in-place CCA, which will depend on the facts and circumstances, should be recognized in the business combination on a CCA-by-CCA basis.

Consistent with our view about in-place lease assets, we believe in-place CCA assets recognized under Topic 805 generally should be reported separately (as an intangible asset) from:

— acquired technology assets (e.g. acquired internal-use software); and
— any favorable contract asset or unfavorable contract liability associated with the CCA arising from off-market hosting service fees.

Asset acquisitions

We believe the above related to business combinations applies equally to asset acquisitions. However, because an entity does not recognize goodwill or a bargain purchase gain in an asset acquisition, the amounts recognized for the implementation costs intangible asset may differ from what would have been recognized under Topic 805 had the transaction been a business combination.

Question 3A.2.160

When is a cloud-based solution ready for its intended use?

Background: Subtopic 350-40 specifies that internal-use software is ready for its intended use when all substantial testing has been completed. [350-40-25-14]

Interpretive response: Subtopic 350-40 specifies that customers in CCAs should capitalize implementation costs based on the guidance for such costs incurred when developing or implementing internal-use software. Therefore, consistent with developed or licensed internal-use software, a cloud-based solution (or module/component thereof) is ready for its intended use when all substantial testing of the solution is complete. [350-40-25-18]

3A.2.40 Step 3: Does a specific requirement apply to the activity?

Excerpt from ASC 350-40

25 Recognition

General

> Costs to Be Expensed as Incurred

25-4 Internal and external training costs are not internal-use software development costs and shall be expensed as incurred.

25-5 Data conversion costs, except as noted in paragraph 350-40-30-1(d),shall be expensed as incurred. The process of data conversion from old to new systems may include purging or cleansing of existing data, reconciliation or balancing of the old data and the data in the new system, creation of new or additional data, and conversion of old data to the new system.

Implementation Costs of a Hosting Arrangement That Is a Service Contract

25-18 An entity shall apply the General Subsection of this Section as though the **hosting arrangement** that is a service contract were an internal-use computer software project to determine when implementation costs of a hosting arrangement that is a service contract are and are not capitalized.

Subtopic 350-40 specifically prohibits costs of the following activities from being capitalized: [350-40-25-4 – 25-5, 25-18]

— personnel training (see Observation below); and
— data conversion (or migration – see Question 3A.2.170).

The requirement to expense costs of data conversion or migration as incurred does not extend to costs to develop or acquire (including obtaining a license to) software to assist in the activity. [350-40-25-3]

Observation#

Expensing training and data conversion costs

AcSEC decided that training costs should be expensed as incurred because: [SOP 98-1.71]

— they are not software development costs; and
— entities are not able to identify the specific future period benefited because they do not control the continued employment of the trained employees.

The discussion in SOP 98-1 outlined in the second bullet above makes clear that the explicit expensing requirement related to 'training' refers to *personnel* training. [SOP 98-1.71]

AcSEC decided that data conversion costs, other than those permitted by paragraph 350-40-25-3, should be expensed as incurred because converting existing data is inherent to the continuing business, rather than creating a new asset for the entity. [SOP 98-1.70]

Question 3A.2.170

Should data 'migration' costs be accounted for in the same manner as data 'conversion' costs?

Interpretive response: Yes.

Data conversion and data migration are not, strictly speaking, synonyms. The former refers to the process of converting computer data from one format to another, while the latter can refer to the process of transferring data from one system or technology to another without necessarily requiring a format change.

However, the terms are often used interchangeably, including by the FASB staff in its written issue summaries during the development of ASU 2018-15. [EITF Issue Summary No. 1, Supplement No. 1, September 28, 2017]

In addition, we believe AcSEC's basis for deciding that data conversion costs should be expensed as incurred (see Observation 'Expensing training and data conversion costs') is equally relevant for data migration costs.

Therefore, although Subtopic 350-40 only specifically refers to expensing 'data conversion' costs as incurred, we believe the same requirement applies to data migration costs.

Question 3A.2.180

Are hosting service fees paid to the cloud service provider in a CCA before completion of implementation activities an implementation cost?

Background: The cloud service provider may initiate the hosting service (i.e. complete the user interface and activate the service) before the customer (or another party on its behalf) completes implementation activities that are necessary for the customer's intended use of the cloud-based solution. Examples of such implementation activities are specific configurations and desired interfaces with other on-premise or cloud-based applications.

In these scenarios, the question arises about whether the customer should capitalize the hosting service fees incurred before the cloud-based solution is ready for its intended use as a CCA implementation cost.

Interpretive response: No. The hosting service fees are outside the scope of the 'Implementation Costs of a Hosting Arrangement That Is a Service Contract' subsections of Subtopic 350-40, and therefore are not an implementation cost. The CCA itself is a service contract and the hosting service fees are expensed as the hosting services are provided, consistent with other service contracts (unless the fees can be capitalized as part of the cost of another asset). [350-40-15-4C, ASU 2018-15.BC7]

Question 3A.4.10 addresses the commencement of expense recognition for hosting service fees.

3A.2.50 Step 4: What activity costs qualify for capitalization?

Excerpt from ASC 350-40

30 Initial Measurement

General

> Capitalizable Costs

30-1 Costs of computer software developed or obtained for internal use that shall be capitalized include only the following:

a. External direct costs of materials and services consumed in developing or obtaining internal-use computer software. Examples of those costs include but are not limited to the following:
    1. Fees paid to third parties for services provided to develop the software
    2. Costs incurred to obtain computer software from third parties
    3. Travel expenses incurred by employees in their duties directly associated with developing software.
b. Payroll and payroll-related costs (for example, costs of employee benefits) for employees who are directly associated with and who devote time to the internal-use computer software project, to the extent of the time spent directly on the project. Examples of employee activities include but are not limited to design of chosen path, including software configuration and software interfaces, coding, installation to hardware, and testing, including parallel processing phase.
c. Interest costs incurred while developing internal-use computer software. Interest shall be capitalized in accordance with the provisions of Subtopic 835-20.
d. Costs to develop or obtain software that allows for access to or conversion of old data by new systems.

30-2 If the entity suspends substantially all activities related to the software developed or obtained for internal use, interest capitalization shall cease until activities are resumed.

30-3 General and administrative costs and overhead costs shall not be capitalized as costs of internal-use software.

> Multiple-Element Arrangements Included in Purchase Price

30-4 Entities may purchase internal-use computer software from a third party or may enter into a **hosting arrangement**. In some cases, the price includes multiple elements, such as the license or hosting, training for the software, maintenance fees for routine maintenance work to be performed by the third party, data conversion costs, reengineering costs, and rights to future upgrades and enhancements. Entities shall allocate the cost among all individual elements. The allocation shall be based on the relative **standalone price** of the elements in the contract, not necessarily separate prices stated within the contract for each element. Those elements included in the scope of this

Subtopic shall be accounted for in accordance with the provisions of this Subtopic.

Implementation Costs of a Hosting Arrangement That Is a Service Contract

30-5 An entity shall apply the General Subsection of this Section as though the **hosting arrangement** that is a service contract were an internal-use computer software project to determine when implementation costs of a hosting arrangement that is a service contract are and are not capitalized.

Although a software activity occurs after the cost capitalization criteria are met (see section 3A.2.30) and the activity is not specifically non-capitalizable (see section 3A.2.40), not all costs of, or related to, the activity are necessarily capitalizable.

Direct vs indirect costs

In general, *direct* costs of eligible internal-use software and cloud computing arrangement development and implementation activities are capitalized, while *indirect* costs (i.e. general and administrative and overhead costs) are not. Examples of indirect costs that would generally not be eligible for capitalization include an allocation of lease cost for the space used by the software developers or depreciation of computer servers not dedicated to the development of the internal-use software. [350-40-30-3, 30-5]

Questions 3A.2.210 to 3A.2.230 address issues we have fielded in practice around whether certain types of costs are direct or indirect costs.

Observation

Direct vs indirect costs

AcSEC excluded indirect costs from the capitalizable cost pool for internal-use software because it concluded a full-costing approach would be complex to apply in an internal-use software context. In doing so, AcSEC acknowledged such costs may be part of the cost of the internal-use software asset and that excluding such costs from the cost basis of an internal-use software asset means its basis will differ in that respect from inventory and property, plant and equipment. [SOP 98-1.80]

Timing of the activity

Entities look to the *nature* of the activities instead of their timing or order of incurrence when determining the accounting for the related costs. For example, an entity would capitalize its eligible coding and testing costs for a software project incurred after the capitalization threshold has been met regardless of when during the capitalization window (see section 3A.2.30) those costs were incurred or whether those costs were incurred before or after other activities were performed. [350-40-55-4]

Software data costs

With the rise of AI, questions have arisen about the proper accounting treatment of costs to license or otherwise acquire data. Section 2.8 discusses accounting for software data and data infrastructure costs.

Interest costs

Interest costs incurred to develop and/or implement internal-use software or a CCA are capitalized based on the guidance in Subtopic 835-20 (interest capitalization). This includes suspending interest cost capitalization if the entity suspends substantially all its software development or implementation activities for reasons other than those permitted under the subtopic. [350-40-30-2, 835-20-25-4]

Observation

Interest capitalization rationale

The following contrasting scenarios illustrate the economic rationale underlying the Subtopic 350-40 requirement to capitalize interest costs.

Scenario 1: Third-party software development

An entity engages a third party to develop internal-use software. The development period lasts 18 months and payment is made at completion of development. The fees paid to the third party reflect the implicit financing of the development costs by the third party.

Capitalizing the fees paid to the third party as the cost basis of the internal-use software therefore capitalizes the implied interest element.

Scenario 2: Internal software development

The entity undertakes the same software development project as in Scenario 1 internally. The entity manages the project internally and makes regular payments for payroll and payroll-related costs incurred during development. The entity incurs financing costs (either by borrowing additional funds or using funds that could otherwise be used to repay outstanding debt obligations) during development.

For the internal-use software to reflect a comparable cost basis to that in Scenario 1, the entity must capitalize its interest costs as part of the cost basis of the internal-use software asset.

However, if the entity did not incur interest costs then it follows that none would be capitalized, and the cost basis of the software would differ between the two scenarios.

Questions and Examples

Question 3A.2.190

Is share-based compensation capitalizable?

Background: It may frequently be the case that employees involved in software development or implementation activities earn share-based compensation.

Interpretive response: Yes. Paragraph 350-40-30-1(b) states that payroll and payroll-related costs are capitalizable to internal-use software to the extent they relate to employees who are directly associated with and who devote time to the internal-use software project. While this Codification paragraph does not specifically state that payroll and payroll-related costs include share-based compensation, we believe it is clear that share-based compensation is included in this broader category of payroll and payroll-related costs.

Question 3A.2.200

Are performance-based share award costs capitalizable if incurred after the software project is substantially complete and ready for its intended use?

Background: Consider a scenario in which a software developer employee of the entity has performance-based share awards. The employee worked on a significant internal-use software development project for which all substantial testing is complete when it becomes probable the performance-based award target will be met. Assume it does not become probable the target will be reached until shortly before the performance measurement date. The awards were granted before the project was substantially complete.

Under Topic 718, the entity did not recognize any compensation cost for the performance-based share awards before achievement of the award target became probable. See paragraph 4.092 in KPMG Handbook, Share-based payment.

In this scenario, the question arises about whether any of the share-based compensation cost should be capitalized to the internal-use software asset for the software development project the employee worked on, assuming the employee spent time working on capitalizable activities (see section 3A.2.40).

Interpretive response: In general, if the compensation cost is not yet incurred under Topic 718 because the performance target is not probable of achievement before the internal-use software development project is substantially complete and ready for its intended use, we believe it should be expensed when it is incurred. [350-40-25-14]

However, if the performance target becomes probable of achievement before the software project is substantially complete, those costs should be capitalized to the extent they relate to capitalizable activities.

Example 3A.2.30

Capitalization of performance-based share award costs

Scenario 1: Performance target not probable of achievement before software is substantially complete and ready for its intended use

ABC Corp. developed internal-use Software Application using internal resources.

On January 1, Year 1, ABC granted Employee restricted stock units (RSUs) that vest if ABC successfully completes an IPO within five years at a specified enterprise valuation. On January 1, Year 1, it was not probable the performance condition would be met.

The following additional facts are relevant.

— ABC completed all substantial testing of Software Application, and it went live in ABC's production environment on January 1, Year 3.

— Between January 1, Year 1 and January 1, Year 3, Employee spent 50% of their work time on development activities that qualify for cost capitalization.

— ABC successfully completed an IPO at a valuation above that required for the RSUs to vest on October 1, Year 4.

— Given the uncertainty associated with successfully completing the IPO and the valuation that would ultimately result, it was not probable that the RSUs performance condition would be met until the IPO was completed.

In this scenario, ABC concludes none of the compensation cost for Employee's RSUs is capitalizable as part of the cost of Software Application. This is because the cost was not incurred under Topic 718 until after all substantial testing of Software Application was completed.

Scenario 2: Performance target probable of achievement before software is substantially complete and ready for its intended use

ABC Corp. developed internal-use Software Application using internal resources.

On January 1, Year 1, ABC granted Employee restricted stock units (RSUs) that vest if Software Application goes live and successfully achieves a specified efficiency target for ABC from its use by the end of its first year in production. On January 1, Year 1, it was not probable the performance condition would be met.

The following additional facts are relevant.

— ABC completed all substantial testing of Software Application, and it went live in ABC's production environment on January 1, Year 3.

— At January 1, Year 3 ABC assessed, based on its testing to-date, that Software Application will meet its design requirements; and therefore, will meet the performance condition – i.e. achieve the specified efficiency target.

— The performance condition is officially achieved on December 31, Year 3.

— Between January 1, Year 1 and January 1, Year 3, Employee spent 50% of their work time on development activities that qualify for cost capitalization.

— Total estimated compensation cost of the award at the grant date was $120,000.

Until January 1, Year 3, ABC recognized no compensation cost for this award because achievement of the performance condition was not probable.

On January 1, Year 3 when the performance condition became probable of achievement, ABC recognized $80,000 of compensation cost.

$120,000 total cost of the award × (24 months since grant date ÷ 36 months total implied service period) = $80,000

Of the $80,000, $40,000 was allocable to the Software Application internal-use software asset. This $40,000 is based on Employee's 50% dedication to capitalizable development activities during the period from January 1, Year 1 to January 1, Year 3.

$120,000 total cost of the award × (24 months working on Software Application ÷ 36 months total implied service period) × 50% = $40,000

Question 3A.2.210

Are fees paid to a vendor under an IaaS arrangement capitalizable to an internal-use software project?

Background: Entities may use an IaaS instance to host internal-use software under development. When this is the case, the question arises as to whether the IaaS hosting service fees paid to the cloud service provider during the software's development are capitalizable.

This question does not address the accounting for any implementation costs incurred related to the IaaS arrangement; those costs are subject to the guidance on CCA implementation costs discussed elsewhere in this section 3A.2.

Interpretive response: It depends. Determining when IaaS fees are capitalizable direct costs of an internal-use software project may involve judgment and depend heavily on the specific facts and circumstances, especially as IaaS models continue to evolve. Whether the IaaS fees incurred can be directly attributed to a particular software project such that the IaaS fees are direct costs instead of indirect costs may be particularly important to this analysis. For example:

— If an entity incurs incremental IaaS fees (e.g. under an 'on-demand' pricing structure) for additional, identifiable compute capacity specifically to host internal-use software during the software's development, we believe those costs would generally qualify as capitalizable, external direct costs under paragraph 350-40-30-1(a)(1).

— By contrast, IaaS fees incurred under a fixed-term (e.g. one or three years), reserved capacity arrangement would likely not qualify for capitalization unless that reserved capacity was obtained specifically for an identified software project and could not be repurposed when the project was completed or abandoned (i.e. before completion). Absent that, any fees allocated to a software project(s) undertaken during the reservation period would, in our view, be an indirect cost akin to lease cost for space used by the software developers or depreciation of computer servers not dedicated to the development of internal-use software (discussed earlier in this section).

Question 3A.2.220**

Are fees paid to a third-party for access to a Large Language AI Model capitalizable to an internal-use software project?

Background: Entities may use large language AI models (LLMs) to assist in developing internal-use software subject to Subtopic 350-40. These models may support development activities such as code generation, testing, documentation or optimization. LLMs are often accessed via the cloud only, under consumption-based or subscription pricing arrangements. When this is the case, the question arises as to whether the fees paid to access the LLM are capitalizable costs of the software project the LLM is being used to complete.

This question does not address the accounting for any implementation costs incurred related to the LLM arrangement (assuming it is a CCA – see section 2.5); those costs are subject to the guidance on CCA implementation costs discussed elsewhere in this section 3A.2.

Interpretive response: It depends. Determining whether LLM fees are capitalizable direct costs of an internal-use software project requires careful consideration of the nature of the arrangement and the specific facts and circumstances. The key consideration is whether the LLM access fees can be directly attributed to a particular software project and whether they represent incremental and direct external costs.

— If an entity incurs incremental LLM fees (e.g. under a usage or consumption-based pricing model) for a particular internal-use software project, those fees may qualify as capitalizable external direct costs. In this circumstance, the usage is analogous to paying for incremental IaaS compute capacity obtained exclusively to develop a particular software project (see Question 3A.2.210).

— Conversely, LLM fees incurred under a subscription model would likely not qualify for capitalization unless the LLM subscription was obtained

specifically for an identified software project and could not be repurposed when the project was completed or abandoned before completion. Absent that, any fees allocated to a software project(s) undertaken during the subscription period would, in our view, be an indirect cost akin to (1) lease cost for space used by the software developers or depreciation of computer servers not dedicated to the development of internal-use software (discussed earlier in this section) and (2) the IaaS fees in Question 3A.2.210.

Question 3A.2.230**

Are data-related infrastructure costs direct or indirect costs?

Background: See Question 2.8.10 on accounting for data-related infrastructure costs.

Interpretive response: We believe data-related infrastructure costs are generally indirect costs unless they relate to infrastructure obtained specifically for and dedicated to data for a *specific* internal-use software project – e.g. servers or compute capacity obtained specifically and exclusively to house data that has no other use to the entity other than training a specific AI internal-use software application.

3A.2.60 Upgrades and enhancements#

Excerpt from ASC 350-40

25 Recognition

General

> Upgrades and enhancements

25-17A Upgrades and enhancements are defined as modifications to existing internal-use software that result in additional functionality—that is, modifications to enable the software to perform tasks that it was previously incapable of performing. Upgrades and enhancements normally require new software specifications and may also require a change to all or part of the existing software specifications. In order for costs of specified upgrades and enhancements to internal-use computer software to be evaluated for capitalization in accordance with paragraphs 350-40-25-17B through 25-17E, it must be probable that those expenditures will result in additional functionality.

25-17B Internal and external costs incurred for upgrades and enhancements shall be expensed or capitalized in accordance with paragraphs 350-40-25-1, 350-40-25-4 through 25-5, 350-40-25-12 through 25-14, and 350-40-25-17.

25-17C Internal and external costs incurred for maintenance shall be expensed as incurred.

25-17D Entities that cannot separate internal costs on a reasonably cost-effective basis between maintenance and relatively minor upgrades and enhancements shall expense such costs as incurred.

25-17E External costs incurred under agreements related to specified upgrades and enhancements shall be expensed or capitalized in accordance with paragraphs 350-40-25-1, 350-40-25-4 through 25-5, 350-40-25-12 through 25-14, and 350-40-25-17. If maintenance is combined with specified upgrades and enhancements in a single contract, the cost shall be allocated between the elements as discussed in paragraph 350-40-30-4 and the maintenance costs shall be expensed over the contract period. However, external costs related to maintenance, unspecified upgrades and enhancements, and costs under agreements that combine the costs of maintenance and unspecified upgrades and enhancements shall be recognized in expense over the contract period on a straight-line basis unless another systematic and rational basis is more representative of the services received.

Implementation Costs of a Hosting Arrangement That Is a Service Contract

25-18 An entity shall apply the General Subsection of this Section as though the **hosting arrangement** that is a service contract were an internal-use computer software project to determine when implementation costs of a hosting arrangement that is a service contract are and are not capitalized.

Upgrades and enhancements are defined as changes to existing internal-use software that result in additional software functionality. Functionality refers to the software's ability to perform a task. [350-40-25-7]

Software development and implementation costs for upgrades and enhancements, including specified upgrades and enhancements to licensed internal-use software for the licensee, are capitalized or expensed on the same basis as if those costs were incurred to develop and implement new software. [350-40-25-8]

— Costs that would be capitalized when developing or implementing new software are capitalized when developing or implementing an upgrade or enhancement.

— Costs that would be expensed as incurred when developing or implementing new software are expensed as incurred when developing or implementing an upgrade or enhancement.

Specified upgrades

A specified upgrade right is generally a software vendor's explicit commitment to deliver, or agreement to deliver on a when-and-if available basis, a specific version of the software or an upgrade with specific features and functionality. Any discussion in the contract with the customer of

possible features and functionality of future versions of the software may give rise to a specified upgrade right.

A software vendor may implicitly grant its customer a specified upgrade right, without there being explicit discussion in the contract. This is the case when the software vendor provides assurance to the customer that a future product release will contain specific features and/or functionality. This might occur, for example, through communication of a detailed software product roadmap (i.e. marketing materials) or in a non-contractual response to a customer's request for proposal.

As a practical matter, claims made in marketing materials available to customers and commitments by sales personnel may be considered by the customer to be part of the arrangement, and therefore represent a specified upgrade right. The specific facts and circumstances should be evaluated on a case-by-case basis, and the entity may need to consult with its financial and legal advisers to determine if a specified upgrade right has been granted implicitly to a customer.

Factors to consider when evaluating whether an implicit specified upgrade right has been granted to a customer include the following.

- ***Upgrade/enhancement detail*** – the level of detail of the features, functionality and general release timeframe of the future product that has been provided to the customer. The greater the level of detail and the closer the release date of the enhancement to the initial contract, the greater the likelihood that the software vendor has created an expectation by the customer of the release of the future identifiable upgrade/enhancement that may have affected the customer's purchase decision.
- ***Caveat language*** – whether the software vendor's use of caveat language detailing the product roadmap and future development efforts in a license arrangement gives rise to uncertainty about whether the customer will receive the product upgrades/enhancements. The greater the level of uncertainty about the future delivery of an upgrade/enhancement, the less likely that the software vendor has created an expectation by the customer of the release of the future identifiable upgrade/enhancement.
- ***Customer communication*** – whether the software vendor communicates the features, functionality and timeframe for general release of the future product, or communicates a release to only certain identifiable customers. The broader the intended distribution of the product and the more specific the communication is about functionality and features of the program, the greater the likelihood that customers would expect to receive the specified product upgrades/enhancements.
- ***Software vendor's history*** – a software vendor's history of charging a substantive amount for product upgrades/enhancements would indicate that the product may not be a specified upgrade.
- ***Customer request*** – whether the customer requests or requires a roadmap to specify specific features and/or functionality that are not available currently in the marketed product. Such a request or

requirement would indicate that the product roadmap may be a specified upgrade.

Example 3A.2.40**

Implicit specified upgrade

ABC Corp. issues a request for proposal (RFP) to a software vendor that indicates ABC desires incremental XYZ functionality in the general ledger software it plans to purchase. In a written response to ABC's RFP, the software vendor states that although XYZ functionality currently is not available in Version 4.0 of Product A (software vendor's currently available general ledger software), the software vendor anticipates that XYZ functionality will be available in Version 4.1 of Product A, which is expected to be released in approximately six months.

Subsequent to the RFP process, the software vendor enters into a contract with ABC to transfer a license to Version 4.0 of Product A and to provide ABC with a right to any future updates, upgrades and enhancements to Product A, when-and-if available, for a period of one year. The arrangement does not explicitly discuss the XYZ functionality.

The software vendor has implicitly promised to provide XYZ functionality to ABC through its communication in the RFP. This created a reasonable expectation on ABC's part that the software vendor will deliver the functionality through a future update. Therefore, the XYZ functionality constitutes a specified upgrade.

Maintenance costs

Internal costs to maintain internal-use software – i.e. to maintain its existing functionality and ensure proper operation – are expensed as incurred and are accounted for separately from costs to develop and implement upgrades and enhancements. [350-40-25-9]

If an entity cannot separate internal costs of maintenance from internal costs of relatively minor upgrades and enhancements on a reasonably cost-effective basis, it expenses all such costs as incurred. When applied, this guidance ensures the entity does not capitalize maintenance costs. Subtopic 350-40 does not define 'relatively minor' or 'reasonably cost-effective'. [350-40-25-10]

External costs to maintain internal-use software – e.g. costs paid to a software vendor to maintain licensed software – are expensed as incurred. [350-40-25-11]

Post-contract customer support

Software licensees typically contract for PCS. PCS generally includes not only software maintenance (e.g. bug fixes and other updates that maintain existing functionality), but also technical support and the right to receive upgrades and enhancements on a when-and-if available (i.e. unspecified) basis.

PCS cost is generally recognized on a straight-line basis over the PCS term, unless another systematic and rational basis is more representative of the pattern in which the services are received. [350-40-25-11]

The right to receive unspecified upgrades and enhancements is accounted for as a service even if the entity has a valid expectation it will receive upgrades and enhancements that will provide additional software functionality. [350-40-25-11]

Specified upgrades and enhancements are not part of PCS and follow the requirements for upgrades and enhancements outlined above.

Question 3A.2.240

When is it appropriate to recognize PCS expense on a basis other than straight-line?

Interpretive response: In general, we believe a straight-line basis is appropriate; we would expect recognition on a basis other than straight-line to be infrequent.

This is because the typical PCS elements normally qualify as 'stand-ready' services; the software vendor stands ready to:

- provide technical support when-and-as needed; and
- transfer maintenance updates (including bug fixes), upgrades and enhancements when-and-if developed.

When recognizing expense for a stand-ready service, it is generally appropriate to consider the entity's consistent and equal access to the service throughout the service period. This supports a straight-line expense recognition pattern and would make an other than straight-line pattern – e.g. on the basis of the entity's expected usage of technical support or timing/significance of upgrades and enhancements – generally inappropriate. [350-40-35-13]

The components of PCS may not always be stand-ready services, and an expense recognition pattern other than straight-line expense may be appropriate when they are not. The following scenarios typically suggest that PCS is not a stand-ready service.

- The software vendor promises a defined number of upgrades or enhancements, suggesting the entity has a right to that specified number of upgrades and enhancements instead of a right to a stand-ready service.
- The entity's right to technical support is for a defined number of support calls or events (or up to a specified number of support events that is substantive – i.e. it is not in excess of any realistic expectation of the entity's use of those services).

Combined fees

The components of PCS are typically not priced separately; an entity typically pays one fee for PCS (e.g. as a percentage of the software license fee). If that is the case, and one (but not all) of the components is not a stand-ready service, we believe it is reasonable to either: [350-40-25-11, 30-4]

— separate the components on a relative stand-alone price basis (see section 2.7) and recognize expense related to each separately; or
— select a single attribution pattern for the combined expense that is reflective of the pattern in which the combined PCS services are received.

An entity should apply its approach consistently to similar circumstances.

Question 3A.2.250

Does an upgrade or enhancement include software changes that extend the software's life but do not add functionality?

Interpretive response: No. Under Subtopic 350-40, an upgrade or enhancement must result in additional software functionality (i.e. the ability of the software to perform additional tasks). Activities undertaken to extend the useful or economic life of internal-use software are maintenance activities, the costs of which are expensed as incurred. [350-40-25-7, 25-9, SOP 98-1.73]

The conclusion that an extension of the useful or economic life of the software is not an upgrade or enhancement differs from that in Subtopic 985-20, which defines a 'product enhancement', in part, as an improvement that extends the life of the software (see chapter 5). AcSEC acknowledged this difference, assessing there to be different considerations for a user of internal-use software versus a software vendor. [985-20 Glossary, SOP 98-1.73]

Question 3A.2.260

Are modifications that increase only the efficiency of internal-use software upgrades or enhancements?

Interpretive response: No. Under Subtopic 350-40, an upgrade or enhancement must result in additional software functionality (i.e. the ability of the software to perform additional tasks). Modifications that increase how efficiently the software system operates but do not add to the tasks the software is able to perform are not upgrades or enhancements. [350-40-25-7]

Question 3A.2.270

Does modifying software so that it can be hosted and operated in a public cloud or on an additional hardware platform or operating system create 'additional functionality'?

Interpretive response: In general, yes. We believe the ability for software to operate in a public cloud, on an additional hardware platform or with a new operating system is added functionality.

Modifications of this nature differ from merely extending the useful or economic life of the software and differ from maintenance activities. This is because the added ability to be hosted in the public cloud or to operate on a different hardware platform or operating system permits additional uses of the software and typically exists together with – i.e. is incremental to – the software's original ability to be hosted other than in the public cloud or to operate on the initial hardware platform or operating system.

Question 3A.2.280

Does the ability to use software in new geographies or with new languages qualify as 'additional functionality'?

Interpretive response: In general, yes. Expanding the software's ability to be used in additional geographies or with additional languages would generally be viewed as additional functionality. The entity now can perform additional tasks using the software – e.g. process transactions in the expansion territories or in the added languages.

3A.2.70 FASB examples**

The following excerpt contains four examples added to Subtopic 350-40 by ASU 2025-06. These examples illustrate how Subtopic 350-40 applies after the adoption of ASU 2025-06:

- **Example 1:** Enterprise resource planning systems
- **Example 2:** Mobile applications
- **Example 3:** Novel technologies
- **Example 4:** Website development.

Excerpt from ASC 350-40

55 Implementation Guidance and Illustrations

General

• > Example 1: Implementation and Customization of an Enterprise Resource Planning System

55-5 On February 1, 20X3, a professional services company starts internal discussions to transform its information technology by implementing an enterprise resource planning system to support finance, human resources, accounting, and client relationships.

55-6 After researching different solutions and performing its due diligence procedures, management executes a contract with a third party on August 1, 20X3, to implement and customize a hybrid solution that offers on-premises software and cloud computing services for the enterprise resource planning system. Within this solution, the third party offers different functionality and features, and the company will have to make customization decisions throughout the development process to select which functionality and features it wants included.

55-7 The company assesses whether the internal and external costs to implement and customize the enterprise resource planning system meet the capitalization requirements in paragraphs 350-40-25-12 through 25-12A, as follows:

a. As part of its assessment under paragraph 350-40-25-12(c), the company evaluates whether there is significant development uncertainty in accordance with paragraph 350-40-25-12A. As of August 1, 20X3, the company determines that:

    1. It has identified the significant **performance requirements** and does not expect to continue to substantially revise those requirements because the only expected customization is selecting from existing functionality and features.
    2. The software being developed does not have technological innovations or novel, unique, or unproven functions or features because the company has selected a developed solution.

    Therefore, as of August 1, 20X3, the company determines that significant development uncertainty does not exist.

b. The company evaluates the requirements in paragraph 350-40-25-12 to determine when to begin capitalizing software costs:

    1. The company determines that management authorized and committed to funding the software project on August 1, 20X3, when it executed the contract with the third party.
    2. Considering all other relevant facts and circumstances (for example, the company has engaged an established and experienced third party to implement and customize the software), as of August 1, 20X3, the company determines that it is probable that the software project will be

completed and the software will be used to perform the function intended.

55-8 As a result, on August 1, 20X3, the company determines that the capitalization requirements in paragraphs 350-40-25-12 through 25-12A are met, and it begins capitalizing eligible software costs, including those related to implementation and customization of the on-premises software license and those related to implementation of the cloud computing service features of the hybrid solution.

• > Example 2: Development of a Mobile Application

55-9 A company is in the process of internally developing X-Crowd, which is a mobile application that will allow users to see how crowded a restaurant or store is on the basis of a user's real-time input. An internet connection is required to be able to access the application.

55-10 On February 1, 20X1, management approved funding for internal development of the application. However, the company has not yet identified what functions and features would be included in the application. Through November 30, 20X1, the company continues to develop the functions and features of the application, including getting feedback on preliminary product versions from user groups and modifying the development of those functions and features to incorporate the feedback. On December 1, 20X1, management determines that it has identified the significant performance requirements (the significant functions and features it needs the application to have), and it does not anticipate substantial changes to those requirements. Throughout the development of X-Crowd, management determines that the application does not have technological innovations or novel, unique, or unproven functions or features.

55-11 The company assesses whether the internal and external costs to develop the application meet the capitalization requirements in paragraphs 350-40-25-12 through 25-12A, as follows:

a. As part of its assessment under paragraph 350-40-25-12(c), the company evaluates whether there is significant development uncertainty in accordance with paragraph 350-40-25-12A. As of February 1, 20X1, the company determines that:

    1. It has not yet identified the significant performance requirements.
    2. The software being developed does not have technological innovations or novel, unique, or unproven functions or features.

    Therefore, as of February 1, 20X1, the company determines that significant development uncertainty exists and, in accordance with paragraph 350-40-25-12A, the software project does not meet the requirements to begin capitalizing software costs in paragraph 350-40-25-12(c).

b. As of December 1, 20X1, the company determines that:

    1. It has identified the significant performance requirements and does not expect to continue to substantially revise those requirements.
    2. The software being developed does not have technological innovations or novel, unique, or unproven functions or features.

Therefore, as of December 1, 20X1, the company determines that significant development uncertainty has been resolved.

c. As of December 1, 20X1, the company evaluates the requirements in paragraph 350-40-25-12 to determine when to begin capitalizing software costs:

    1. The company determines that management authorized and committed to funding the software project on February 1, 20X1, when it approved funding for internal development of the application.
    2. Considering all other relevant facts and circumstances, as of December 1, 20X1, the company determines that it is probable that the software project will be completed and the software will be used to perform the function intended.

55-12 As a result, on December 1, 20X1, the company determines that the capitalization requirements in paragraphs 350-40-25-12 through 25-12A are met, and it begins capitalizing eligible software costs.

• > Example 3: Development of a Novel Technology

55-13 On January 1, 20X1, a software development company starts discussions to develop software with novel functionality.

55-14 On February 1, 20X1, management completes its due diligence procedures, approves a budget to internally develop the software, and allocates an internal development team to start developing the novel software. At the time that the company started discussions and management approved a budget, the software still had novel functionality.

55-15 On March 1, 20X3, the company resolves the uncertainty related to the novel functionality through coding and testing. Additionally, on March 1, 20X3, the company determines that it does not expect substantial changes to the identified significant performance requirements (the significant functions and features) included in the software. On April 1, 20X3, the company determines that all substantial testing is completed.

55-16 The company assesses whether the internal and external costs to develop the software meet the capitalization requirements in paragraphs 350-40-25-12 through 25-12A, as follows:

a. As part of its assessment under paragraph 350-40-25-12(c), the company evaluates whether there is significant development uncertainty in accordance with paragraph 350-40-25-12A. As of February 1, 20X1, the company determines that:

    1. It has not yet identified the significant performance requirements.
    2. The software being developed has novel functionality and that functionality has not been resolved through coding and testing.

    Therefore, as of February 1, 20X1, the company determines that significant development uncertainty exists and, in accordance with paragraph 350-40-25-12A, the software project does not meet the requirements to begin capitalizing software costs in paragraph 350-40-25-12(c).

b. As of March 1, 20X3, the company determines that:

   1. It has identified the significant performance requirements and does not expect to continue to substantially revise those requirements.
   2. The uncertainty related to the novel functionality has been resolved through coding and testing.

   Therefore, as of March 1, 20X3, the company determines that significant development uncertainty has been resolved.

c. As of March 1, 20X3, the company evaluates the requirements in paragraph 350-40-25-12 to determine when to begin capitalizing software costs:

   1. The company determines that management authorized and committed to funding the software project on February 1, 20X1, when it approved a budget and allocated an internal development team.
   2. Considering all other relevant facts and circumstances, as of March 1, 20X3, the company determines that it is probable that the software project will be completed and the software will be used to perform the function intended.

55-17 As a result, on March 1, 20X3, the company determines that the capitalization requirements in paragraphs 350-40-25-12 through 25-12A are met, and it begins capitalizing eligible software costs. On April 1, 20X3, the company determines that the software project is substantially complete and ready for its intended use because all substantial testing has been completed. Therefore, the company ceases capitalizing eligible software costs on April 1, 20X3, in accordance with paragraph 350-40-25-14.

• > Example 4: Development of a Website

55-18 An animal rescue organization starts discussions on June 15, 20X5, to develop a website that will be used to share information with users of the organization, including hours of operation, contact details, animals available for adoption, and standard adoption procedures.

55-19 After researching different website developers and performing its due diligence procedures, management executes a contract with a third party on August 1, 20X5, to develop a website for the organization. The third party is an established website developer and offers different templates that the organization can use to create its website. In addition to website development fees paid to the third party, the organization incurs costs:

a. To obtain and register an internet domain
b. To input content into the website
c. To develop initial graphics for the website
d. To register the website with internet search engines
e. For ongoing website hosting fees.

55-20 The organization assesses whether the internal and external costs to develop the website meet the capitalization requirements in paragraphs 350-40-25-12 through 25-12A, as follows:

a. As part of its assessment under paragraph 350-40-25-12(c), the organization evaluates whether there is significant development

uncertainty in accordance with paragraph 350-40-25-12A. As of August 1, 20X5, the organization determines that:

1. It has identified the significant performance requirements and does not expect to continue to substantially revise those requirements because the website will be created from existing templates that the organization can use to share the information described in paragraph 350-40-55-18.
2. The website being developed does not have technological innovations or novel, unique, or unproven functions or features because it will be developed from existing templates.

Therefore, as of August 1, 20X5, the organization determines that significant development uncertainty does not exist.

b. The organization evaluates the requirements in paragraph 350-40-25-12 to determine when to begin capitalizing costs:

1. The organization determines that management authorized and committed to funding the development of the website on August 1, 20X5, when it executed the contract with the third party.
2. Considering all other relevant facts and circumstances (for example, the organization has engaged an established and experienced third party to develop the website), as of August 1, 20X5, the organization determines that it is probable that the project will be completed and the website will be used to perform the function intended.

55-21 As a result, on August 1, 20X5, the organization determines that the capitalization requirements in paragraphs 350-40-25-12 through 25-12A are met, and it begins capitalizing eligible costs. In evaluating which costs are eligible for capitalization, the organization determines the following:

a. Fees paid to the third party for services to develop the website are evaluated for capitalization in accordance with paragraph 350-40-30-1.
b. Costs incurred to obtain and register the internet domain are evaluated for capitalization in accordance with paragraph 350-40-25-17J.
c. Costs incurred to input content into the website are expensed as incurred in accordance with paragraph 350-40-25-17G.
d. Costs incurred to develop initial graphics for the website are evaluated for capitalization in accordance with paragraph 350-40-25-17H.
e. Costs incurred to register the website with internet search engines are expensed as incurred in accordance with paragraph 350-40-25-17I.
f. Ongoing website hosting fees are expensed over the period of benefit in accordance with paragraph 350-40-25-17F.

3A.3 Internal-use software licenses

Excerpt from ASC 350-40

15 Scope and Scope Exceptions

General

> Transactions

15-4A The guidance in the General Subsections of this Subtopic applies only to internal-use software that a customer obtains access to in a **hosting arrangement** if both of the following criteria are met:

a. The customer has the contractual right to take possession of the software at any time during the hosting period without significant penalty.
b. It is feasible for the customer to either run the software on its own hardware or contract with another party unrelated to the vendor to host the software.

15-4C Hosting arrangements that do not meet both criteria in paragraph 350-40-15-4A are service contracts and do not constitute a purchase of, or convey a license to, software.

20 Glossary

Hosting Arrangement

In connection with accessing and using software products, an arrangement in which the customer of the software does not currently have possession of the software; rather, the customer accesses and uses the software on an as-needed basis.

25 Recognition

General

> Capitalization of Cost

25-17 Entities often license internal-use software from third parties. A software license within the scope of this Subtopic (see paragraphs 350-40-15-1 through 15-4C) shall be accounted for as the acquisition of an intangible asset and the incurrence of a liability (that is, to the extent that all or a portion of the software licensing fees are not paid on or before the acquisition date of the license) by the licensee. The intangible asset acquired shall be recognized and measured in accordance with paragraphs 350-30-25-1 and 350-30-30-1, respectively.

Excerpt from ASC 350-30

25 Recognition

General

25-1 An intangible asset that is acquired either individually or with a group of other assets shall be recognized.

30 Initial Measurement

General

30-1 An intangible asset that is acquired either individually or with a group of other assets (but not those acquired in a business combination) shall be initially measured based on the guidance included in paragraphs 805-50-15-3 and 805-50-30-1 through 30-4.

Excerpt from ASC 805-50

30 Initial Measurement

Acquisition of Assets Rather than a Business

> Determining Cost

30-1 Paragraph 805-50-25-1 discusses exchange transactions that trigger the initial recognition of assets acquired and liabilities assumed. Assets are recognized based on their cost to the acquiring entity, which generally includes the transaction costs of the asset acquisition, and no gain or loss is recognized unless the fair value of noncash assets given as consideration differs from the assets' carrying amounts on the acquiring entity's books. For transactions involving nonmonetary consideration within the scope of Topic 845, an acquirer must first determine if any of the conditions in paragraph 845-10-30-3 apply. If the consideration given is nonfinancial assets or in substance nonfinancial assets within the scope of Subtopic 610-20 on gains and losses from the derecognition of nonfinancial assets, the assets acquired shall be treated as noncash consideration and any gain or loss shall be recognized in accordance with Subtopic 610-20.

30-2 Asset acquisitions in which the consideration given is cash are measured by the amount of cash paid, which generally includes the transaction costs of the asset acquisition. However, if the consideration given is not in the form of cash (that is, in the form of noncash assets, liabilities incurred, or equity interests issued) and no other generally accepted accounting principles (GAAP) apply (for example, Topic 845 on nonmonetary transactions or Subtopic 610-20), measurement is based on either the cost which shall be measured based on the fair value of the consideration given or the fair value of the assets (or net assets) acquired, whichever is more clearly evident and, thus, more reliably measurable. For transactions involving nonmonetary consideration within the scope of Topic 845, an acquirer must first determine if any of the conditions in paragraph 845-10-30-3 apply. If the consideration given is nonfinancial assets or in substance nonfinancial assets within the scope of Subtopic 610-20, the assets acquired shall be treated as noncash consideration and any gain or loss shall be recognized in accordance with Subtopic 610-20.

Entities frequently do not develop internal-use software; they license it from third parties. For example, most entities license their internal-use enterprise resource planning (ERP) software.

Entities may obtain a license to internal-use software through a 'hosting arrangement' if that arrangement is determined to grant the entity a license to the hosted software. If the hosting arrangement does not include a software license, it is a service arrangement. Section 2.5 addresses how to determine if a hosting arrangement includes a software license. [350-40-15-4A – 15-4C]

When licensing internal-use software, the entity recognizes an intangible asset for the software license. Its cost basis includes the sum of the following: [350-40-25-17, 30-4, 805-50-30-1 – 30-2]

- the license fees, which may be an allocated number if there are multiple elements in the arrangement – e.g. license and other services, such as PCS or hosting services in a hosting arrangement that includes a license (see section 2.7);
- capitalized development costs – e.g. customization or modification of the software, and implementation costs (see section 3A.2); and
- transaction costs, if any.

If all or a portion of the software license fees are unpaid when the license is obtained, a liability is recognized for the unpaid fees. In that case, the initial measurement of the license asset is either: [350-40-25-17, 805-50-30-2]

- the cost basis of the license asset, measured based on the fair value of the license fees liability incurred; or
- the fair value of the acquired license.

The initial measurement basis of the license asset is not a choice; an entity must use the former if the fair value of the license fees liability is more reliably measurable than the fair value of the acquired license (see Question 3A.3.30). [805-50-30-2]

Section 6.2.10 addresses the subsequent measurement of intangible software license assets and unpaid license fee liabilities.

Question 3A.3.10#

At what date is a new internal-use software license asset and any associated liability recognized?

Interpretive response: Subtopic 350-40 does not provide guidance in this regard. Therefore, we believe it is appropriate to consider the guidance in Subtopic 350-30 on recognizing identifiable intangible assets and the CON 8 definition of an asset. In this regard, we believe the scope exclusion in paragraph 350-30-15-4(f) does not apply because Subtopic 350-40 does not address when to recognize an internal-use software *license* asset. [350-30-25-4, CON 8.E16 – E17]

Thereunder, we believe the license should be recognized if (1) it qualifies as an 'identifiable asset' and (2) when the entity (customer) has the present right to the economic benefits of the license.

Identifiable assets include those that meet either the contractual-legal criterion or separability criterion for being an asset. Software licenses will generally meet

the contractual-legal criterion; that is, the software license gives rise to contractual rights for the entity stemming from the applicable software license agreement. [350-30 Glossary]

We believe this asset should be recognized *when* it meets the CON 8 definition of an asset for the entity, meaning it constitutes a *present* right of the entity to economic benefits. [CON 8.E16 – D17]

Any license liability should be recognized at the same time because the entity's financial obligation arises from the obligating event of the software vendor making the software available for the entity's use.

We believe the software license constitutes a present right to economic benefits for the entity when the entity:

a. takes possession, or has the right to take possession, of a copy of the software; and
b. has the right to begin to use and benefit from the license.

We believe (a) occurs when:

— a copy of the software has been physically delivered to the entity;
— the entity has taken possession of the software via download;
— the entity has been provided with the access code (or key) that allows it to take immediate possession of the software; or
— the entity has the present, enforceable right to request an access code (or key) at any time and transfer of such code (or key) is effectively administrative or perfunctory.

We believe (b) occurs at the earlier of:

— the beginning of the contractual license period; and

— when the software is made available for the entity (or any other party on its behalf, including the software vendor) to undertake the development or implementation activities (e.g. entity-specific customizations, installation to the entity's hardware or third-party hosting environment or user acceptance testing) necessary to make it ready for the entity's intended use. Concluding the entity has the right to use and benefit from the software at this point in time is analogous to the conclusion that a lease commences when the underlying asset has been made available to be modified or customized for the lessee's use (e.g. leasehold improvements installed). See section 5.1 of KPMG Handbook, Leases. [842 Glossary, 842-10-55-19 – 55-21]

The contractual license period usually begins before the software is available for the entity to customize or implement it. An entity usually needs to have an in-force license to the software before it can begin either custom development or implementation activities, even if those activities will be undertaken by the software vendor. However, this may not always be the case.

It is possible capitalizable activities will occur before the license qualifies for recognition by the entity. We believe the capitalizable costs of such activities should be capitalized as part of an in-process intangible asset that will include the costs of the software license once it is recognized. This is provided the activities do not give rise to their own asset (e.g. a software interface that qualifies as internal-use software on its own) or constitute activities to ready a

different asset for its intended use (e.g. costs to ready a computer server for its intended use hosting the internal-use software). This approach is similar to how an entity constructing a building accounts for costs to clear or grade land before construction of the building structure commences.

Availability of software when software vendor performs certain activities

For the entity, or a third party (including the software vendor) on its behalf, to install the licensed software or undertake other implementation activities such as commencing data migration or user acceptance testing, it must have a copy of the software.

However, if the software vendor is undertaking customization or configuration activities, it might not provide a copy of the software (or make the software available to the entity for download) before it completes those activities. Instead, the vendor might provide a copy of the customized or configured software only.

In that case, we believe it is generally appropriate to recognize the software license when those activities commence, just as the entity would if it or a third party were undertaking those activities. This approach is consistent with both of the following.

— The commencement date for a lease does not change if the lessor constructs lessee-owned leasehold improvements, rather than the lessee (or a third party on its behalf). See Example 5.1.10 in KPMG Handbook, Leases.

— Software vendors often recognize software customization services over the customization period on the basis that the customization enhances an asset the customer controls. See Question F200 in KPMG Handbook, Revenue for software and SaaS).

However, because there is no explicit licensee guidance for this scenario, we believe there may be diversity in practice. Some entities may not recognize the software license until the customized/configured software is made available for installation. In the absence of further guidance from the FASB or the SEC staff, and assuming the entity does not have a present enforceable right to take possession of the unmodified/non-configured software, we believe this alternative approach is also acceptable.

Example 3A.3.10#

Initial recognition and measurement of an internal-use software license asset – license fees prepaid

Customer and Vendor execute a five-year term license on December 1, Year 1. The contractual license term commences on January 1, Year 2, which coincides with Vendor providing the key necessary for Customer to download the software.

Customer prepays the $300,000 license fee and $60,000 for Year 2 PCS on December 31, Year 1.

Customer incurs the following implementation costs related to the software between January 1 and May 31, Year 2. On May 31, Year 2, Customer concludes that the software is ready for its intended use.

This example assumes that all amounts for the license and services (including PCS) from Vendor reflect their stand-alone prices.

Cost	Amount
Installation, configuration and testing[1]	$40,000
Data conversion/migration[2]	15,000
Training[2]	3,000
Software interfacing (of Vendor software to other Customer on-premise and hosted applications)[1]	20,000
Total	**$78,000**

Notes:
1. Implementation activities performed by Vendor.
2. Implementation activities performed by Customer or a third-party consultant.

At December 31, Year 1, Customer has recognized a $300,000 prepaid asset and $60,000 in prepaid PCS.

On January 1, Year 2 Customer recognizes the software license, on the basis that it has obtained the key necessary to download the software and begin implementation efforts, and reclassifies the $300,000 prepaid asset to an in-process intangible software license asset.

— Customer capitalizes the $40,000 in installation, configuration and testing costs as part of the cost basis of the software license asset as those costs are incurred between January 1 and May 31.

— Customer also capitalizes the $20,000 cost of the software interfaces; those interfaces are capitalized as internal-use software assets separate from the software license asset.

— Customer expenses the $15,000 and $3,000 in data conversion/migration and training costs as incurred during the implementation period.

On May 31, Year 2 (when the internal-use software acquired from Vendor is ready for its intended use), Customer begins amortizing the $340,000 software license asset.

The substantial testing of the new interfaces developed to work with Vendor software is completed at the same time as the testing of Vendor software itself. Therefore, Customer also begins amortizing the software interface assets on that date.

Question 3A.3.20

At what date is an internal-use software license asset recognized for a license renewal?

Background: Assume an entity enters into a three-year software license with a software vendor. The term of that license is January 1, Year 1 – December 31, Year 3. On July 1, Year 3, the parties agree to a three-year extension of that license – i.e. for the period January 1, Year 4 – December 31, Year 6. No renewal option existed in the original license agreement. This is **Scenario 1**.

Consider a variation on this example whereby on July 1, Year 3 the entity formally executes a three-year renewal option that was available to it in the original license agreement (**Scenario 2**).

The question arises about whether the renewal license in these scenarios is a separate intangible asset, and, if so, at what date it (and any related license fees liability) should be recognized.

Interpretive response: Subtopic 350-40 does not provide guidance specific to either background scenario.

Scenario 1: License renewal was not an option in original contract

We believe there is likely diversity in practice, and in the absence of further guidance from the FASB or the SEC staff, we believe either of the following approaches, applied consistently, is acceptable.

Approach 1: Recognize separate renewal license asset only at start of renewal period

Topic 606 treats renewal software licenses as separate and distinct from the initial software license as long as they are not required to be combined under the contract combination guidance. Renewal licenses are not transferred to the customer – i.e. the customer does not obtain control of the renewal license – before the start of the renewal period. Chapter F of KPMG Handbook, Revenue for software and SaaS, discusses this further. [606-10-55-58C(b), 55-392A – 55-392D, ASU 2016-10.BC50(a)]

Under this approach, the entity analogizes to that software *vendor* guidance. It concludes that the renewal license is not yet a present right of the entity to economic benefits before the start of the renewal period. Therefore, it recognizes the renewal software license only at the start of the renewal period (e.g. January 1, Year 4 in the background scenario). If the entity prepays for the renewal license, it would recognize a prepaid asset, rather than an intangible license asset, until it recognizes the renewal license at the start of the renewal period.

Approach 2: Recognize modification of the license asset when renewal is agreed

Topic 606 does not apply to customers in revenue arrangements. Therefore, we believe it is reasonable to not consider the software vendor's accounting when considering that of the entity (customer).

Under this approach, the renewal is accounted for as a modification of the software license (i.e. an extension of the time attribute thereof). The renewal is not treated as a separate or distinct license because the entity already has the rights being renewed and already has a copy of the software.

The cost of the renewal is added to the existing carrying amount of the intangible software license asset on the date the renewal is agreed to by the vendor and the entity (July 1, Year 3 in the background scenario). The useful life of the license asset is extended by the term of the renewal. The change in useful life is accounted for prospectively (see Question 6.2.20).

Scenario 2: License renewal option included in original license agreement

We believe either approach to Scenario 1 is also acceptable for Scenario 2 as an accounting policy election.

Question 3A.3.30

Which is typically more reliably measurable, the fair value of the consideration given or the fair value of the license asset received?

Background: If all or a portion of the software licensing fees are unpaid when the entity recognizes the license (see Questions 3A.3.10 and 3A.3.20), a liability is recognized for those unpaid fees. In that case, the initial measurement of the license asset is either: [350-40-25-17, 805-50-30-2]

- the cost basis of the license asset, measured based on the fair value of the license fees liability incurred; or
- the fair value of the acquired license.

The initial measurement basis of the license asset is not a choice; an entity must use the former if the fair value of the license fees liability is more reliably measurable than the fair value of the acquired license. [805-50-30-2]

This question assumes none of the following apply (consider the guidance in chapter A of KPMG Handbook, Revenue for software and SaaS):

- Topic 606 (revenue from contracts with customers) – e.g. the entity is a software vendor and receives the internal-use software license as full or partial payment for granting a license to its software;
- Subtopic 610-20 (gains and losses from the derecognition of nonfinancial assets) – e.g. the entity receives the internal-use software license as payment for the sale of an asset to a non-customer; or
- Topic 845 (nonmonetary transactions).

The question arises about whether the fair value of the consideration given (e.g. the liability to pay the license fees over the license term) is more reliably measurable than the fair value of the acquired license.

The response to this question assumes the unpaid license fees are payable in cash. For guidance that addresses the acquisition of an asset, including an

internal-use software license, for consideration other than cash, see section 3.3 of KPMG Handbook, Asset acquisitions.

Interpretive response: We believe in most cases the fair value of the unpaid license fees liability will be more reliably measurable than the fair value of the software license acquired. The generally proprietary nature of a vendor's software and the typically unobservable information about the prices at which it sells licenses to the software contribute to this conclusion.

Question 3A.3.40

Should an unpaid software license fees liability be discounted?

Interpretive response: Yes. Ignoring the possibility the effect of discounting may be immaterial, an unpaid license fees liability should be discounted to reflect fair value. [805-50-30-2]

In our experience, the license fees liability will typically be measured at the present value of the amounts yet to be paid under the license agreement using an appropriate discount rate from the perspective of a market participant. [820-10-55-4, 835-30-25-12 – 25-13]

Example 3A.3.20

Initial recognition and measurement of an internal-use software license asset – license fees paid over license term

Customer and Vendor execute a five-year term license on December 1, Year 1 for total license fees of $300,000, payable in five upfront annual installments of $60,000. The contractual license term commences on January 1, Year 2, which coincides with Vendor providing the key necessary for Customer to download the software.

Customer prepays the Year 1 $60,000 license fee and $60,000 for Year 1 post-contract customer support (PCS) on December 31, Year 1.

Customer incurs the following implementation costs related to the software between January 1 and May 31, Year 2. On May 31, Year 2, Customer concludes the software is ready for its intended use.

This example assumes all amounts for the license and services (including PCS) from Vendor reflect their stand-alone prices.

Cost	Amount
Installation, configuration and testing[1]	$40,000
Data conversion/migration[2]	15,000
Training[2]	3,000

Cost	Amount
Software interfacing (Vendor software to other Customer on-premise and hosted applications)[1]	20,000
Total	**$78,000**

Notes:

1. Implementation activities performed by Vendor.
2. Implementation activities performed by Customer or a third-party consultant engaged by Customer.

At December 31, Year 1 Customer has recognized a $60,000 prepaid asset and $60,000 in prepaid PCS.

On January 1, Year 2 Customer recognizes the software license and recognizes an in-process intangible software license asset of $267,906, comprising:

— $207,906 for the unpaid license fees liability (which is also recognized at this date) – present value of the four remaining payments of $60,000 due under the license contract, discounted at 6%; plus

— the $60,000 prepayment made on December 31, Year 1.

Customer capitalizes the $40,000 in installation, configuration and testing costs as part of the cost basis of the software license asset as those costs are incurred between January 1 and May 31.

Customer also capitalizes the $20,000 cost of the software interfaces; those interfaces are capitalized as internal-use software assets separate from the software license asset.

Customer expenses the $15,000 and $3,000 in data conversion/migration and training costs as incurred during the implementation period.

On May 31, Year 2 (when the internal-use software acquired from Vendor is ready for its intended use), Customer begins amortizing the $307,906 ($267,906 + $40,000) software license asset.

The substantial testing of the new interfaces developed to work with Vendor software is completed at the same time as the testing of Vendor software itself. Therefore, Customer also begins amortizing the software interface assets on that date.

Question 3A.3.50

Does the measurement of the software license fees liability include usage-based fees?

Background: In addition to fixed license fees, an entity licensing internal-use software may be required to pay usage- or transaction-based fees. The following are examples.

— An entity licenses software it will use to route customer service tickets to appropriate departments and locations. In addition to a fixed license fee, the entity is also required to pay a fixed fee for each customer service ticket routed using the software.

— An entity licenses research software for use by its employees. In addition to a fixed license fee, the entity pays a usage-based fee for each user that logs into the application each month. If 100 employees log into the application in a given month and the usage-based fee is $100 per user, the entity will owe a usage-based fee of $10,000 for that month.

— An entity licenses accounting software for a fee based on the entity's estimated annual revenues of $100 million. If the entity's annual revenues exceed $100 million, it will owe an incremental fee based on the excess revenues.

These examples are derived from Examples C390.1 – C390.3 in KPMG Handbook, Revenue for software and SaaS, which contain additional detail.

Interpretive response: In general, no. Subtopic 350-40 requires entities to measure the license fees liability on the basis of the asset acquisition guidance in Subtopic 805-50 (see section 3.5 in KPMG Handbook, Asset acquisitions). [350-40-25-17, 350-30-30-1, 805-50-30-2]

Consistent therewith, we believe usage-based fees should generally be excluded from the initial measurement of the license fees liability. Determining when to recognize those fees as an expense involves judgment. In general, we believe the following.

— Usage-based fees that are expected to be recurring – e.g. as the entity uses the software throughout the license period – should be recognized as a cost of the period in which they are incurred.

— Usage-based fees that are not expected to recur – e.g. a one-time milestone payment – should adjust the cost basis of the acquired license asset on a cumulative effect basis.

An exception may arise in more unusual circumstances, such as if the usage-based fees are either (1) payable in the form of the entity's equity instruments or (2) meet the requirements in Topic 815 (derivatives and hedging). Section 3.5 of KPMG Handbook, Asset acquisitions, discusses this further.

Interim reporting considerations

Entities reporting financial information on an interim basis under Topic 270 frequently assign estimated costs and expenses to interim periods so that interim period results more closely reflect anticipated annual results. [270-10-45-4(b)]

In the case of usage-based fees based on an annual benchmark, such as in the third background example, an entity may estimate the usage-based fees it will incur for the entire annual period and accrue a proportion during interim periods. This applies even though the entity will not owe the usage-based fee to the software vendor if the annual benchmark or target is not ultimately met.

Question 3A.3.60

Is the interest on the unpaid license fees liability incurred during application development capitalized?

Background: Interest costs incurred while developing and implementing internal-use software are capitalized under Subtopic 835-20. Therefore, if an entity has outstanding borrowings, the historical cost of internal-use software (i.e. its carrying amount) will include financing costs. [350-40-30-1(c)]

In the case of an internal-use software license that will be paid for over time, the license is directly financed through the software vendor.

Interpretive response: It depends. Like software developed for internal use, interest is capitalized for acquired software licenses that require substantial implementation or customization. This includes interest on any unpaid license fees liability. [835-20-05-01, 15-2 – 15-3, 15-5]

However, an entity is not required to capitalize the interest cost for an acquired license if the customization or implementation activities are not substantial. [835-20-15-3, 15-6]

Consistent with other assets subject to interest capitalization under Subtopic 835-20, interest capitalization (if any) should cease if the implementation or customization project is intentionally delayed or is suspended other than for reasons permitted under Subtopic 835-20. [835-20-25-3 – 25-4, 25-6]

3A.4 Hosting service fees in a CCA

Excerpt from ASC 350-40

15 Scope and Scope Exceptions

General

> Transactions

15-4C Hosting arrangements that do not meet both criteria in paragraph 350-40-15-4A are service contracts and do not constitute a purchase of, or convey a license to, software.

20 Glossary

Hosting Arrangement

In connection with accessing and using software products, an arrangement in which the customer of the software does not currently have possession of the software; rather, the customer accesses and uses the software on an as-needed basis.

A CCA is a service contract. Consequently, the hosting service fees – i.e. the subscription fees paid to the cloud service provider for the right to access the hosted software – are accounted for in the same manner as fees for any other stand-ready service the entity receives (e.g. most equipment maintenance or internet access contracts). [350-40-15-4C]

Section 3A.2 addresses accounting for implementation costs incurred in a CCA, and section 2.7 addresses allocating arrangement consideration between hosting service fees and implementation costs of a CCA.

Observation

Accounting for hosting service fees

In the summary to ASU 2018-15, the FASB expressed the view that accounting for a CCA as a service contract generally means the hosting service fees should be expensed as incurred. [ASU 2018-15.Summary]

Question 3A.4.10

Should hosting service fees in a CCA begin to be recognized as expense if the arrangement term begins before go-live?

Background: Completion of implementation activities frequently requires access to the cloud-based solution – i.e. the implementation activities cannot occur before the customer can access the hosted software. For example, a customer cannot begin to migrate data from its existing system before it has access to the cloud-based solution. Access is also typically required to begin to implement and test interfaces. Therefore, go-live frequently occurs after the contractual CCA term commences and the customer has been provided access to the cloud-based solution.

If the CCA term begins before completion of implementation activities integral to going live with the cloud-based solution, the question arises about whether the customer should begin recognizing the costs of the CCA as expense before go-live.

Question 3A.2.180 explains that hosting service fees due under the CCA are not 'implementation costs'. Instead, they are service fees.

Interpretive response: Yes. CCA hosting service fees should begin to be recognized as an expense when the customer (including third-party consultants working on the customer's behalf) obtains access to the cloud-based solution such that it can begin to undertake its implementation activities. The customer begins to consume and receive benefit from the CCA at that time.

For example, the cloud service provider initiates the CCA on January 1, Year 1, such that the customer has access to the hosted software to commence its implementation activities from that date. The customer should begin

recognizing expense attributable to the CCA as of that date, even if it does not plan to go live until January 1, Year 2.

This is consistent with how lessees recognize lease cost when they install leasehold improvements (when they are the accounting owner of the improvements). In general, a lease commences (and lease cost begins to be recognized) when the lessee obtains access to the underlying asset to begin installing leasehold improvements to make the asset (e.g. retail space) ready for its intended use. Section 5.1 of KPMG Handbook, Leases, discusses this in further detail.

In addition, we believe expensing the hosting service fees incurred during the implementation phase of a CCA is also supported by the accounting treatment for analogous website hosting fees incurred by a website developer during the application and infrastructure development stage of a website. Under Subtopic 350-40, website hosting fees are generally expensed over the period the hosting services are provided. [350-40-25-17F]

Cloud service provider performs the implementation activities

The timing of expense recognition for hosting service fees should not differ based on whether it is the customer (or a third party on its behalf) or the cloud service provider (or a third party on its behalf) that undertakes the implementation activities. While the customer may not be given access to the hosted software during the implementation period if the cloud service provider undertakes the activities, the customer is still consuming and receiving benefit from the CCA as the implementation activities specific to its needs (e.g. customer-requested configurations, customer data migration) are performed, just as if it had obtained access to undertake the activities itself.

This is consistent with the view a lease commences (and lease cost begins to be recognized) when lessee-owned leasehold improvements begin to be constructed even if it is the lessor that is constructing them (rather than the lessee or a third party engaged by the lessee). Example 5.1.10 in KPMG Handbook, Leases, illustrates this.

4. Initial recognition and measurement: Website development (pre-ASU 2025-06)

Detailed contents

4.1 How the standard works

This chapter discusses the accounting for website development costs prior to the adoption of ASU 2025-06, which eliminates Subtopic 350-50. Certain guidance from Subtopic 350-50 has been relocated to Subtopic 350-40, specifically paragraphs 350-40-25-17F – 17I. After adopting ASU 2025-06, entities should refer to chapter 3A for guidance on the initial recognition and measurement of website development costs.

Subtopic 350-50 addresses whether website development costs incurred are capitalized or expensed. This determination depends on the activity and the stage of website development. [350-50-05-1]

Subtopic 350-50 relies heavily on the guidance in Subtopic 350-40. As a result, this chapter frequently refers to chapter 3.

Consistent with the accounting for internal-use software development and implementation costs, the following steps apply to determine what website development costs are capitalized.

- **Step 1:** Determine the stage of website development during which the cost is being incurred
- **Step 2:** Determine if specific requirements apply to the activity
- **Step 3:** Determine which costs of the activity qualify for capitalization
- **Step 4:** Determine when to start and stop capitalization of eligible costs

4.2 Step 1: Determine website development stage & Step 2: Determine if specific requirements apply to the activity

Excerpt from ASC 350-50

25 Recognition

General

25-1 The guidance in this Section refers to various website development stages. See Section 350-50-55 for details regarding the types of costs and activities incurred during those stages.

55 Implementation Guidance and Illustrations

General

> Implementation Guidance

55-1 The following guidance describes or provides examples of various activities that take place at different stages of website development. See Section 350-50-25 for the relevant accounting guidance.

Note: The above excerpts have been superseded by ASU 2025-06 and therefore do not apply once this ASU is adopted.

Subtopic 350-50 describes website development activities as occurring in five website development stages. [350-50-25-2 – 25-17, 55-2 – 55-9]

- Planning
- Website application and infrastructure development
- Graphics development
- Content development
- Operating

The first step in determining whether the costs of a website development activity should be capitalized is to identify the stage to which the activity relates. This is because Subtopic 350-50 requires costs incurred for activities undertaken during the planning and operating stages to be expensed as incurred, while requiring most costs incurred for activities during the other three stages to be capitalized or expensed consistent with the internal-use software guidance in Subtopic 350-40 (see section 3.2). The sub-sections that follow outline the exceptions to this general approach. [350-50-25-2 – 25-17]

Costs related to software to operate the website are accounted for under Subtopic 350-40 unless a plan exists or is being developed to market the software externally. In that case, the software is subject to Subtopic 985-20 (see chapter 5). [350-50-25-4]

Observation

Subtopic 350-50 assumes website development is for internal use

Subtopic 350-50 comes from EITF Issue No. 00-2, Accounting for Web Site Development Costs (EITF 00-2). Exhibit 00-2A states that the guidance in EITF 00-2 was written assuming that any software developed for the website is for internal use.

Therefore, other than paragraph 350-50-25-4 (originally, paragraph 5 of EITF 00-2), Subtopic 985-20 is not referenced in Subtopic 350-50.

The guidance from Subtopic 350-50, including the recognition requirements during each development stage and examples of activities that occur during the different stages of website development, is detailed in the sub-sections that follow. [350-50-25-1, 55-1]

Unless otherwise noted, the sub-sections below assume any software developed or acquired as part of a website development is for internal use and not for R&D with (1) no alternative future use, or (2) internally developed and representing a pilot project or software being used in a specific R&D project. [350-50-25-4, 25-6]

4.2.10 Planning stage

Excerpt from ASC 350-50

25 Recognition

General

> Costs Incurred in the Planning Stage

25-2 Regardless of whether the website planning activities specifically relate to software, all costs incurred in the planning stage shall be expensed as incurred

55 Implementation Guidance and Illustrations

General

> Planning Stage

55-2 Planning stage activities include the following:

a. Develop a business, project plan, or both. This may include identification of specific goals for the website (for example, to provide information, supplant manual processes, conduct e-commerce, and so forth), a competitive analysis, identification of the target audience, creation of time and cost budgets, and estimates of the risks and benefits.

b. Determine the functionalities (for example, order placement, order and shipment tracking, search engine, email, chat rooms, and so forth) of the website.
c. Identify necessary hardware (for example, the server) and web applications. Web applications are the software needed for the website's functionalities. Examples of web applications are search engines, interfaces with inventory or other back-end systems, as well as systems for registration and authentication of users, commerce, content management, usage analysis, and so forth.
d. Determine that the technology necessary to achieve the desired functionalities exists. Factors might include, for example, target audience numbers, user traffic patterns, response time expectations, and security requirements.
e. Explore alternatives for achieving functionalities (for example, internal versus external resources, custom-developed versus licensed software, company-owned versus third-party-hosted applications and servers).
f. Conceptually formulate and/or identify graphics and content (see paragraphs 350-50-25-8 through 25-13).
g. Invite vendors to demonstrate how their web applications, hardware, or service will help achieve the website's functionalities.
h. Select external vendors or consultants.
i. Identify internal resources for work on the website design and development.
j. Identify software tools and packages required for development purposes.
k. Address legal considerations such as privacy, copyright, trademark, and compliance.

Note: The above excerpts have been superseded by ASU 2025-06 and therefore do not apply once this ASU is adopted.

The above are examples of activities that occur during the planning stage of website development and are not exhaustive. Regardless of whether the activity is included above, or relates to software, costs incurred for website development activities during the planning stage are expensed as incurred. [350-50-25-2, 55-2]

4.2.20 Website application and infrastructure development stage

Excerpt from ASC 350-50

25 Recognition

General

> Costs Incurred in the Website Application and Infrastructure Development Stage

25-3 The discussion of website application and infrastructure development assumes that any software is developed for the entity's internal needs and no plan exists or is being developed to market the software externally.

25-4 All costs relating to software used to operate a website shall be accounted for under Subtopic 350-40 unless a plan exists or is being developed to market the software externally. Software for which a plan exists or is being developed to market the software externally is subject to Subtopic 985-20, and costs associated with the development of that software shall be expensed until technological feasibility is established. See paragraph 985-20-25-2.

25-5 Fees incurred for website hosting, which involve the payment of a specified, periodic fee to an Internet service provider in return for hosting the website on its server(s) connected to the Internet, generally are expensed over the period of benefit.

25-6 Costs incurred to purchase software tools, or costs incurred during the application development stage for internally developed tools, shall be capitalized unless they are used in research and development and meet either of the following conditions:

a. They do not have any alternative future uses.
b. They are internally developed and represent a pilot project or are being used in a specific research and development project (see paragraph 350-40-15-7).

25-7 Costs to obtain and register an internet domain shall be capitalized under Section 350-30-25.

55 Implementation Guidance and Illustrations

General

> Application and Infrastructure Development Stage

55-3 The website application and infrastructure development stage involves acquiring or developing hardware and software to operate the website. The activities in this stage include the following:

a. Acquire or develop the software tools required for the development work (for example, HTML editor, software to convert existing data to HTML form, graphics software, multimedia software, and so forth).
b. Obtain and register an Internet domain name.
c. Acquire or develop software necessary for general website operations, including server operating system software, Internet server software, web browser software, and Internet protocol software.
d. Develop or acquire and customize code for web applications (for example, catalog software, search engines, order processing systems, sales tax calculation software, payment systems, shipment tracking applications or interfaces, email software, and related security features).
e. Develop or acquire and customize database software and software to integrate distributed applications (for example, corporate databases and accounting systems) into web applications.
f. Develop HTML web pages or develop templates and write code to automatically create HTML pages.
g. Purchase the web and application server(s), Internet connection (bandwidth), routers, staging servers (where preliminary changes to the website are made in a test environment), and production servers (accessible to customers using the website). Alternatively, these services may be provided by a third party via a hosting arrangement.

h. Install developed applications on the web server(s).
i. Create initial hypertext links to other websites or to destinations within the website. Depending on the site, links may be extensive or minimal.
j. Test the website applications (for example, stress testing).

Note: Paragraphs 350-50-25-5 and 25-7 have been moved to Subtopic 350-40 by ASU 2025-06. The remaining excerpts above have been superseded by ASU 2025-06 and therefore do not apply once this ASU is adopted.

The following table summarizes the accounting for example activities that occur during the website application and infrastructure development stage.

Website application and development stage activity	Subtopic 350-50 accounting
Acquire or develop software for the development work – e.g. HTML editor, software to convert existing data to HTML, graphics software, multimedia software	Apply Subtopic 350-40 in accounting for these activities. [350-50-25-4] See chapter 3.
Acquire or develop software necessary for general website operations – e.g. server operating system software, internet server software, web browser software, internet protocol software	
Develop or acquire and customize code for web applications – e.g. catalog software, search engines, order processing systems, sales tax calculation software, payment systems, shipment tracking applications or interfaces, email software and related security features	
Develop or acquire and customize database software and/or software to integrate distributed applications (e.g. corporate databases and accounting systems) into web applications	
Develop HTML web pages or develop templates and write code to automatically create HTML pages	
Install developed applications	
Create initial hypertext links to other websites or to destinations within the website	
Test website applications – e.g. stress testing	
Purchase web, application, staging and/or production servers, routers, and internet connection Alternatively, purchase services from a third party to acquire the same	The purchase of servers, routers and other property, plant or equipment is outside the scope of Subtopic 350-50. Apply other Topics – e.g. Topic 360 (property, plant and equipment).

Website application and development stage activity	Subtopic 350-50 accounting
capabilities (e.g. through a hosting arrangement)	Fees for third-party website hosting services are generally expensed as the services are provided. [350-50-25-5] Implementation costs incurred for hosting arrangements are accounted for under Subtopic 350-40 (see sections 3.2.10 and 3.2.30). Any other services obtained (e.g. high-speed internet access) are accounted for consistently with any other similar services – i.e. the fees for such services are generally expensed as incurred.
Obtain and register an internet domain name	Recognize as an intangible asset. [350-50-25-7]

4.2.30 Graphics development stage

Excerpt from ASC 350-50

25 Recognition

General

> Costs Incurred in the Graphics Development Stage

25-8 Graphics are a component of software. The costs of developing initial graphics shall be accounted for under Subtopic 350-40 for internal-use software, and Subtopic 985-20 for software marketed externally.

25-9 Modifications to graphics after a website is launched shall be evaluated to determine whether the modifications represent maintenance or enhancements of the website.

55 Implementation Guidance and Illustrations

General

> Graphics Development Stage

55-4 For purposes of this Subtopic, graphics involve the overall design of the web page (use of borders, background and text colors, fonts, frames, buttons, and so forth) that affect the look and feel of the web page and generally remain consistent regardless of changes made to the content.

55-5 Graphics include the design or layout of each page (that is, the graphical user interface), color, images, and the overall look and feel and usability of the website. Creation of graphics may involve coding of software, either directly or through the use of graphic software tools. The amount of coding depends on the complexity of the graphics.

Note: Paragraphs 350-50-25-8 and 55-4 have been moved to Subtopic 350-40 by ASU 2025-06. The remaining excerpts above have been superseded by ASU 2025-06 and therefore do not apply once this ASU is adopted.

The following table summarizes the accounting for initial and subsequent graphics development, including modifications to graphics.

Graphics development stage activity	Subtopic 350-50 accounting
Create initial graphics for the website	Graphics are a component of software. Therefore, costs to create initial graphics are accounted for under Subtopic 350-40. [350-50-25-8] See chapter 3.
Modify website graphics	See section 4.2.50.

4.2.40 Content development stage

Excerpt from ASC 350-50

25 Recognition

General

> Costs Incurred in the Content Development Stage

25-10 Accounting for website content involves issues that also apply to other forms of content or information that are not unique to websites.

25-11 Costs to input content into a website shall be expensed as incurred.

25-12 Software used to integrate a database with a website shall be capitalized under paragraphs 350-40-25-2 through 25-4.

25-13 Data conversion costs shall be expensed as incurred (see paragraph 350-40-25-5).

55 Implementation Guidance and Illustrations

General

> Content Development Stage

55-6 Content refers to information included on the website, which may be textual or graphical in nature (although the specific graphics described in paragraph 350-50-55-4 are excluded from content). For example, articles, product photos, maps, and stock quotes and charts are all forms of content. Content may reside in separate databases that are integrated into (or accessed from) the web page with software, or it may be coded directly into the web pages.

55-7 Content may be created or acquired to populate databases or web pages. Content may be acquired from unrelated parties or may be internally developed.

55-8 Content is text or graphical information (exclusive of graphics described in paragraphs 350-50-55-4 through 55-5) on the website which may include information on the entity, products offered, information sources that the user subscribes to, and so forth. Content may originate from databases that must

be converted to HTML pages or databases that are linked to HTML pages through integration software. Content also may be coded directly into web pages.

Note: Paragraphs 350-50-25-10 – 25-11 and paragraph 350-50-55-6 have been moved to Subtopic 350-40 by ASU 2025-06. The remaining excerpts above have been superseded by ASU 2025-06 and therefore do not apply once this ASU is adopted.

The following table summarizes the accounting for example activities that occur during the content development stage.

Content development stage activity	Subtopic 350-50 accounting
Create content	Subtopic 350-50 does not specify the accounting for content creation, instead noting that content creation and the related accounting issues are not unique to website development. [350-50-25-10] See Question 2.2.10.
Input content (see paragraph 350-50-55-8) into a website, or convert content	Expense costs as incurred. Such costs are like data conversion or migration costs (see section 3.2.20). [350-50-25-11, EITF 00-2.Exhibit 00-2A]
Acquire or develop software to integrate a database with the website	Software to input, migrate or convert content is internal-use software. Costs to acquire or develop such software follow the guidance in Subtopic 350-40 on software to access or convert existing data (see section 3.2.20). [350-40-25-3, 350-50-25-12]

4.2.50 Operating stage

Excerpt from ASC 350-50

25 Recognition

General

> Costs Incurred in the Operating Stage

25-14 Costs of operating a website shall not be accounted for differently from the costs of other operations; that is, those costs shall be expensed as incurred.

25-15 Costs incurred in the operation stage that involve providing additional functions or features to the website shall be accounted for as, in effect, new software. That is, costs of upgrades and enhancements that add functionality shall be expensed or capitalized based on the general model of paragraph 350-40-25-7 (which requires certain costs relating to upgrades and enhancements to be capitalized if it is probable that they will result in added functionality) or, for software that is marketed, paragraphs 985-20-25-3 through 25-4 (which

apply a software capitalization model to product enhancements, which include improvements that extend the life or significantly improve the marketability of a product).

25-16 The determination of whether a change to website software results in an upgrade or enhancement (if internal-use software), or a product enhancement (if externally marketed software), is a matter of judgment based on the specific facts and circumstances. Paragraph 350-40-25-10 states that entities that cannot separate internal costs on a reasonably cost-effective basis between maintenance and relatively minor upgrades and enhancements shall expense such costs as incurred.

25-17 Costs to register the website with Internet search engines represent advertising costs and shall be expensed as incurred under paragraph 720-35-25-1.

55 Implementation Guidance and Illustrations

General

> Operating Stage

55-9 Costs incurred during the operating stage include training, administration, maintenance, and other costs to operate an existing website. Activities in the operating stage include the following:

a. Train employees involved in support of the website.
b. Register the website with Internet search engines.
c. Perform user administration activities.
d. Update site graphics (for updates of graphics related to major enhancements, see [h]).
e. Perform regular backups.
f. Create new links.
g. Verify that links are functioning properly and update existing links (that is, link management or maintenance).
h. Add additional functionalities or features.
i. Perform routine security reviews of the website and, if applicable, of the third-party host.
j. Perform usage analysis.

Note: The above excerpts have been superseded by ASU 2025-06 and therefore do not apply once this ASU is adopted.

The above are examples of activities that occur during the website operating stage and are not exhaustive. Website operating stage activities, including all the examples above other than (d) and (g) (see below), are expensed as incurred, consistent with other operating activities of an entity. [350-50-25-14, 25-17, 55-9]

Entities capitalize or expense the costs of graphics and other changes to the features and functionalities of a website based on the guidance in Subtopic 350-40 (see section 3.2.50). [350-50-25-15 – 25-16]

— First, the entity determines whether the changes are upgrades or enhancements. [350-40-25-7]
— Then, the entity accounts for the costs of the changes based on that determination. [350-40-25-8 – 25-11]

4.3 Step 3: Determine which costs of the activity qualify for capitalization

Subtopic 350-50 does not contain its own guidance like that in Section 350-40-30 about what activity costs qualify for capitalization. There are no initial or subsequent measurement sections (350-50-30 or 350-50-35) in Subtopic 350-50. Therefore, the following applies.

- Costs capitalizable under Subtopic 350-40 (see sections 4.2.20 – 4.2.50) follow the initial measurement guidance in that Subtopic when deciding which activity costs to capitalize (see section 3.2.30).
- Costs capitalizable under other Topics should follow the applicable initial measurement guidance when deciding which activity costs to capitalize. For example, purchased computer servers capitalized under Topic 360 are initially measured based on the guidance in Section 360-10-30.

4.4 Step 4: When to begin and cease capitalization

For costs capitalizable under Subtopic 350-50 by applying Subtopic 350-40, entities follow the guidance in Subtopic 350-40 on when capitalization begins and ceases. See section 3.2.40.

Other Topics that may apply generally do not contain a capitalization period (or window) during which eligible costs can be capitalized. As outlined in section 4.3, entities should follow the guidance in other Topics for costs in their respective scopes.

5. Initial recognition and measurement: Software to be sold, leased or marketed

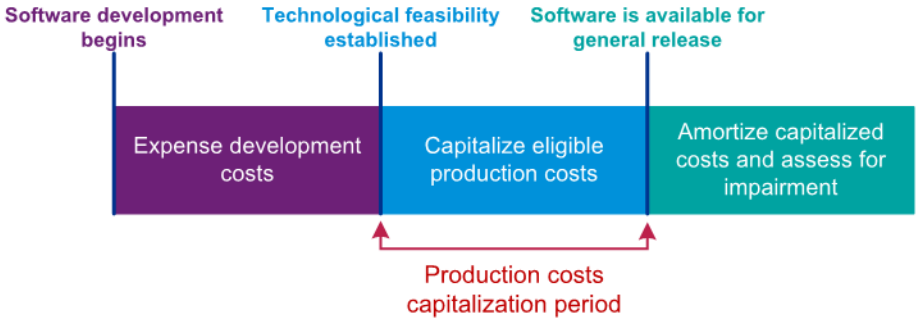
Detailed contents

Item significantly updated in this edition: #

5.1 How the standard works

Subtopic 985-20 addresses the accounting for software development costs incurred for software that will be sold, leased or otherwise marketed (external-use software). [985-20-05-1]

Subtopic 985-20 requires all development costs to establish the technological feasibility of external-use software to be expensed as incurred. Development costs incurred after establishing technological feasibility to produce the 'product master', referred to as 'production costs', are capitalized to the extent recoverable by the NRV of the software product until the product is available for general release. [985-20-25-1, 25-3, 25-6]



The following steps apply to determine what external-use software development costs are capitalized.

- **Step 1:** Determine when technological feasibility is established
- **Step 2:** Determine which costs qualify for capitalization
- **Step 3:** Determine when to cease capitalization of production costs

Costs to develop product enhancements generally follow the same guidance applicable to costs to develop a new software product. [985-20-55-18]

Costs to produce software inventory follow the same guidance as costs to produce other product inventory. [985-20-25-11]

Software maintenance and customer support costs are expensed after general release of the software product. [985-20-25-6]

5.2 Step 1: Determine when technological feasibility is established

5.2.10 Core requirements

Excerpt from ASC 985-20

20 Glossary

Coding

Generating detailed instructions in a computer language to carry out the requirements described in the detail program design. The coding of a computer software product may begin before, concurrent with, or after the completion of the detail program design.

Detail Program Design

The detail design of a computer software product that takes product function, feature, and technical requirements to their most detailed, logical form and is ready for coding.

Product Design

A logical representation of all product functions in sufficient detail to serve as product specifications.

Testing

Performing the steps necessary to determine whether the coded computer software product meets function, feature, and technical performance requirements set forth in the product design.

Working Model

An operative version of the computer software product that is completed in the same software language as the product to be ultimately marketed, performs all the major functions planned for the product, and is ready for initial customer testing (usually identified as beta testing).

25 Recognition

General

> Research and Development Costs of Computer Software

25-1 All costs incurred to establish the technological feasibility of a computer software product to be sold, leased, or otherwise marketed are research and development costs. Those costs shall be charged to expense when incurred as required by Subtopic 730-10.

25-2 For purposes of this Subtopic, the technological feasibility of a computer software product is established when the entity has completed all planning, designing, **coding**, and **testing** activities that are necessary to establish that the product can be produced to meet its design specifications including functions, features, and technical performance requirements. At a minimum,

the entity shall have performed the activities in either (a) or (b) as evidence that technological feasibility has been established:

a. If the process of creating the computer software product includes a **detail program design**, all of the following:
    1. The **product design** and the detail program design have been completed, and the entity has established that the necessary skills, hardware, and software technology are available to the entity to produce the product.
    2. The completeness of the detail program design and its consistency with the product design have been confirmed by documenting and tracing the detail program design to product specifications.
    3. The detail program design has been reviewed for high-risk development issues (for example, novel, unique, unproven functions and features or technological innovations), and any uncertainties related to identified high-risk development issues have been resolved through coding and testing.

b. If the process of creating the computer software product does not include a detail program design with the features identified in (a), both of the following:
    1. A product design and a **working model** of the software product have been completed.
    2. The completeness of the working model and its consistency with the product design have been confirmed by testing.

55 Implementation Guidance and Illustrations

General

> Implementation Guidance

• > Software Research and Development Costs

• • > Establishing Technological Feasibility

55-4 Paragraph 985-20-25-2 specifies the minimum activities an entity should have performed as evidence that technological feasibility has been established, by either inclusion of a **detail program design** or completion of a **working model**. However, an entity may need to defer capitalization until after meeting the working model criteria in paragraph 985-20-25-2(b), even though technological feasibility had previously been established by meeting the detail program design criteria in paragraph 985-20-25-2(a).

55-5 Paragraph 985-20-25-2(a) specifies three criteria relating to the detail program design to be satisfied before capitalization begins. Entities whose software product process fits the description in that paragraph should look to that paragraph for the applicable technological feasibility criteria. However, if the three criteria in that paragraph are not met until a working model is completed, this Subtopic requires capitalization to begin upon completion of the working model and satisfaction of the other criteria in paragraph 985-20-25-2(b).

55-6 Management shall not require more stringent criteria than specified in paragraph 985-20-25-2 to begin capitalizing software production costs. One of the purposes of this Subtopic is to identify an objective point in the software

product process at which research and development activities end and production activities begin. If management were to modify the Subtopic's criteria or impose additional criteria of its own, this objective would be thwarted.

> Technological Feasibility of the Product as a Whole

55-7 When a product comprises various modules that are not separately saleable, technological feasibility is established for the product as a whole, not on a module-by-module basis. The detail program design or the working model of the entire product (all modules linked together) must be completed before capitalization.

> Working Model

55-8 Some entities in the software industry use the term working model to mean a prototype in which critical parts of the product have been coded or written in pseudocode. This definition of working model does not meet the criteria in paragraph 985-20-25-2(b). This Subtopic defines a working model as having several key characteristics not found in that description of a prototype.

55-9 To meet this Subtopic's criteria, the working model must meet all of the following conditions:

a. It must be operative.
b. It must be in the same language as the product that will be marketed.
c. It must be complete with all the major functions that were planned for the product.
d. It must be ready for initial **customer testing**.

> Issues Arising After Establishing Technological Feasibility

55-10 A high-risk development issue may arise after an entity has established technological feasibility by meeting the criteria in paragraph 985-20-25-2. The previously capitalized costs and the costs to resolve the high-risk development issue should be accounted for as a change in accounting estimate in accordance with paragraph 250-10-45-17. That paragraph states that changes in accounting estimates result from new information. The discovery of a high-risk development issue after the entity's personnel thought technological feasibility was established meets this definition. Any previously capitalized costs for that product, as well as any additional costs incurred to establish technological feasibility, should be charged to expense as research and development until the criteria in paragraph 985-20-25-2 are met.

Determining when a software product reaches technological feasibility under Subtopic 985-20 is a judgmental and critical assessment.

Provided no high-risk development issues remain unresolved, technological feasibility is evidenced upon completion of: [985-20-25-2]

- the ‘product design’; and
- the earlier of:
  - a ‘detail program design’; or
  - a ‘working model’.

Subtopic 985-20 includes criteria that must be met to conclude that a detail program design or a working model are completed (see 'Technological feasibility established by detail program design' and 'Technological feasibility established by a working model' sub-sections below). [985-20-25-2, 55-8 – 55-9]

When a software product comprises multiple modules that cannot be sold (i.e. licensed) separately, technological feasibility is established only for the product as a whole (i.e. the linked modules as a unit). Until the criteria in paragraph 985-20-25-2 are met for the product, technological feasibility cannot be established for any of the linked modules. [985-20-55-7]

Question 5.2.10

Can a software module be a 'software product'?

Excerpt from ASC 985-20

Definition of a Software Product

55-1 A software product is most easily defined by describing its necessary qualities. As a product, it is complete and has exchange value. As software, it is a set of programs that interact with each other. A program is further defined as a series of instructions or statements that cause a computer to do work.

Background: Subtopic 985-20 refers to a software product that 'comprises various modules'. Consequently, the question arises about whether a module of a larger program or application can be a 'software product' [985-20-55-7]

Interpretive response: In some cases, yes. We believe a software module can itself be a software product under Subtopic 985-20 if it can be sold (i.e. licensed) separately.

Subtopic 985-20 states if an application includes multiple modules that cannot be sold (i.e. licensed) separately, technological feasibility is established only for the application. In that case, the software product is the application – i.e. all the modules that must be bundled together. In contrast, while not explicitly stated, we believe a separately saleable module is therefore also a software product under Subtopic 985-20. [985-20-55-7]

In addition, we believe a separately saleable module meets the definition of a software product because it must have a complete set of functionalities and its separate saleability evidences 'exchange value'. [985-20-55-1]

There may be judgment involved in assessing whether a software module that is not yet ready for general release is separately saleable and require input from entity personnel outside of the accounting function. However, in general, we believe if a substantive plan as defined in paragraph 350-40-15-2B exists or is being developed to market the module separately (e.g. to license it as a stand-alone offering), that suggests the module *is* separately saleable. [350-40-15-2A – 15-2B]

Observation

Costs incurred to reach technological feasibility like R&D costs

The FASB concluded that activities performed to establish technological feasibility of external-use software are analogous to R&D activities. During this stage, software development costs should be expensed as they are incurred, consistent with other R&D costs. [FAS 86.BC28]

Specifically, Subtopic 730-10 (research and development) includes the following example R&D activity that the FASB concluded was consistent with the activity normally undertaken during the preliminary project stage: 'Engineering activity required to advance the design of a product to the point that it meets specific functional and economic requirements and is ready for manufacture.' [FAS 86.BC28]

The Subtopic 985-20 model, which requires capitalization of certain costs, is an exception to Subtopic 730-10, which requires immediate expensing of all R&D costs. In reaching its conclusions, the FASB attempted to prescribe an accounting principle that could be applied while mitigating some of the concerns about asset recognition of internally generated R&D costs.

Technological feasibility established by a detail program design

A detail program design establishes technological feasibility when all of the following criteria are met. [985-20-25-2(a)]

Technological feasibility by detail program design
— **Completion** of a detailed program and product design — **All resources** necessary to produce the product are identified and available — **Completeness** of the detailed program design has been confirmed by documenting and tracing to product specifications — All **high-risk development issues**[1], must have been resolved through coding and testing
Note: 1. Subtopic 985-20 gives the following examples of high-risk development issues: novel, unique, unproved functions and features or technological innovations. [985-20-25-2(a)(3)]

Subtopic 985-20 defines a detail program design as "the detail design of a computer software product that takes product function, feature, and technical requirements to their most detailed, logical form and is ready for coding." [985-20 Glossary]

In practice, the form and level of detail of a detail program design varies from entity to entity and can also differ for projects within an entity. However, to meet the definition in Subtopic 985-20, a detail program design would generally comprise a combination of flowcharts, narratives, and outlines that, together, detail elements of the software such as (not exhaustive):

— data flows;
— routines;
— report and field definitions;
— algorithms; and
— interaction with other software programs.

Complex projects may be more likely to involve a detail program design. Complex projects will also generally necessitate developers' involvement during the process to determine (1) the appropriateness of the detailed design documentation and (2) when technological feasibility is established.

In some cases, an entity will not be able to evidence the detail program design criteria are met until a working model is completed. In that case, technological feasibility is not established until the working model is completed. [985-20-55-5]

Question 5.2.20

What are the effects of detail program design changes after technological feasibility has been established?

Background: A detail program design is frequently not final at the point in time it is used to establish technological feasibility of a software product. Changes will frequently occur throughout the entire development process.

Interpretive response: In general, only minor changes to the detail program design after technological feasibility has been established are treated as post-technological feasibility production activities.

However, significant changes to elements of the detail program design may call into question whether it remains sufficiently complete to establish technological feasibility. In that case, similar to when new high-risk development issues emerge after technological feasibility has been established, the entity treats the change in its assessment of the detail program design as a change in accounting estimate under Topic 250 (accounting changes and error corrections). See section 3.4 in KPMG Handbook, Accounting changes and error corrections.

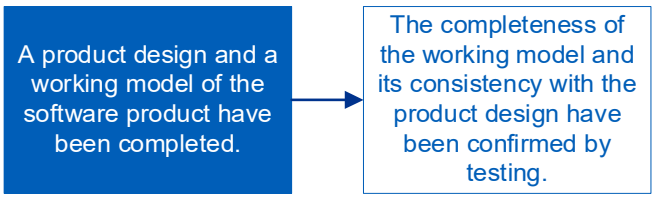
As a result, the entity:

— expenses all previously capitalized development costs as R&D costs of the period in which the conclusion is reached that the changed detail program design is not sufficiently complete; and

— expenses new development costs as incurred until technological feasibility is re-established based on the criteria in paragraph 985-20-25-2.

Technological feasibility established by a working model

Subtopic 985-20 also permits establishing technological feasibility of a software product by completion of a working model when both of the following criteria are also met.



A working model is a version of the software product that meets four conditions. [985-20-55-9]

Working model

- Operative
- Written in the same language as the product that will be marketed
- Complete with all major functions that will be marketed
- Ready for initial customer testing

Entities assess internal definitions of working models or prototypes against the criteria in Subtopic 985-20. Different internal definitions or parameters cannot be substituted. Subtopic 985-20 gives the example of an entity defining a prototype software product with critical parts written in pseudocode (i.e. not written in computing language) as a working model. This prototype software product does not meet the condition in paragraph 985-20-25-2(b)(2) or the conditions in paragraph 985-20-55-9. Therefore, it does not meet the definition of a working model to establish technological feasibility. [985-20-25-2(b), 55-8 – 55-9]

Question 5.2.30

Does pre-releasing software establish its technological feasibility under the working model approach?

Background: Some software entities make their software available to customers for their own review and familiarization (e.g. in a test, or 'sandbox', environment) before releasing it generally. During a pre-release (or early release) stage, customers may be able to provide feedback to the entity that it can still incorporate into the software. The entity usually advises those who may download the software at this stage that it has not been through the full quality assurance (QA) process, is still under development and/or that bugs may still exist.

While terminology varies in the software industry, such that individual entities' use and understanding of certain terms differ, the software released during a pre-release stage is often a post-beta version of the software in terms of its stability and level of remaining software bugs (frequently referred to as a 'release candidate' or 'release version'). However, some software entities will make earlier versions of their software (e.g. alpha, beta or even pre-alpha versions) available on a pre-release basis.

Question 5.4.10 discusses the effects of a pre-release on deciding when capitalization of eligible production costs should cease.

Interpretive response: It depends on the facts and circumstances. This is because the pre-release stage of software development often varies by entity.

In the case of a pre-release that follows beta testing or involves a release version that is past typical beta conditions, the criteria in paragraph 985-20-25-2(b) may be met before pre-release occurs. If those criteria are met before pre-release occurs, it is not appropriate to defer technological feasibility until the pre-release occurs.

However, in cases of early pre-release (e.g. of an alpha or pre-alpha version), the pre-release stage may precede establishing technological feasibility. This early pre-release stage may be intended to assist the entity in determining whether its software is complete and can meet its design specifications. In that case, if the entity has not yet reached this determination, technological feasibility has not been established. [985-20-25-2]

While Subtopic 985-20 states a working model is 'usually' a beta version, an alpha or pre-alpha version might meet the definition of a working model and the conditions in paragraph 985-20-55-9. [985-20 Glossary, 985-20-55-9]

Question 5.2.40

How is the assessment of technological feasibility under the working model approach affected when beta testing does not occur?

Background: An entity may not produce a beta version of its software. This is because software beta testing may be prohibited in some cases – e.g. because of legal or regulatory restrictions. In other cases, entities (e.g. those following an agile development methodology – see section 5.5) will not produce a beta version or undertake beta testing as part of their development process.

Interpretive response: The Subtopic 985-20 definition of working model does not specify a working model is a beta version of the software, or that customer beta testing is required. It instead indicates a working model is 'usually' a version *ready* for beta testing. [985-20 Glossary]

An entity that does not undertake customer beta testing nonetheless has a completed working model when it has completed a software version that meets the 'working model' definition and the four conditions in paragraph 985-20-55-9. [985-20 Glossary, 985-20-55-9]

In some cases, particularly entities following an agile software development methodology, the first completed version of the software that meets these conditions (and the criterion in paragraph 985-20-25-2(b)(2)) will be a post-beta version of the software in terms of its stability and level of remaining software bugs, sometimes referred to as a 'release candidate' (or similar, terms and entities' use thereof are not uniform).

The absence of a beta version milestone may introduce additional judgment into determining when a working model is complete. Further, the absence of the beta testing may make conclusions about the completeness of the working model and the absence of high-risk development issues less certain. Lastly, we have observed that not undertaking beta testing may shorten the period of time

between when a working model is completed and the software is made available for general release; in that case, the capitalization window (see section 5.4) may be short. [985-20-25-2(b)(2)]

Detail program design or working model?

If the entity's development process for the software product includes a detail program design, completion of those activities should generally guide when technological feasibility is established. Regardless, if a working model and its related activities are completed first, technological feasibility has been established. [985-20-25-2(a), 55-5]

If the entity's development process for the software product does not include a detail program design, technological feasibility must be established by completion of a working model. [985-20-25-2(b)]

Question 5.2.50

Is establishing technological feasibility by detail program design or working model an accounting policy election?

Interpretive response: No. An entity's development process and the facts and circumstances of the product's development will dictate which technological feasibility milestone applies. [985-20-25-2]

Question 5.2.60

Does the approach to establishing technological feasibility affect when it is established?

Interpretive response: It depends. This is because entities' applications of each approach vary. However, in our experience, the working model approach typically establishes technological feasibility later in the development cycle than the detail design program approach. Subtopic 985-20 also seems to imply this will frequently be the case. [985-20-25-2, 55-4]

This may especially be the case if the entity's software development process only results in a relatively late stage working model (see Question 5.2.40 and section 5.5).

Observation

Working model approach more common

In our experience, the working model approach is more frequently applied in practice. We believe the growing prevalence of agile software development methodology has contributed to this; see Question 5.5.10.

Alternatives to detail program design or working model

An entity is not permitted to apply more stringent criteria to delay establishing technological feasibility. [985-20-55-6]

Question 5.2.70

Can an entity select a milestone other than a completed detail program design or working model to evidence technological feasibility?

Interpretive response: No. An entity is required to meet either the (1) detail program design or (2) working model criteria to evidence technological feasibility. [985-20-25-2]

High-risk development issues arising after technological feasibility is established

Excerpt from ASC 985-20

55 Implementation Guidance and Illustrations

General

> Implementation Guidance

• > Software Research and Development Costs

• • > Issues Arising After Establishing Technological Feasibility

55-10 A high-risk development issue may arise after an entity has established technological feasibility by meeting the criteria in paragraph 985-20-25-2. The previously capitalized costs and the costs to resolve the high-risk development issue should be accounted for as a change in accounting estimate in accordance with paragraph 250-10-45-17. That paragraph states that changes in accounting estimates result from new information. The discovery of a high-risk development issue after the entity's personnel thought technological feasibility was established meets this definition. Any previously capitalized costs for that product, as well as any additional costs incurred to establish technological feasibility, should be charged to expense as research and development until the criteria in paragraph 985-20-25-2 are met.

Technological feasibility is not established until high-risk development issues are resolved through coding and testing. No production costs are eligible for capitalization before that occurs. [985-20-25-2, 55-10]

Identifying new high-risk development issues after technological feasibility is established should be infrequent. However, if it occurs and is not the result of an error in the previous assessment of technological feasibility, the entity treats

the identification of new issues as a change in accounting estimate under Topic 250 (accounting changes and error corrections). That is, the entity: [985-20-55-10]

— expenses all previously capitalized development costs as R&D costs of the period in which the new issues were identified; and

— expenses new development costs as incurred until technological feasibility is re-established based on the criteria in paragraph 985-20-25-2, which includes curing the new high-risk issues.

5.2.20 Software product enhancements

Excerpt from ASC 985-20

20 Glossary

Product Enhancement

Improvements to an existing product that are intended to extend the life or improve significantly the marketability of the original product. Enhancements normally require a product design and may require a redesign of all or part of the existing product.

55 Implementation Guidance and Illustrations

General

> Implementation Guidance

• > Product Enhancements

• • > Accounting for Costs of Product Enhancements

55-18 Costs incurred for **product enhancements** should be charged to expense as research and development until the technological feasibility of the enhancement has been established. If the original product will no longer be marketed, any unamortized cost of the original product should be included with the cost of the enhancement for purposes of applying the net realizable value test and amortization provisions. If the original product will remain on the market along with the enhancement, the unamortized cost of the original product should be allocated between the original product and the enhancement.

• • > Technological Feasibility

55-20 The technological feasibility criteria in paragraph 985-20-25-2 must be met for a product enhancement if the criteria had been met for the original product.

55-21 Product enhancements are specifically included in the scope of this Subtopic and, as such, are subject to the same requirements as any other software product. However, technological feasibility may be more easily established for a product enhancement than for a new product, and capitalization of costs may, therefore, begin relatively earlier in the software process. For example, an enhancement that adds one function to an already

successful product may require only minor modifications to the original product's detail program design to establish technological feasibility.

55-22 Similarly, in some cases, software that is ported (made available for a different piece of hardware) may not require a new detail program design, and capitalization of the enhancement costs may begin once any high-risk development issues have been resolved.

Software product enhancements extend the life or significantly improve the marketability of the original software product. Enhancements normally require a 'product design' and may require a redesign of all or part of the existing product. [985-20 Glossary]

Technological feasibility is assessed for each enhancement and may be more easily established than for a new software product. This is because a single enhancement may require only: [985-20-55-18, 55-21]

— a minor update to a detail program design for the larger product; or
— a minor change to the existing product's code to get to a working model of the updated product (i.e. including the enhancement).

Porting software – i.e. enabling it to run on different hardware or in a different hardware environment (e.g. a public cloud), or on a different software operating system (e.g. Linux, Windows, macOS) – may not require a new detail program design (if one was used originally). Capitalization can, instead, begin when any high-risk development issues have been resolved. See Example 5.4.10. [985-20-55-22]

5.2.30 Development costs for software that is an integral part of a product or process

Excerpt from ASC 985-20

25 Recognition

General

> Production Costs of Computer Software

25-4 Software production costs for computer software that is to be used as an integral part of a product or process shall not be capitalized until both of the following conditions have been met:

a. Technological feasibility has been established for the software.
b. All research and development activities for the other components of the product or process have been completed. All research and development activities for the other components of the product or process have been completed.

55 Implementation Guidance and Illustrations

General

> Implementation Guidance

• > Interpretation of Scope

• • > Software Marketed as Part of a Product or Process

55-3 The costs of software that is marketed as part of a product or process are included in the scope of this Subtopic. Software is sometimes embedded in a product and sold as part of the product as a whole. Examples are calculators and robots. This type of software is sometimes known as firmware. Also, some services provided to customers would not be possible without software. Time-sharing and service bureaus are two straightforward examples.

Entities developing software that will be integral to a product or process (e.g. firmware) should ensure the following conditions are met before commencing capitalization: [985-20-25-4]

— technological feasibility has been established; and
— the other components of the product or process are no longer in the R&D phase.

Entities may encounter scenarios where they establish the software's technological feasibility before completing all R&D activities for the related product or process. The software development costs continue to be expensed in these scenarios until all product or process-related R&D activities have been completed. [985-20-25-4]

Example 5.2.10
Capitalization of embedded firmware

ABC Corp. develops and markets robots that assist individuals with home care. ABC is developing a new robot vacuum that will embed its proprietary robotic vision and guidance software (firmware).

The following additional facts are relevant (all in Year 1):

— ABC established technological feasibility of the firmware on February 1.
— R&D activities related to the robotic infrastructure were completed June 30.
— The product became available for sale on August 1.

ABC begins capitalization of eligible software production costs after June 30.

Although ABC established technological feasibility of the firmware on February 1, R&D for the other components of the product were not completed until June 30. Therefore, firmware development costs incurred from February 1 to June 30 were expensed as incurred. Capitalization ceases on August 1 when the product is available for general release.

5.2.40 Purchased software

Excerpt from ASC 985-20

25 Recognition

General

> Purchased Computer Software

25-7 Some entities purchase software as an alternative to developing it internally. Purchased computer software may be modified or integrated with another product or process.

25-8 The cost of purchased computer software to be sold, leased, or otherwise marketed that has no alternative future use shall be accounted for the same as the costs incurred to develop such software internally, as specified in paragraphs 985-20-25-1 through 25-6.

25-9 An entity shall capitalize the total cost of purchased software that has no alternative future use if the criteria specified in paragraph 985-20-25-2 are met at the time of purchase. Otherwise, the cost will be charged to expense as research and development. For example, if the technological feasibility of a software product as a whole (that is, the product that will be ultimately marketed) has been established at the time software is purchased, the cost of the purchased software shall be capitalized and further accounted for in accordance with the other provisions of this Subtopic. The cost of software purchased to be integrated with another product or process shall be capitalized only if technological feasibility is established for the software component and if all research and development activities for the other components of the product or process are completed at the time of purchase.

25-10 If purchased software has an alternative future use, the cost shall be capitalized when the software is acquired and accounted for in accordance with its use. The alternative future use test also applies to purchased software that will be integrated with a product or process in which the research and development activities for the other components are not complete.

55 Implementation Guidance and Illustrations

General

> Implementation Guidance

• > Purchased Computer Software

• • > Purchased Software to be Integrated into Another Product

55-13 An entity may purchase software that will be integrated into another software or hardware product. Assuming that purchased computer software has no alternative future use, its costs can be capitalized only if the technological feasibility of the product to be ultimately marketed has been established at the time of purchase. Such factors as the timing of receipt or the status of hardware and internal software development may be crucial in determining whether technological feasibility is established at the time of purchase.

• • > Software Purchased Before Technological Feasibility Established

55-14 An entity may purchase software before technological feasibility has been established. For example, an entity purchases software for $100,000 that can be resold for $75,000. The amount of $25,000 would be charged to research and development, and $75,000 would be capitalized. If the software product reached technological feasibility, the $75,000 would be included in the cost of the software product. If the technological feasibility of the software was never established, the $75,000 would be classified as inventory.

Entities will frequently purchase (acquire or license) software to embed in a larger software product, or another product or process. The cost of the purchased software is expensed at the time of acquisition, as R&D costs of the larger software product or other product/process, if: [985-20-25-9, 55-13]

— the technological feasibility of the software product or other product or process with which it will be integrated has not yet been established; and
— it has no alternative future use – e.g. resale, licensing to other entities, internal use.

If the first criterion is met, the entire cost of the purchased software is capitalized as a production cost if the larger software product or other product/process is not yet available for general release. [985-20-55-13]

If the first criterion is *not* met at the time of purchase, but the purchased software has an alternative future use, the cost of the software is capitalized to the extent that cost is realizable from the alternative use, and the software asset is classified based on the nature of that alternative use (e.g. as inventory, intangible asset). If the larger software product or other product/process later achieves technological feasibility, the purchased software asset is reclassified as part of the cost of the larger software product asset. [985-20-55-14]

Section 5.3.30 further discusses the accounting for the costs of purchased software.

Question 5.2.80

How is external-use software purchased outside of a business combination accounted for if technological feasibility has not been established?

Background: An entity may purchase (acquire or license) software for which technological feasibility has not been established at the time of purchase. The entity may, for example, intend to complete its development and then either:

— market it to customers as a stand-alone product;
— embed it into another software product to be marketed to customers; or
— integrate it into a non-software product or process.

Interpretive response: We believe such software is in-process R&D (IPR&D). Consistent with Question 4.2.20 in KPMG Handbook, Asset acquisitions, the cost (which may be an allocated amount) of the software (license) is expensed upon acquisition if it does not have an alternative future use.

Question 5.2.90

Is it necessary to have a product design or a detail program design to establish technological feasibility of purchased software?

Interpretive response: Not if the entity acquires software that is already a working model that meets the requirements in paragraph 985-20-55-9. In that case, we believe the working model establishes technological feasibility without a product design or a detail program design.

Question 5.2.100

Do seller restrictions on the purchased software affect whether it has an alternative future use?

Interpretive response: Yes. We believe alternative future uses include only those the entity is permitted to exploit. For example, an entity may be prohibited from embedding purchased software intended to be used as firmware in another product, or from reselling or licensing the software to third parties, under the terms and conditions of its contract with the software seller. In that case, such uses would not be viable alternatives for the future use of the software when deciding whether and how much of the cost of the purchased software to capitalize.

5.3 Step 2: Determine which costs qualify for capitalization

Question 5.3.10

What is the unit of account for external-use software development costs?

Interpretive response: Initially, each individual software development or implementation cost is its own unit of account. Therefore, *each* cost incurred is capitalized or expensed based on whether: [985-20-25-1, 25-3, 25-6]

- the software product to which it relates has reached technological feasibility (see section 5.2), but is not yet available for general release (see section 5.4); and
- the cost is an eligible cost (see section 5.3).

After initial recognition, capitalized software costs are amortized and assessed for impairment product-by-product. Chapter 6 addresses capitalized software cost amortization and impairment. [985-20-35-1, 35-4]

5.3.10 Production costs

Excerpt from ASC 985-20

20 Glossary

Coding

Generating detailed instructions in a computer language to carry out the requirements described in the detail program design. The coding of a computer software product may begin before, concurrent with, or after the completion of the detail program design.

Testing

Performing the steps necessary to determine whether the coded computer software product meets function, feature, and technical performance requirements set forth in the product design.

25 Recognition

General

> Production Costs of Computer Software

25-3 Costs of producing product masters incurred subsequent to establishing technological feasibility shall be capitalized. Those costs include coding and testing performed subsequent to establishing technological feasibility.

25-5 An entity may capitalize an allocated amount of indirect costs, such as overhead related to programmers and the facilities they occupy. However, an allocation of general and administrative expenses is not appropriate because those costs relate to the period in which they are incurred.

Excerpt from ASC Master Glossary

Product Master

A completed version, ready for copying, of the computer software product, the documentation, and the training materials that are to be sold, leased, or otherwise marketed.

The period during which development costs of a software product or product enhancement are capitalized (i.e. the capitalization window) extends from: [985-20-25-3, 25-6]

— when technological feasibility is established (see section 5.2); until
— the software product (or product enhancement) is available for general release (see section 5.4).

During the capitalization window, development costs (e.g. coding and testing) to produce the 'product master' are capitalized. These development costs are referred to as 'production costs'. This applies equally to new software products and product enhancements (see section 5.3.50). [985-20-25-3, 55-18]

Costs to manufacture the software (e.g. reproduce product masters and related documentation or training materials) are outside the scope of Subtopic 985-20. See section 5.3.20. [985-20-25-11]

Question 5.3.20

Can an entity elect not to capitalize software production costs under Subtopic 985-20?

Background: Many software entities, particularly those that develop software using an agile software development methodology, do not capitalize software production costs under Subtopic 985-20.

Interpretive response: No. Capitalization of software production costs during the capitalization window is not an accounting policy election. While many entities do not capitalize software production costs, this is not the result of an accounting policy election. [985-20-25-3]

Instead, many entities validly conclude, based on their software development process and other entity- or product-specific facts and circumstances, the capitalization window is short, therefore the costs incurred during that window are clearly inconsequential.

Eligible production costs include both direct and indirect costs. Costs that are not directly or indirectly attributable to a specific software product, such as general and administrative expenses, are not capitalized. [985-20-25-5]

Observation

Overhead costs: Subtopic 985-20 vs Subtopic 350-40

Indirect (or overhead) costs (see Question 5.3.30) capitalized under Subtopic 985-20 are not capitalized under Subtopic 350-40. [985-20-25-5, 350-40-30-3]

Question 5.3.30#

What are example direct and indirect software production costs?

Background: Subtopic 985-20 does not provide examples of direct or indirect production costs that would qualify for capitalization.

Interpretive response: We believe the following are examples of direct and indirect production activity (e.g. coding and testing) costs that would generally qualify for capitalization if incurred during the applicable capitalization window.

Direct costs	Indirect costs
— Payroll and payroll-related costs (e.g. employment taxes, benefits) of employees directly involved with production activities, to the extent of time spent thereon — Travel expenses incurred by employees in performing production activities — Fees paid to third parties for software production services — Interest costs on borrowings that fund the software's production — Costs of software and/or data acquired or licensed specifically for the purpose of developing the software product	— Facility charges for facilities used by employees involved in the production activities (e.g. software developers and engineers) — Hardware or third-party hosting costs — Costs of software and/or data acquired or licensed for multiple uses, one of which is to develop the software product

Interest cost capitalization is governed by the guidance in Subtopic 835-20. This includes that entities should cease capitalizing interest if development activities are suspended for reasons other than those permitted under that Subtopic. [835-20-25-4]

Entities need to review overhead costs to ensure appropriate amounts are capitalized after achieving technological feasibility. Judgments and estimates might be necessary when indirect costs are not tracked or allocated across projects in a systematic fashion.

See section 2.8.30 for specific considerations around data costs incurred in developing external-use AI software.

Example 5.3.10
Capitalization of software development costs

ABC Corp. developed external-use Software Product X using internal and external resources.

ABC incurred the following costs after achieving technological feasibility:

Costs related to Product X	Amount
External consultants that assisted with testing	$20,000
Allocation of internal payroll and payroll-related costs of software developers coding and testing	$40,000
Allocation of rent and facilities costs for facilities used by developers and engineers	$ 5,000
Allocation of third-party hosting space	$ 5,000
Promotional marketing costs before general release	$30,000

ABC capitalizes the first four items, totaling $70,000, as eligible production costs of Product X. ABC expenses the $30,000 in marketing costs because they are not production costs.

Question 5.3.40

How does an entity account for outsourced software product R&D payments?

Background: Entities may outsource their new software (i.e. products or enhancements) R&D to a third party. Assume that an entity has done so and has agreed to fund the R&D for a new software product by making multiple, nonrefundable payments to a third-party developer over the software development life cycle.

- Nonrefundable payments are paid to the third party at (1) inception of the arrangement, (2) upon delivery of a working model and (3) upon delivery of the product master.
- No further payments are owed to the third party after the first payment unless and until the achievement of each subsequent milestone occurs.
- Technological feasibility is established by completion of a working model and meeting the criteria in paragraph 985-20-25-2(b) given the planned development process for the software product (see Question 5.2.50).

In scenarios like this, the question arises about how to account for the milestone payments.

Interpretive response: Nonrefundable payments for R&D services, including those to develop external-use software, that have no alternative future use are deferred and recognized as expense as the services are provided. [730-20-25-13 – 25-14, 35-1]

Milestone software development payments that are nonrefundable even if technological feasibility is never achieved (such as the first milestone payment in the background) are payments for R&D services. They are deferred when paid and expensed over the period the R&D services are provided – e.g. from services commencement until the new software product or enhancement's technological feasibility is established.

Payments not owed unless future milestones are met (e.g. the second and third payments in the background) are not accrued in advance of being incurred.

When future milestones are met, the related payments are capitalized if the payments are for software production activities undertaken by the third party developer during the capitalization window for the software product or enhancement.

Example 5.3.20
Accounting for outsourced software development payments

ABC Corp. develops and markets large enterprise resource planning (ERP) software systems. ABC outsourced the R&D for a new customer relationship management (CRM) software product to unrelated third party XYZ Company.

Under the contract with ABC, XYZ will receive the following nonrefundable milestone payments.

Milestone	Amount
Inception of the arrangement	$1,500,000
Delivery of a functioning beta version1	1,000,000
Delivery of completed product master	500,000
Total	**$3,000,000**

Note:

1. In this case, a functioning beta version would meet the criteria in paragraph 985-20-25-2(b) to establish the technological feasibility of the software.

The initial, nonrefundable milestone payment of $1,500,000 is deferred and expensed over the R&D service period until the functioning beta version of the new CRM software product is delivered to ABC (note: this assumes the instructions to XYZ are not sufficiently detailed to constitute a detail program design). Because this payment is nonrefundable even if a functioning beta version is not delivered (i.e. technological feasibility is not established), the entirety of that payment is for R&D services.

At inception, and until the functioning beta version is delivered, ABC does not accrue for either of the remaining two milestone payments. ABC also does not accrue for the third milestone payment when XYZ delivers the functioning beta version – i.e. ABC accrues for that payment only upon completion of the product master.

The final two milestone payments are due only after technological feasibility of the CRM product is established. Therefore, the payments are capitalized under Subtopic 985-20 to the extent they are for eligible production costs – i.e. costs of development activities incurred after technological feasibility is established.

5.3.20 Inventory costs

Excerpt from ASC 985-20

25 Recognition

General

> Inventory Costs

25-11 For guidance on costs incurred for duplicating the computer software, documentation, and training materials from product masters and for physically packaging the product for distribution, see Subtopic 985-330.

Excerpt from ASC 985-330

20 Glossary

Product Master

A completed version, ready for copying, of the computer software product, the documentation, and the training materials that are to be sold, leased, or otherwise marketed.

25 Recognition

General

25-1 The costs incurred for duplicating the computer software, documentation, and training materials from the **product masters** and for physically packaging the product for distribution shall be capitalized as inventory on a unit-specific basis.

Subtopics 985-20 and 985-330 differentiate software production costs (e.g. coding and testing) necessary to produce the product master from costs to create a physical product (e.g. boxed software). Costs incurred after producing the product master to duplicate and package the software product and its related documentation/training materials are not production costs and are outside the scope of Subtopic 985-20. [985-20-25-3, 25-11]

Software inventory costs are recognized consistent with any other type of inventory item. [985-330-25-1]

5.3.30 Purchased software

Excerpt from ASC 985-20

25 Recognition

General

> Purchased Computer Software

25-7 Some entities purchase software as an alternative to developing it internally. Purchased computer software may be modified or integrated with another product or process.

25-8 The cost of purchased computer software to be sold, leased, or otherwise marketed that has no alternative future use shall be accounted for the same as the costs incurred to develop such software internally, as specified in paragraphs 985-20-25-1 through 25-6.

25-9 An entity shall capitalize the total cost of purchased software that has no alternative future use if the criteria specified in paragraph 985-20-25-2 are met at the time of purchase. Otherwise, the cost will be charged to expense as research and development. For example, if the technological feasibility of a software product as a whole (that is, the product that will be ultimately marketed) has been established at the time software is purchased, the cost of the purchased software shall be capitalized and further accounted for in accordance with the other provisions of this Subtopic. The cost of software purchased to be integrated with another product or process shall be capitalized only if technological feasibility is established for the software component and if all research and development activities for the other components of the product or process are completed at the time of purchase.

25-10 If purchased software has an alternative future use, the cost shall be capitalized when the software is acquired and accounted for in accordance with its use. The alternative future use test also applies to purchased software that will be integrated with a product or process in which the research and development activities for the other components are not complete.

55 Implementation Guidance and Illustrations

General

> Implementation Guidance

• > Purchased Computer Software

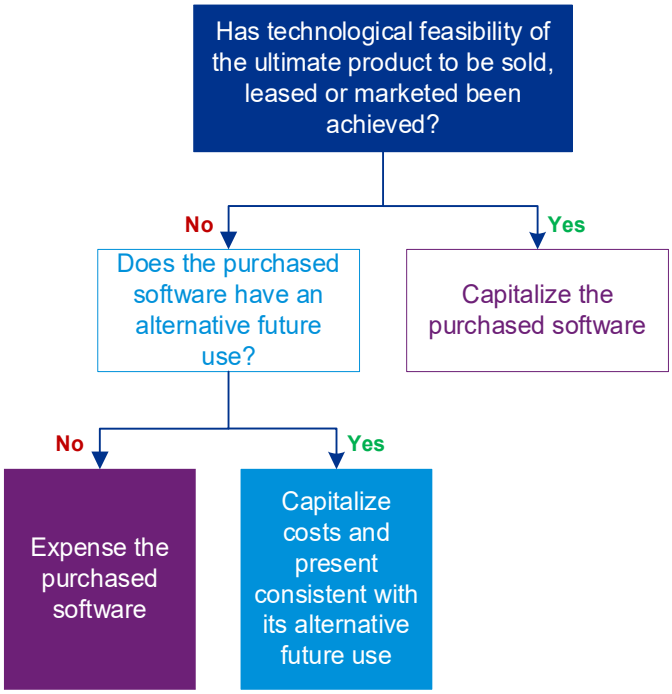
• • > Purchased Software to be Integrated into Another Product

55-13 An entity may purchase software that will be integrated into another software or hardware product. Assuming that purchased computer software has no alternative future use, its costs can be capitalized only if the technological feasibility of the product to be ultimately marketed has been established at the time of purchase. Such factors as the timing of receipt or the status of hardware and internal software development may be crucial in determining whether technological feasibility is established at the time of purchase.

• • > Software Purchased Before Technological Feasibility Established

55-14 An entity may purchase software before technological feasibility has been established. For example, an entity purchases software for $100,000 that can be resold for $75,000. The amount of $25,000 would be charged to research and development, and $75,000 would be capitalized. If the software product reached technological feasibility, the $75,000 would be included in the cost of the software product. If the technological feasibility of the software was never established, the $75,000 would be classified as inventory.

The following diagram illustrates the accounting for purchased software.



Example 5.3.30

Capitalization of purchased software with an alternative use

ABC Corp. sells a large system tool to the semi-conductor industry. The firmware for the tool includes embedded third-party process automation software. ABC purchased the process automation software outright from the third-party software vendor, who was exiting that line of business.

ABC acquired the software before establishing technological feasibility of the firmware and completing R&D activities for the entire system tool.

The cost of the purchased software was $200,000. At the time of acquisition, ABC concluded the purchased software had alternative future uses to it; the software could be licensed in its present form to other entities with an NRV of $190,000 or re-sold outright for $125,000.

Scenario 1: ABC achieves technological feasibility of the firmware and completes other R&D activities for the tool after acquiring the third-party software

ABC capitalizes the $190,000 and expenses $10,000 when the software is acquired. Upon establishing technological feasibility of the firmware and completing the other necessary R&D activities for the tool, the $190,000 is accounted for as part of the cost basis of the firmware asset.

Scenario 2: ABC does not achieve technological feasibility of the firmware and complete R&D activities for the tool

Consistent with Scenario 1, ABC capitalizes the $190,000 and expenses $10,000 when the software license is obtained.

Because the R&D activities for the tool are not successful, the purchased software that will now be licensed to other entities remains a separately recognized asset, subject to the subsequent measurement guidance in Subtopic 985-20 (see section 6.4).

5.3.40 Maintenance and customer support

Excerpt from ASC 985-20

20 Glossary

Customer Support

Services performed by an entity to assist customers in their use of software products. Those services include any installation assistance, training classes, telephone question and answer services, newsletters, on-site visits, and software or data modifications.

Maintenance

Activities undertaken after the product is available for general release to customers to correct errors or keep the product updated with current information. Those activities include routine changes and additions.

25 Recognition

General

> Production Costs of Computer Software

25-6 Capitalization of computer software costs shall cease when the product is available for general release to customers. Costs of **maintenance** and **customer support** shall be charged to expense when related revenue is recognized or when those costs are incurred, whichever occurs first.

55 Implementation Guidance and Illustrations

General

> Implementation Guidance

• > Maintenance and Customer Support

55-11 When selling systems software, an entity may promise to keep the software current with revisions in the hardware, and incur costs in connection with this service.

55-12 This activity appears to meet the definition of **maintenance** because it keeps the product updated with current information. The cost of maintenance is charged to expense when related revenue is recognized or when those costs are incurred, whichever occurs first. The distinctions among maintenance, **customer support**, and product enhancements are sometimes very fine lines; in each case, the particular circumstances and intentions of the entity should be evaluated in light of the definitions in this Subtopic for each activity.

Maintenance and customer support costs are not production costs. They are generally expensed as incurred, regardless of when incurred – e.g. before or after the software is available for general release. [985-20-25-6, 55-12]

Observation

Maintenance and customer support revenue recognition when FAS 86 was issued

FASB Statement (FAS) No. 86 was issued in 1985. The guidance in paragraph 985-20-25-6 comes originally from a February 1986 FASB Staff Q&A. That guidance refers to recognizing maintenance and customer support costs at the earlier of (1) when the related (i.e. PCS) revenue is recognized or (2) when those costs are incurred. [Highlights of Financial Reporting Issues February 1986 – Computer Software: Guidance on Applying Statement 86]

The earlier of guidance exists because, at that time, some entities recognized PCS revenue at the beginning of the PCS term (e.g. on January 1, Year 1 for PCS to be provided throughout all of Year 1). Accordingly, those entities were required to accrue the expected costs of providing the PCS at the time the PCS revenue was recognized.

5.3.50 Product enhancements

Excerpt from ASC 985-20

20 Glossary

Product Enhancement

Improvements to an existing product that are intended to extend the life or improve significantly the marketability of the original product. Enhancements normally require a product design and may require a redesign of all or part of the existing product.

55 Implementation Guidance and Illustrations

General

> Product Enhancements

• > Accounting for Costs of Product Enhancements

55-18 Costs incurred for **product enhancements** should be charged to expense as research and development until the technological feasibility of the enhancement has been established. If the original product will no longer be marketed, any unamortized cost of the original product should be included with the cost of the enhancement for purposes of applying the net realizable value test and amortization provisions. If the original product will remain on the market along with the enhancement, the unamortized cost of the original product should be allocated between the original product and the enhancement.

Production costs of product enhancements – i.e. development costs (e.g. coding and testing) incurred after technological feasibility of the enhancement is established (see section 5.2.20) – are capitalized or expensed on the same basis as production costs of a new software product (see section 5.3.10).

The capitalization window for a product enhancement – i.e. the period between when (1) technological feasibility of the enhancement is established and (2) the enhancement is available for general release – is assessed independently from the software product to which it relates. In general, a product enhancement is not undertaken until after the software product it will enhance has been made available for general release.

Product enhancements extend the life or significantly improve an existing software product's marketability. Other updates to existing software are generally considered maintenance (see section 5.3.40). [985-20 Glossary]

Distinguishing between maintenance and product enhancements frequently requires judgment. Involvement of the software development team will often be necessary. Involving the software development team early in the process will help to ensure costs are accounted for appropriately.

Question 5.3.50

What is the difference between the definitions of enhancement in Subtopic 985-20 vs Subtopic 350-40?

Interpretive response: Product enhancements under Subtopic 985-20 include improvements that extend the useful life of the software only. In contrast, enhancements to internal-use software must result in additional software functionality (i.e. the ability to perform additional tasks). [985-20 Glossary, 350-40-25-7]

Question 5.3.60

What does it mean to significantly improve the marketability of a software product?

Background: Marketability is not a defined term in Subtopic 985-20 or US GAAP more broadly. However, we believe marketability generally refers to a product's ability to be sold or its appeal to potential customers.

Interpretive response: In the context of the general definition of marketability and software, we believe a significant improvement to marketability typically necessitates additional or improved software functionality – i.e. the ability to perform additional tasks or functions, or to perform existing tasks or functions better or more efficiently. Additional or improved software functionality may increase the pool of customers that might license a software product or incent potential customers already considering the product to purchase a license.

We believe the marketability assessment should evaluate the expected marketability of the software with the enhancement as compared to what the marketability of the software would be without it over the same period.

Question 5.3.70

Can an entity use its expected point release scheme to determine if an update is an enhancement or maintenance?

Background: Entities in the software industry will frequently use a point (or dot) release scheme to maintain software versions. Major revisions or updates may result in a change in numerical value left of the decimal place (e.g. version 1.0 is replaced by version 2.0). Minor version updates will typically result in a change in the numerical value right of the decimal place (e.g. version 10.1 updated to version 10.2, or version 10.1.1 updated to version 10.1.2). However, point release schemes are not standardized across the software industry.

Interpretive response: It depends. Because point release schemes are not standard across software entities, each entity may have its own specific criteria and scheme. Entities should evaluate whether their internal point release

scheme is consistent with the definitions of 'product enhancement' and 'maintenance' in Subtopic 985-20 before using it to determine if updates in a point release include product enhancements or only maintenance updates.

Original product capitalized production costs

When a software product is enhanced, a key consideration in the go-forward accounting is whether both the original product and the enhanced product will be sold (licensed) to customers. Question 5.3.80 addresses when the original product will not, while Question 5.3.90 addresses when it will. [985-20-55-18]

Question 5.3.80

How should an entity account for capitalized original product production costs when the original product will not be marketed going forward?

Interpretive response: When only the enhanced product will be marketed to customers, the unamortized production costs of the original product are combined with the capitalized production costs of the enhancement, and accounted for as a single, enhanced software product asset going forward – i.e. for purposes of amortization and impairment testing. See section 6.4. [985-20-55-18]

Question 5.3.90

How are unamortized development costs allocated between an original product and an enhancement?

Background: If the original product will continue to be sold (licensed) to customers, the unamortized original product production costs should be allocated between the original product and the enhancement. [985-20-55-18]

Subtopic 985-20 does not provide guidance on how entities should allocate unamortized original product production costs between an original product that will continue to be marketed to customers together with an enhanced product.

Interpretive response: In the absence of authoritative guidance, we believe entities have the flexibility to allocate costs on any systematic and rational basis, applied consistently to similar circumstances.

In our experience, entities have applied a variety of approaches in practice. The following are two examples that we believe may be appropriate.

- **Future expected sales of each product.** For example, if sales going forward are expected to be 75/25 in favor of the enhanced product, that may suggest 75% of the remaining unamortized production costs should be allocated to the enhanced product asset.
- **Percentage of shared code.** For example, if the enhanced product incorporates all the original product code, that might suggest a 50/50

allocation of the unamortized original production costs to the two product assets.

5.4 Step 3: Determine when to cease capitalization of production costs

Excerpt from ASC 985-20

25 Recognition

General

> Production Costs of Computer Software

25-6 Capitalization of computer software costs shall cease when the product is available for general release to customers. Costs of **maintenance** and **customer support** shall be charged to expense when related revenue is recognized or when those costs are incurred, whichever occurs first.

Excerpt from ASC Master Glossary

Product Master

A completed version, ready for copying, of the computer software product, the documentation, and the training materials that are to be sold, leased, or otherwise marketed.

Capitalization of software production costs ceases once the software product is available for general release. [985-20-25-6]

Subtopic 985-20 does not define 'general release' or 'available for general release'. However, this typically refers to the point when the software is available for purchase by customers or for release to existing PCS customers.

General release availability may follow completion of the 'product master'. Costs other than software production costs, which include only the development costs (e.g. coding and testing) to produce the product master (see section 5.3.10), incurred before general release availability are outside the scope of Subtopic 985-20. Such costs may be inventory costs (see section 5.3.20). [985-20-25-3, 25-11]

The software product that is made available for general release is often referred to as the 'production release' or a 'stable release' and may refer to the 'product master'.

Question 5.4.10

Does capitalization cease at the start of a 'pre-release' stage?

Background: See Question 5.2.30 for background on pre-releases (or early releases).

Interpretive response: It depends. Software capitalization ceases when the software is available for *general* release – i.e. when the software is available for purchase (license) by customers. If the software is not yet available to the general customer market for purchase, or available to be released generally (e.g. through an update), during the pre-release stage, then the pre-release will not trigger the cessation of cost capitalization.

However, because software entities' own definitions and use of terms such as 'pre-release' or 'early release' are often not uniform, entities will need to consider their own pre-release facts and circumstances when considering this question. For example, an entity may offer an early version of its software to all (or most) of its customer base for license. In that case, despite characterization as an early release version of one software product, the early release version may meet the definition of a software product that is available for general release in its own right, and subsequent development costs should be considered under the guidance applicable to product enhancements.

Example 5.4.10

Capitalization of development costs incurred after general release

ABC Corp. developed external-use Software Product X using internal resources. X was available for general release (i.e. for customer purchase) on January 1, Year 1. Upon general release, X was only marketed and designed for Operating System Z. ABC originally established the technological feasibility of X based on meeting the detail program design criterion in paragraph 985-20-25-2(a).

The following activities occurred after general release.

— On June 1, Year 1, ABC released a software patch that updated X for software bugs identified after general release. Total remediation costs were $20,000.

— On December 1, Year 1, ABC released an updated version of X that executes on Operating System Y. $75,000 in development costs were incurred for this update. No high-risk development issues were encountered in making X executable on Operating System Y.

ABC expenses the $20,000 in costs incurred to fix the software bugs for the June Year 1 release. The updates were maintenance because they corrected existing software errors and did not either extend the life of X or significantly improve its marketability.

In contrast, ABC capitalizes the $75,000 in costs to port X to Operating System Y. Enabling X to execute on Operating System Y is a product enhancement

because Y is a widely used operating system such that porting X to execute on Y significantly increases the market to which ABC can sell licenses to X. ABC capitalizes all of the development costs in this case because:

— no substantive updates were needed to the detail program design of X; and
— no high-risk development issues were identified either before or during the development process for porting X to Y.

Affecting these conclusions was the fact that ABC has successfully ported software products from one operating system to another, including to Y, in the past.

ABC will sell licenses to both Product X for Operating System Z and Product X for Operating System Y. Therefore, ABC now has two Product X software assets (X for Z, and X for Y). ABC allocates the remaining, unamortized development costs of X to the two Product X software assets on a reasonable, systematic, and rational basis (see Question 5.3.90).

— The carrying amount of the X for Z asset comprises the remaining, unamortized original development costs of X allocated to it.

— The carrying amount of the X for Y asset includes its allocation of the remaining, unamortized original development costs plus the $75,000 in new development costs.

5.5 Agile software development

Section 3.2.60 and section 3A.2.10 provide an overview of agile software development methodology. Question 3.2.170 and section 3A.2 address applying Subtopic 350-40 to internal-use software development using an agile method.

Like Subtopic 350-40 (originally, SOP 98-1) before ASU 2025-06, Subtopic 985-20 comes from guidance issued before agile software development became commonplace (FAS 86 issued in 1985 and a February 1986 FASB staff Q&A). Therefore, questions also arise about how using an agile methodology affects the application of Subtopic 985-20. [Highlights of Financial Reporting Issues February 1986 – Computer Software: Guidance on Applying Statement 86]

Question 5.5.10

How does an agile software development process affect application of Subtopic 985-20?

Interpretive response: While not written for the agile method of software development, and issued before the agile method was introduced, Subtopic 985-20 nonetheless applies to external-use software development projects undertaken using an agile development process.

That said, we believe an agile development process may influence both:

— how and when technological feasibility is established; and
— the length and existence of any 'capitalization window'.

How and when technological feasibility is established

The sub-sections below address key ways (not exhaustive) an entity's agile software development process may affect when and how technological feasibility is established.

Use of detail program designs

As discussed in section 5.2, when and how technological feasibility is established depends on the entity's development process for the software product. This includes whether that process includes producing a detail program design. [985-20-55-2(a), 55-5]

In our experience, entities following an agile development methodology are less likely to produce detail program designs as part of their software development process. This is because the use of detail program designs, in general, reflects a level of planning and 'structure' to the development efforts that is inconsistent with an agile methodology that is based on more limited pre-planning, fast-paced development 'sprints' and *expected* re-work/iteration.

Therefore, it is more frequently the case that agile entities look to completion of a working model to establish the technological feasibility of a new software product or enhancement. And as discussed in Question 5.2.60, completion of a working model often comes after an entity with a different development process would have completed a detail program design.

For entities that may still produce detail program designs for some projects, the nature of agile development, which encourages iteration and re-work, may affect when an entity can conclude that all high-risk development issues are resolved, recalling that technological feasibility is not established until high-risk development issues are resolved through coding and testing (see section 5.2). [985-20-25-2(a)(3)]

Working model

As also discussed in Question 5.2.40, an entity applying an agile methodology may not produce a complete working model that meets the conditions in paragraph 985-20-55-9 and the criterion in paragraph 985-20-25-2(b)(2) until late in the development process – e.g. upon completion of a post-beta 'release candidate' (or similar version). 'Post-beta' in this context refers to a level of stability of, and remaining bugs in, the software that is superior to that typical of a beta version.

Length and existence of a 'capitalization window'

An agile development process may affect the duration and existence of the 'capitalization window' between when (1) technological feasibility of a software product or enhancement is established and (2) that product or enhancement is available for general release.

Late establishment of technological feasibility

For the reasons outlined above, some entities using an agile software development process may not establish technological feasibility, starting the capitalization window, until relatively late in the development cycle.

Release cycle

Customer and other testing of the software before general release may be limited under an agile approach. The entity may accept it will make more frequent releases to correct software bugs or make product enhancements, instead of undertaking more extensive testing and refinement of the software before releasing it. Consequently, the capitalization period after technological feasibility is established may be short.

Minor product updates

Product enhancements extend the life or significantly improve an existing software product's marketability; other updates to existing software are generally considered maintenance. Section 5.3.50 discusses the definition of, and what qualifies as, a product enhancement under Subtopic 985-20. [985-20 Glossary]

Because agile software development is often characterized by small-scale software development projects, some development efforts (i.e. updates) related to existing software products may not extend the life, or significantly improve the marketability, of the software product. Consequently, assuming the existing product is already available to the general market, there may be no capitalization window for these particular updates if they will not be bundled into a release version with other functionally interdependent updates that, together, meet the definition of a product enhancement.

Summary effect

Together, these frequent hallmarks of an agile development process – i.e. producing relatively late-stage working models to establish technological feasibility and a frequent release cycle based on limited pre-release testing – may have the effect of limiting (or reducing, as compared to non- or pre-agile software development) the pool of capitalizable production costs (see sections 5.3 and 5.4) for software products and enhancements developed under an agile process.

Question 5.5.20

How is the application of Subtopic 985-20 to agile development affected by whether the development is of a new software product or an enhancement?

Interpretive response: We believe an entity's considerations about applying Subtopic 985-20 to agile software development may be affected by whether the development is of a new 'software product' or a 'product enhancement' (see section 5.2.20).

New software products

Agile software development is often characterized by small software development projects or efforts that are not separately saleable and do not have independent exchange value – i.e. do not meet the definition of a 'software product' (see Question 5.2.10).

Because of this, technological feasibility is frequently not assessed for a single feature development or as part of each development sprint. Rather, it may be that only multiple sprints (often, many) together will produce a 'software product'. [985-20-55-1, 55-7]

Therefore, even though the development of a single feature may run through a complete design cycle, none of the development costs incurred may qualify for capitalization because technological feasibility has not yet been established – e.g. a complete working model does not yet exist – for the software product when the costs are incurred. [985-20-25-2(b), 55-7]

Product updates and enhancements

In contrast to new product development, even a single sprint may produce a 'product enhancement' to an existing software product (see 'Minor product updates' discussion in Question 5.5.10). Establishing technological feasibility for that enhancement may only require minor modifications to: [985-20 Glossary, 985-20-55-21]

— the existing product's detail program design; or
— the underlying software code to produce a working model that includes the enhancement.

In that way, bearing in mind the considerations in Question 5.5.10, the same feature for which no development costs were capitalized in a new product development *may* result in some cost capitalization if developed as an enhancement to an existing software product.

However, it is not a given that a single enhancement, or even small bundles of enhancements, will be released separately. Similar to new product development, multiple features may be bundled into a single release version, and therefore tested only together. So, technological feasibility may not be established for any single feature or smaller bundle of features to be included together in the release version before technological feasibility is established for *all* the features.

This bundling may have the effect of reducing the pool of capitalizable production costs for one or both of the following reasons.

— Entities bundling enhancements in this manner may release the enhanced software on a general basis shortly after a working model encompassing the bundle is completed (see 'release cycle' discussion in Question 5.5.10).
— Remaining production costs post-technological feasibility may be limited if the time taken to complete later features included in the bundle is used to continue to refine the development of features developed earlier, reducing the amount of such costs incurred on those earlier features after technological feasibility is established.

5.6 Funded software development arrangements

Excerpt from ASC 985-20

20 Glossary

Contract

An agreement between two or more parties that creates enforceable rights and obligations.

Customer

A party that has contracted with an entity to obtain goods or services that are an output of the entity's ordinary activities in exchange for consideration.

Revenue

Inflows or other enhancements of assets of an entity or settlements of its liabilities (or a combination of both) from delivering or producing goods, rendering services, or other activities that constitute the entity's ongoing major or central operations.

25 Recognition

General

> Funded Software-Development Arrangements

25-12 A funded software-development arrangement within the scope of Subtopic 730-20 shall be accounted for in conformity with that Subtopic. If the technological feasibility of the computer software product pursuant to the provisions of this Subtopic has been established before the arrangement has been entered into, Subtopic 730-20 does not apply because the arrangement is not a research and development arrangement. If capitalization of the software-development costs commences pursuant to this Subtopic and the funding party is a collaborator or a partner, any income from the funding party under a funded software-development arrangement shall be credited first to the amount of the development costs capitalized. If the income from the funding party exceeds the amount of development costs capitalized, the excess shall be deferred and credited against future amounts that subsequently qualify for capitalization. Any deferred amount remaining after the project is completed (that is, when the software is available for general release to customers and capitalization has ceased) shall be credited to income. If the counterparty is a **customer**, the entity shall apply the guidance of Topic 606 on **revenue** from **contracts** with customers.

60 Relationships

General

60-3 For software-development arrangements that are fully or partially funded by a party other than the vendor that is developing the software and for which technological feasibility of the computer software product has not been established before entering into the arrangement, see Subtopic 730-20 on research and development arrangements.

Software vendors often obtain funds from third parties for use in developing software. Funded software development arrangements may be structured differently depending on the circumstances. The accounting for such arrangements depends on the nature and terms of the arrangement.

See Question A30 and Examples A30.1 through A30.7 in KPMG Handbook, Revenue for software and SaaS, for guidance on software vendor accounting for funded software development arrangements.

6. Subsequent measurement

Detailed contents

New item added in this edition: **

6.2.130 How is abandonment of an internal-use software or CCA implementation cost asset that is part of a larger asset group accounted for?

6.2.140 Are unpaid license and non-license component fees accrued when an entity ceases use of the licensed software?

6.2.150 Are unpaid hosting service fees accrued when an entity ceases use of the cloud-based solution?

6.2.155 How should an entity account for unamortized software costs when an internal-use software license is converted to a CCA? **

6.2.160 Is the liability for an acquired software license to be paid for over time included in the carrying amount of the asset group?

6.2.170 Does the recoverability test include fees for non-license elements of a software licensing arrangement?

6.2.180 Does the recoverability test include CCA hosting service fees?

6.2.190 Does the recoverability test include variable payments?

6.2.200 Is internal-use software that is not expected to be completed written down before testing the related asset group for impairment?

6.2.210 What constitutes a 'pattern' of marketing internal-use software?

6.2.220 What evidence is required to overcome a rebuttable presumption that developed software will be marketed externally?

6.2.230 What are the accounting effects of a 'one-off' licensing arrangement for the vendor?

6.2.240 What proceeds should reduce the carrying amount of the internal-use software asset?

6.2.250 What Subtopic governs the accounting for any remaining software asset if its carrying amount is not reduced to zero under paragraphs 350-40-35-7 and 35-8?

Examples

6.2.10 Term software license paid for over license period

6.2.20 Term of the hosting arrangement

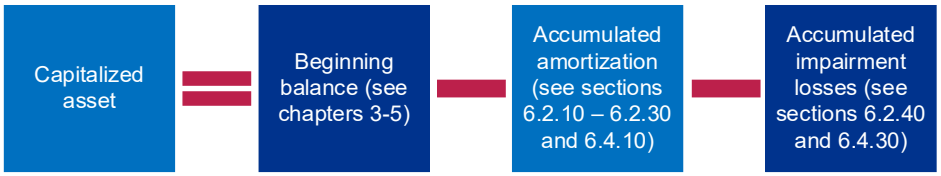
6.2.30 Amortization period and impairment testing considerations for multiple user licenses

6.2.40 Abandoning a module of a cloud-based solution subject to a CCA

6.2.50 Rebuttable presumption exists

6.1 How the standards work

After initial recognition and measurement, the software, website development and CCA implementation cost assets capitalized under Subtopics 350-40, 350-50 and 985-20 will generally all be measured as follows.



Amortization

Software development, website development and CCA implementation costs capitalized under Subtopics 350-40 and 350-50 are generally amortized on a straight-line basis.

Software costs capitalized under Subtopic 985-20 are also amortized. However, the amortization recorded each annual period is the *greater of*:

- the ratio of (1) gross current annual period revenues for the software product as compared to (2) gross current annual period revenues for the software product *plus* total expected future revenue for the software product; and
- the amortization that would result for the period from amortizing the software costs on a straight-line basis from the beginning of the annual period over the software product's remaining economic life.

Abandonment

Internal-use software assets (including those created from website development) and CCA implementation cost assets recognized under Subtopics 350-40 and 350-50 may be abandoned. That is, the entity may decide to cease use of the software, website or hosting service (to which a CCA implementation cost asset relates) before the end of its useful life or the end of the 'term of the hosting arrangement'. When an asset is slated to be abandoned, the entity accelerates its amortization so the carrying amount of the asset equals its salvage value at the date the entity abandons (i.e. ceases to use) it.

Impairment

In general, internal-use software assets (including those created from website development) and CCA implementation cost assets recognized under Subtopics 350-40 and 350-50 are assessed for impairment under the long-lived assets impairment guidance in Topic 360 (property, plant and equipment).

As an exception, an internal-use software asset may be impaired even if the asset group to which it belongs is not. This is the case when it is no longer probable that the internal-use software will be completed and placed in service;

the asset is measured at the lower of its carrying amount and fair value less costs to sell.

Capitalized production costs of external-use software are measured at the lower of their amortized carrying amount and NRV of the software product.

6.2 Internal-use software and cloud computing arrangements

6.2.10 Amortization: internal-use software

Excerpt from ASC 350-40

35 Subsequent Measurement

General

> Amortization

35-4 The costs of computer software developed or obtained for internal use shall be amortized on a straight-line basis unless another systematic and rational basis is more representative of the software's use.

35-5 In determining and periodically reassessing the estimated useful life over which the costs incurred for internal-use computer software will be amortized, entities shall consider the effects of all of the following:

a. Obsolescence
b. Technology
c. Competition
d. Other economic factors
e. Rapid changes that may be occurring in the development of software products, software operating systems, or computer hardware and whether management intends to replace any technologically inferior software or hardware.

Given the history of rapid changes in technology, software often has had a relatively short useful life.

35-6 For each module or component of a software project, amortization shall begin when the computer software is ready for its intended use, regardless of whether the software will be placed in service in planned stages that may extend beyond a reporting period. For purposes of this Subtopic, computer software is ready for its intended use after all substantial testing is completed. If the functionality of a module is entirely dependent on the completion of other modules, amortization of that module shall begin when both that module and the other modules upon which it is functionally dependent are ready for their intended use.

An internal-use software asset is amortized on a straight-line basis, unless another systematic and rational basis is more representative of the software's use, over its useful life. [350-40-35-4]

Amortization begins when a module or component of a software project is ready for its intended use – i.e. all substantial testing is completed. If the functionality of a module depends entirely on the completion of other modules, amortization of that module begins when both that module – and the other modules on which it is functionally dependent – are ready for their intended use. [350-40-35-6]

In both determining and periodically reassessing the estimated useful life of the software, an entity considers the following: [350-40-35-5]

- obsolescence;
- technology;
- competition;
- other economic factors; and
- rapid industry changes in the development of software products, software operating systems, or computer hardware; and whether management intends to replace any technologically inferior software or hardware.

New software development should trigger reassessment of the remaining useful lives of the existing software. When the existing software is replaced, its unamortized carrying amount is expensed when the new software is ready for its intended use. [350-40-25-15]

Question 6.2.10

What is the useful life of an internal-use software license asset?

Background: An acquired internal-use software license is an intangible asset. The useful life of an intangible asset that is based on a legal right is constrained by, and cannot extend beyond the duration of, that legal right. [350-30-35-3(c), 350-40-25-17]

Interpretive response: Determining the useful life of an internal-use software license asset considers the specific factors listed in Subtopic 350-40 (see above). [350-40-35-5]

However, as an intangible asset, we believe the useful life of an internal-use software license asset should be limited to the shorter of:

- the entity's legal right to use the software; and
- and the remaining economic life of the software, which may be relatively short.

In a term (i.e. non-perpetual) license scenario, we believe the period for which the entity has the legal right to use the software is the existing enforceable license period (i.e. the non-cancellable term) only.

We believe the useful life of an internal-use software license asset should not include expected renewal periods, even if the contract grants the entity renewal or extension options. This is principally because we do not believe the license fees the entity would owe in one or more renewal periods meet the definition of a liability before the entity exercises the related renewal or extension option.

And in the absence of recognizing a license fees liability, a license asset inclusive of renewal periods would not reflect its appropriate cost basis. For example, if an entity purchases a one-year term license with a one-year renewal option, the license asset cannot reflect a two-year useful life without including the second year license fees in the asset's cost basis and recording a liability for those unpaid fees.

Question 6.2.20

When is the useful life of an internal-use software asset reassessed?

Background: Subtopic 350-40 outlines what an entity should consider when reassessing the useful life of an internal-use software asset; however, it does not specify *when* or *how often* reassessment should occur. [350-40-35-5]

Interpretive response: We believe the useful life should be re-evaluated whenever there is a change in the factors, including those specifically outlined in Subtopic 350-40, that are important to the entity's existing useful life estimate. However, consistent with the guidance applicable to other intangible assets in Subtopic 350-30, the useful life should be evaluated at least each reporting period to determine whether events or circumstances, including changes to the factors specifically outlined in Subtopic 350-40, warrant a change to that life. Any change in the useful life is accounted for as a change in accounting estimate under Topic 250; see section 3.4 in KPMG Handbook, Accounting changes and error corrections. [350-30-35-9, 350-40-35-5]

Question 6.2.30

How does an entity assess whether the functionality of a module or component is 'entirely dependent' on another module or component?

Background: Subtopic 350-40 does not provide guidance about how to make this assessment, nor does the basis for conclusions to original AICPA Statement of Position (SOP) No. 98-1 (internal-use software) that was ultimately codified as Subtopic 350-40 discuss it.

Interpretive response: The guidance refers to the completed (i.e. ready for its intended use) module or component being *entirely* dependent on a not-yet-completed (but expected to be completed) module or component. However, we do not believe the guidance was intended to require amortization to begin if the completed module has only very limited (or insignificant) functionality on its own or together with other completed resources (e.g. other software applications, other modules/components or a cloud-based solution). [350-40-35-6]

Therefore, we believe a completed module or component should begin to be amortized if it has substantive (i.e. more than insignificant) functionality on its own or together with other completed resources. If that is not the case when the module or component is completed, amortization should commence when sufficient additional resources (e.g. additional modules or components) are completed such that the module or component then has substantive functionality together with those additional resources.

Question 6.2.40

How is an internal-use software license liability accounted for after initial recognition?

Background: When licensing internal-use software, the entity recognizes an intangible asset for the software license, and a liability for any unpaid license fees at the date the license is acquired. [350-40-25-17]

Section 3.3 (or section 3A.3) discusses initial recognition and measurement of an internal-use software license liability. Subtopic 350-40 provides no guidance on the subsequent measurement of the liability.

Interpretive response: We believe that after initial recognition:

- The entity measures the license liability on an amortized cost basis, consistent with any other financial liability. The liability is increased to reflect interest on the liability and decreased to reflect license payments made during the period.
- Interest on the license liability is calculated on an effective interest basis. See Question 3.3.40 (or Question 3A.3.40) for discussion of the appropriate discount rate to use for the liability.

Example 6.2.10

Term software license paid for over license period

Licensee LE and Licensor LR enter into an initial, three-year term license for LR's software product D, which is an inventory management application that LE will use internally only.

License payments:	Fixed payments of $10,000 per quarter in advance
Renewal options:	None
Termination options:	None
Discount rate:	10%
Transaction costs (LE):	None
Capitalized implementation costs (LE):	$15,000

LE expects to consume the economic benefits of the software license evenly over its useful life, which is three years (the contractual license period). Accordingly, LE amortizes the license asset on a straight-line basis.

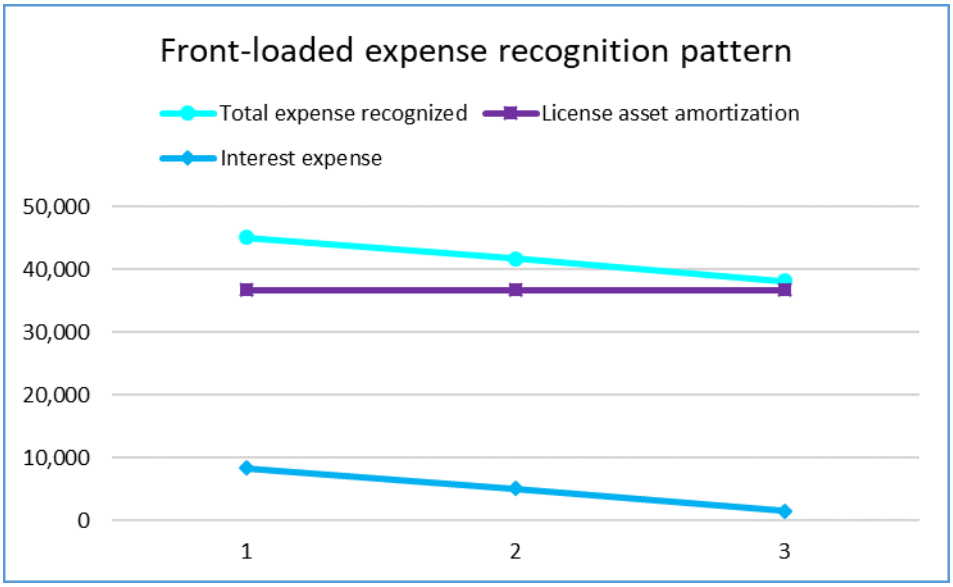
During the lease term, LE will account for the license asset and liability as follows (assuming no changes to the estimated useful life, modifications to the license or impairments).

Year	License liability				License asset		
	Beg. balance	Interest	Pmts.	End. balance	Beg. balance	Amort.	End. balance
1	$95,142	$8,353	$(30,000)	$73,495	$110,142	$(36,714)	$73,428
2	73,495	5,066	(40,000)	38,561	73,428	(36,714)	36,714
3	38,561	1,439	(40,000)	0	36,714	(36,714)	0

Observation

Front-loaded expense pattern for software licenses paid for over time

Because the internal-use software asset is generally amortized on a straight-line basis while interest is calculated on the license liability using the effective interest method, total expenses for an internal-use software license paid for over the license period will generally result in a front-loaded pattern of total expense recognition. This is illustrated in the following chart using the fact pattern in Example 6.2.10.



6.2.20 Amortization: CCA implementation cost assets

Excerpt from ASC 350-40

35 Subsequent Measurement

Implementation Costs of a Hosting Arrangement That Is a Service Contract

> Amortization

35-13 Implementation costs capitalized in accordance with the Implementation Costs of a Hosting Arrangement That Is a Service Contract Subsections of this Subtopic shall be amortized over the term of the associated hosting arrangement, considering the guidance in paragraph 350-40-35-17, on a straight-line basis unless another systematic and rational basis is more representative of the pattern in which the entity expects to benefit from access to the hosted software. This Subsection considers the right to access the hosted software to be equivalent to actual use, which shall not be affected by the extent to which the entity uses, or the expectations about the entity's use of, the hosted software (for example, how many transactions the entity processes or expects to process or how many users access or are expected to access the hosted software).

35-14 An entity (customer) shall determine the term of the hosting arrangement that is a service contract as the fixed noncancellable term of the hosting arrangement plus all of the following:

a. Periods covered by an option to extend the hosting arrangement if the entity (customer) is reasonably certain to exercise that option
b. Periods covered by an option to terminate the hosting arrangement if the entity (customer) is reasonably certain not to exercise that option
c. Periods covered by an option to extend (or not to terminate) the hosting arrangement in which exercise of the option is controlled by the vendor.

35-15 An entity (customer) shall periodically reassess the estimated term of the arrangement and shall account for any change in the estimated term as a change in accounting estimate in accordance with Topic 250 on accounting changes and error corrections.

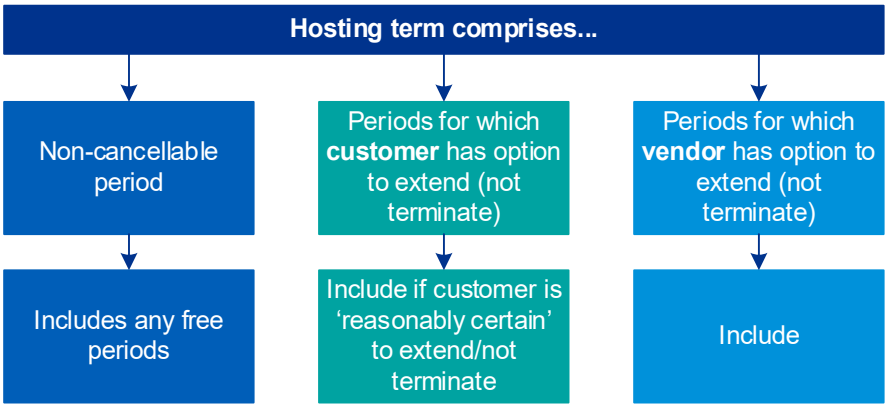
35-16 An entity shall consider the effects of all the following when determining the term of the hosting arrangement in accordance with paragraph 350-40-35-14 and when reassessing the term of the hosting arrangement in accordance with paragraph 350-40-35-15:

a. Obsolescence
b. Technology
c. Competition
d. Other economic factors
e. Rapid changes that may be occurring in the development of hosting arrangements or hosted software
f. Significant implementation costs that are expected to have significant economic value for the entity (customer) when the option to extend or terminate the hosting arrangement becomes exercisable.

35-17 For each module or component of a hosting arrangement, an entity shall begin amortizing the capitalized implementation costs related to the hosting arrangement that is a service contract when the module or component of the hosting arrangement is ready for its intended use, regardless of whether the overall hosting arrangement will be placed in service in planned stages that may extend beyond a reporting period. For purposes of this Subsection, a hosting arrangement (or a module or component of a hosting arrangement) is ready for its intended use after all substantial testing is completed. If the functionality of a module or component is entirely dependent on the completion of other modules or components, the entity shall begin amortizing the capitalized implementation costs related to that module or component when both that module or component and the other modules or components upon which it is functionally dependent are ready for their intended use.

CCA implementation cost assets are amortized on a straight-line basis, unless another systematic and rational basis is more representative of the software's use, over the 'term of the hosting arrangement'. [350-40-35-13]

The term of the hosting arrangement comprises the noncancellable period of the CCA plus any optional renewal periods that are (1) reasonably certain to be exercised by the customer or (2) for which exercise of the option is controlled by the vendor. [350-40-35-14]



The FASB staff observed during public EITF deliberations that 'reasonably certain' should be applied consistently with how it is applied under Topic 842 (leases).

Under Topic 842, 'reasonably certain' is a high threshold of probability that must be met to include optional lessee payments in the measurement of lease assets and lease liabilities, intended to have the parties account for a lessee renewal option (or option *not* to terminate the lease) only when the lessee has a *compelling* economic reason to exercise it (see section 5.2 of KPMG Handbook, Leases). [ASU 2016-02.BC194, BC197, BC218]

An entity is required to 'periodically reassess' the estimated term of the hosting arrangement over which capitalized implementation costs are amortized. When doing so, the entity considers the same factors as it considered initially. A change in the amortization period is accounted for prospectively as a change in accounting estimate. [350-40-35-15]

An entity considers all of these factors when determining or reassessing the term of the hosting arrangement – i.e. when determining or reassessing whether the entity is reasonably certain to exercise a renewal option (or not to exercise a termination option): [350-40-35-16]

- obsolescence;
- technology;
- competition;
- other economic factors;
- rapid changes that may be occurring in the development of hosting arrangements or hosted software; and
- significant implementation costs that are expected to have significant economic value for the customer when the option to extend (or not to terminate) the hosting arrangement becomes exercisable.

Observation

Determining the 'term of the hosting arrangement' for cancellable (evergreen) CCAs may be difficult

The term for evergreen CCAs (i.e. those on a month-to-month or year-to-year basis) is established in the same manner as for all other CCAs, which means considering whether the entity is reasonably certain to exercise one or more available renewal options (including by not terminating the CCA).

Determining whether an entity is reasonably certain to exercise a renewal option in an evergreen arrangement may involve significant judgment. In general, we believe the shorter the non-cancellable period of the CCA, the greater the likelihood the entity is reasonably certain to exercise one or more renewal options. This is because, in many cases, it may be cost prohibitive to continually substitute software solutions – i.e. implement either a different cloud-based or on-premise software solution.

For example, if an entity uses a cloud-based solution to process its payroll for employees, it would generally be cost-prohibitive to frequently change solutions, especially if doing so would involve significant implementation costs – including those not capitalizable under Subtopic 350-40, such as training and data conversion/migration costs.

Question 6.2.50

Do significant implementation costs that cannot be capitalized influence the term of the hosting arrangement?

Interpretive response: Yes. Subtopic 350-40 does not indicate that only capitalized implementation costs can influence whether an entity is reasonably certain to extend (or not terminate) a CCA. [350-40-35-16]

Further, it is rational an entity would consider the incurrence of significant non-capitalizable implementation costs, such as significant data conversion or training costs, when deciding whether to extend (or not terminate) a CCA.

Question 6.2.60

Can capitalized CCA implementation costs be amortized based on expected usage of the hosted software?

Interpretive response: No. Subtopic 350-40 bases the amortization pattern for capitalized CCA implementation costs on the entity's access to the hosted software, not on its use thereof. [350-40-35-13]

Therefore, if the entity has equal access to the hosted software throughout the term of the hosting arrangement, amortization is on a straight-line basis. This applies even if the entity expects to use the hosted software on an uneven basis – e.g. more during certain periods of the year, or during certain years of the CCA.

Question 6.2.70

What determines the non-cancellable period of a CCA?

Interpretive response: The non-cancellable period of a CCA is the period for which there are presently enforceable rights and obligations on the entity and the cloud service provider. This determination depends on the relevant laws and enforcement practices in the governing jurisdiction to which the contract is subject.

However, as a practical matter, in most cases presently enforceable rights and obligations do not exist beyond when the CCA can be cancelled unilaterally by either party, assuming each party's cancellation/termination option is substantive.

- Once the customer has the right to cancel the CCA, the customer no longer has an enforceable obligation to make the payments and the vendor no longer has an enforceable right to receive payments.
- Similarly, once the vendor has the right to terminate the CCA, the customer no longer has an enforceable right to access the hosted software and the vendor no longer has an enforceable obligation to permit the customer to access the hosted software.

Substantive cancellation/termination option

Determining whether a cancellation or termination option is substantive frequently requires judgment. This is because there is no single, uniform US GAAP definition or application of 'substantive', even in the narrow context of contractual cancellation or termination options.

However, we believe it is instructive the EITF fashioned the 'term of the hosting arrangement' (which is the amortization period for capitalized CCA implementation costs) guidance after that in Topic 842 (leases). Under Topic 842, when determining the 'lease term', a cancellation or termination option that gives rise to a more-than-insignificant penalty is treated as non-substantive; see section 5.3 of KPMG Handbook, Leases.

Further, 'penalty' has a broad meaning; it encompasses economic penalties beyond any requirement for the terminating party to make a cash payment to the other party. [842 Glossary]

Consequently, we believe it is reasonable to similarly conclude a CCA cancellation or termination option is not substantive when it gives rise to a more than insignificant economic penalty.

Question 6.2.80

Does a clause that requires notice before formal termination affect the non-cancellable period of a CCA?

Interpretive response: Yes. When either the entity or the cloud service provider has the right to terminate a CCA at any time on giving notice to the other party, the non-cancellable period of the CCA is the 'notice' period.

For example, a CCA may grant each party the unilateral, substantive right to terminate the contract, for any reason (see Question 6.2.70), on giving 90 days' notice to the other party. This means at any point before such notice is given by either party, enforceable rights and obligations exist for both parties for 90 days. Therefore, at commencement of the CCA, and until either party gives notice of its intent to terminate it, the non-cancellable period of the CCA is 90 days.

Question 6.2.90

Can the term of the hosting arrangement include periods over which neither the customer nor vendor have a renewal option?

Background: CCAs may not include stated renewal options, even though the customer and the cloud service provider expect to agree on one or more renewals (e.g. through contract amendment).

Interpretive response: No. The amortization period for capitalized CCA implementation costs in these arrangements is expressly limited to the non-cancellable term of the CCA plus periods for which the entity or the cloud service provider has an option to extend. If the CCA does not include any renewal options, the amortization period for capitalized implementation costs in these arrangements is limited to the non-cancellable term of the CCA. [350-40-35-14]

Therefore, the term of the hosting arrangement may be shorter than either (1) the period over which the customer reasonably expects to benefit from the

costs incurred, or (2) what the term of the hosting arrangement would have been if the CCA had included renewal options.

In this respect, the guidance on the term of the hosting arrangement is consistent with the following EITF conclusions.

— The CCA implementation costs 'attach to the [CCA] contract'. Because the CCA implementation cost asset attaches to the contract – i.e. versus another asset, such as developed internal-use software or an internal-use software license – its amortization period cannot extend beyond the enforceable period of the contract. [ASU 2018-15.BC8]

— The amortization period for the capitalized costs should be the 'term of the hosting arrangement' because the future economic benefit that will be derived from those costs is inextricably linked to the customer having the continued right to access the hosted software. [ASU 2018-15.BC11]

Question 6.2.100

When does an entity reassess the term of the hosting arrangement over which capitalized implementation costs are amortized?

Background: Subtopic 350-40 does not provide guidance on: [350-40-35-15]

— how often an entity should reassess the term of the hosting arrangement; or
— what events or changes in circumstances should trigger a reassessment.

Therefore, the question arises about how often reassessment should occur, and whether, similar to lessees' reassessment of the lease term under Topic 842 (see section 6.6 of KPMG Handbook, Leases), reassessments should occur only on the occurrence of specified events or circumstances. [842-10-35-1, 55-28]

Interpretive response: We believe it is clear, although not explicitly stated, that the term of the hosting arrangement must be reassessed if the CCA is modified or the entity exercises a renewal or termination option that is not already included in that term.

Otherwise, we believe a plain reading of the guidance indicates entities should develop and maintain processes and controls to identify substantive changes to any of the factors expressly identified for consideration when determining or reassessing the term of the hosting arrangement. To the extent there is a change to one or more of them, the entity should reassess the term of the hosting arrangement. [350-40-35-16]

In addition, the term of the hosting arrangement should be revised if the entity commits to a plan to abandon (i.e. cease use of) the cloud-based solution. Section 6.2.30 discusses abandonment.

In the absence of further amendments to US GAAP or guidance from the SEC staff, we do not believe entities should infer a 'triggering event' approach to reassessing the term of the hosting arrangement so as to reassess the term of the hosting arrangement less frequently. The FASB's decision to adopt such an approach for lease term reassessments was specifically debated and codified in

Topic 842; no such debate occurred when developing, and no similar guidance exists in, Subtopic 350-40.

Example 6.2.20
Term of the hosting arrangement

Customer enters into a non-cancellable CCA with Cloud Service Provider (CSP) for its hosted payroll processing solution. The non-cancellable term of the CCA is three years, commencing January 1, Year 1.

The contract grants Customer an option to renew the CCA for an additional three years at an annual fee equal to that of the non-cancellable term, subject to adjustment based on increases in the Consumer Price Index (CPI) during the initial, non-cancellable term.

Customer incurs implementation costs it appropriately capitalizes when incurred. Some of these costs are incurred after the beginning of the non-cancellable term (January 1, Year 1), but before go-live (see Question 3.4.10 and Question 3A.4.10).

Scenario 1: Minor implementation costs

The following additional facts are relevant.

— Customer's incurred implementation costs (those capitalized and those expensed as incurred – see Question 6.2.50) are minor.

— There are several alternative payroll processing solutions available in the marketplace, many of which also require only minor costs to implement.

— Because of ongoing technological advancement in this space, it is at least reasonably possible an improved alternative will be available by the time Customer will need to decide whether to exercise its renewal option with CSP.

Based on these facts and circumstances, Customer concludes it is not reasonably certain to exercise its renewal option. Therefore, the term of the hosting arrangement is only three years, equal to the non-cancellable term of the CCA.

The hosted solution went live on March 1, Year 1; therefore, the capitalized implementation costs will be amortized over the remainder of the hosting arrangement term from that date (34 months; March 1, Year 1 – December 31, Year 3).

Scenario 2: Significant implementation costs

Assume the same facts as Scenario 1, except that Customer's incurred implementation costs (capitalized and expensed as incurred – see Question 6.2.50) are significant; more than the hosting service fees for the non-cancellable term. In addition, the hosted solution took longer to implement, going live on July 1, Year 1.

Despite other factors (e.g. the availability of alternatives and ongoing technological advancement), the significance of the implementation costs leads Customer to conclude it is reasonably certain to exercise its option to renew the

hosted solution for an additional three-year period. Customer's upfront investment in this solution is such it has a compelling economic reason to renew. Consequently, the term of the hosting arrangement is six years (January 1, Year 1 – December 31, Year 6).

Because the hosted solution did not go live until July 1, Year 1, Customer's capitalized implementation costs will be amortized from that date to the end of the hosting arrangement term (a period of 66 months).

Example 6.2.30

Amortization period and impairment testing considerations for multiple user licenses

Customer enters into a CCA with Cloud Service Provider (CSP) that permits 500 named users to access the SaaS. The initial, non-cancellable term of the CCA is three years.

Customer can renew for three additional years. The annual fee each year in Years 4-6 will be the same as the fee in the prior year plus an adjustment for any increase in the Consumer Price Index (CPI), provided Customer retains the same maximum number of named users.

Customer can renew as few as 100 users, and up to the original 500 users, in 50 user blocks. The price per user increases with fewer users.

Customer incurs significant costs to implement the SaaS solution; both the amounts capitalized and expensed as incurred are significant (see Question 6.2.50). The significance of the implementation costs, combined with Customer's expected continued need for the CSP solution or a comparable solution, leads Customer to conclude it is reasonably certain to renew the CCA beyond the initial, non-cancellable three-year period.

Customer's capacity needs fluctuate over time. Therefore, it is reasonably possible its needs for the SaaS solution will be less than 500 users after the end of the initial term. However, Customer concludes it cannot realize meaningful benefit from use of the solution with less than 300 users; if it were to go below that number, it would need to implement a supplemental solution to fulfill its needs or replace the solution entirely.

Based on these facts, Customer concludes it is reasonably certain to renew 300 users beyond the non-cancellable period, but it is not reasonably certain to renew the other 200 user rights.

The unit of subsequent accounting for capitalized CCA implementation costs is the cloud-based solution (there are not separate modules/components in this example). Therefore, Customer does not allocate implementation costs to users. Even though Customer is only reasonably certain to renew the SaaS for 300 users (not 500), it does not estimate a separate hosting arrangement term (i.e. amortization period) for a portion of the CCA implementation cost asset.

Note: Customer's conclusions in this regard will need to be considered in other aspects of its accounting for the capitalized CCA implementation costs for this SaaS solution – e.g. when evaluating the implementation cost asset for

impairment, its Step 1 recoverability test cash flows should reflect these same assumptions (see section 6.2.40).

Question 6.2.110

What is the effect of new CCAs on existing CCA implementation cost assets?

Background: The internal-use software guidance in Subtopic 350-40 states that new software development activities trigger reassessment of the remaining useful lives of the existing software that will be replaced. When the existing software is replaced because the new software is ready for its intended use, the entity expenses the unamortized costs of the old software. [350-40-25-15]

The guidance on implementation costs of a hosting arrangement that is a service contract does not contain similar explicit guidance for capitalized CCA implementation costs. Therefore, the question arises about what effect new CCAs could have on existing CCA implementation cost assets.

Question 3.2.100 and Question 3A.2.140 discuss the treatment of CCA implementation costs incurred after go-live, including those incurred as a result of entering into a new CCA.

Interpretive response: We believe an entity with existing CCA implementation cost assets that enters into a new CCA should consider the effect of the new cloud-based solution on existing cloud-based solutions for which implementation costs are capitalized.

For example, if the new solution will replace an existing solution (or module/component thereof), that may mean any related CCA implementation cost asset(s) is impaired (see section 6.2.40) or likely to be abandoned before the end of its current amortization period (see section 6.2.30).

Less overtly, the new solution may not replace an existing solution, but may suggest changes in factors that should trigger a reassessment of the term of the hosting arrangement for one or more existing CCA implementation cost assets (see Question 6.2.100). For example, the new solution may indicate a shift or advancement in relevant technology or affect whether a CCA for an existing solution that interfaces/interacts with the new solution is likely to be renewed or extended.

6.2.30 Abandonment

Excerpt from ASC 350-40

35 Subsequent Measurement

General

> Impairment

35-2 Paragraphs 360-10-35-47 through 35-49 requires that the asset be accounted for as abandoned when it ceases to be used.

Implementation Costs of a Hosting Arrangement That Is a Service Contract

> Impairment

35-12 Paragraphs 360-10-35-47 through 35-49 require that the asset be accounted for as abandoned when it ceases to be used. Implementation costs related to each module or component of a hosting arrangement that is a service contract shall be evaluated separately as to when it ceases to be used.

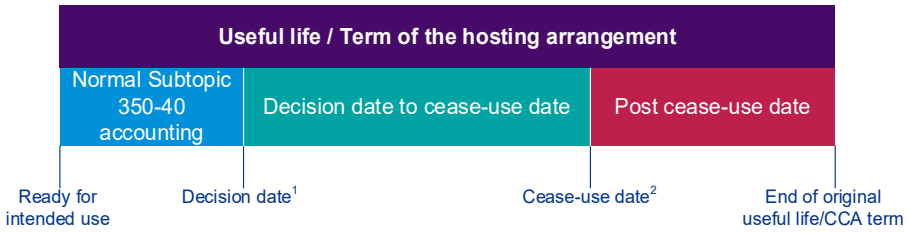
When an entity ceases use of internal-use software (including an internal-use software license), the related asset is accounted for as abandoned under paragraphs 360-10-35-47 to 35-49. [350-40-35-2]

When an entity ceases to use a cloud-based solution subject to a CCA, any CCA implementation cost asset associated with that solution is accounted for as abandoned under paragraphs 360-10-35-47 to 35-49. [350-40-35-12]

A long-lived asset to be abandoned is disposed of when it ceases to be used. If the entity commits to a plan to abandon a long-lived asset before the end of its previously estimated useful life, depreciation (or amortization) estimates are revised under Topic 250 (accounting changes and error corrections). At the point when the asset ceases to be used, its carrying amount should not exceed its salvage value. [360-10-35-47 – 35-48]

An asset that is only temporarily idled is not abandoned because it will be used in the future. [360-10-35-49]

The following diagram illustrates the terms and timing of the accounting considerations that are discussed throughout this section.



Notes:

1. The 'decision date' is the date on which the entity commits to the plan to cease use of the internal-use software or cloud-based solution (or module/component thereof).

2. The 'cease-use date' is the date on which the entity permanently stops using the internal-use software or cloud-based solution – e.g. the date the entity permanently stops processing transactions using the software/cloud-based solution.

Question 6.2.120

Does an entity recognize any effect of a planned abandonment before it ceases use of the internal-use software or cloud-based solution?

Interpretive response: Yes. Consistent with the Topic 360 guidance applicable to other long-lived assets, amortization of an internal-use software or CCA implementation cost asset is accelerated from the point in time the entity commits to a plan to cease use of the related internal-use software or cloud-based solution (or module/component). [360-10-35-47 – 35-48]

This applies to a third-party software license or a CCA even if the cease-use date will precede the end of the contractual license or CCA term.

In our experience, it is rare for an abandoned internal-use software license or CCA implementation cost asset to have salvage value. Therefore, it would generally not be appropriate to continue to recognize an internal-use software license or CCA implementation cost asset beyond the cease-use date of the licensed software or cloud-based solution (or module/component).

If the software or cloud-based solution has multiple modules or components and the entity commits to a plan to cease use of only one or some of them, amortization of the internal-use software or CCA implementation cost asset(s) related to that (those) module(s)/component(s) is accelerated, consistent with the entity's plan to cease use. The amortization of the internal-use software or CCA implementation cost assets for modules/components that will not be abandoned is not accelerated.

Because of this, entities may frequently need to identify separate internal-use software or CCA implementation cost assets for the modules/components within a larger software application or cloud-based solution (see Question 3.2.10 and Question 3A.2.20). It may be easiest to do this upfront (e.g. as the costs to implement the CCA are incurred), but otherwise entities will need to do this subsequently if they abandon only one or some modules/components of a software or cloud-based solution.

Question 6.2.130

How is abandonment of an internal-use software or CCA implementation cost asset that is part of a larger asset group accounted for?

Interpretive response: We believe an entity should undertake the following steps to account for the abandonment of an internal-use software or CCA implementation cost asset that is part of a larger Topic 360 asset group.

1. Assess fair value of uncompleted software

If internal-use software is uncompleted and not in service when the entity commits to the plan to abandon it, it is immediately written down to its fair value (presumed to be zero), less costs to sell. See section 6.2.40. [350-40-35-3]

If the internal-use software is complete and in service when the entity commits to a plan to abandon it, or the asset to be abandoned is a CCA implementation cost asset, this step does not apply. Proceed to Step 2.

2. Evaluate whether the asset group is impaired

The entity continues to evaluate impairment under Topic 360 at the same asset group level after committing to the plan to abandon the asset as before committing to the plan. This is because the plan, by itself, would not trigger a reassessment of the asset group to which the software or CCA implementation cost asset belongs. See section 3.3.50 of KPMG Handbook, Impairment of nonfinancial assets, for guidance on revising asset groups.

However, committing to the plan of abandonment may constitute an impairment triggering event, requiring an assessment of possible impairment for the larger asset group at the decision date. That assessment, as well as the recognition of any impairment loss, follows the guidance in Topic 360. An impairment indicator associated with a single internal-use software or CCA implementation cost asset within a larger asset group may not signify a need to test the entire asset group for impairment. An entity should consider the significance of the to-be-abandoned asset to the asset group before concluding the asset group needs to be tested for impairment. See section 4.3 of KPMG Handbook, Impairment of nonfinancial assets, for guidance about when to test an asset (asset group) for impairment. [360-10-35-21]

If the asset group is not impaired, there is no basis in Topic 360 or Subtopic 350-40 to immediately write down the carrying amount of the internal-use software or CCA implementation asset that will be abandoned except as required by paragraph 350-40-35-3 (see Step 1).

3. Determine go-forward accounting

It is necessary to consider the go-forward accounting at the decision date.

Asset accounting

If the internal-use software or CCA implementation cost asset is either (1) not impaired or (2) only partially impaired in Step 1 or Step 2, the entity should shorten its estimate of the useful life of the internal-use software or the term of the hosting arrangement to ensure that the to-be-abandoned internal-use software or CCA implementation cost asset is amortized to its salvage value (typically $0) over the period of its remaining expected use.

License liability accounting

A decision to abandon an internal-use software license asset does not affect the accounting for any unpaid license fees liability. The license liability continues to be accounted for as before until it is extinguished (see Question 6.2.40).

Question 6.2.140

Are unpaid license and non-license component fees accrued when an entity ceases use of the licensed software?

Interpretive response: The answer differs for unpaid license fees and non-license element (e.g. PCS, hosting services) fees.

Unpaid license fees

Unpaid license fees are not accrued under Topic 420 (exit or disposal cost obligations) because a liability for any unpaid license fees is already recorded under Subtopic 350-40. [350-40-25-17]

Unpaid non-license element fees

An entity that ceases use of licensed internal-use software, but remains liable for non-license element fees – i.e. these fees will continue to be incurred after the cease-use date without remaining economic benefit to the entity – will accrue a liability under Topic 420 for the remaining unpaid payments attributable to those elements at the cease-use date. [420-10-15-3, 420 Glossary, 420-10-25-11]

Question 6.2.150

Are unpaid hosting service fees accrued when an entity ceases use of the cloud-based solution?

Interpretive response: Yes. CCAs are executory contracts in the scope of Topic 420 (exit or disposal cost obligations). A customer that ceases use of a cloud-based solution subject to a CCA, or a module or component thereof, accrues a liability, measured at fair value, for the remaining unpaid hosting service fees – and any other ancillary costs that will continue to be incurred under the CCA without economic benefit to the entity – as of the cease-use date. This includes an estimate of any variable hosting service fees (e.g. usage or transaction-based fees). [420-10-15-3, 420 Glossary, 25-11, 30-7, 30-9]

Unpaid implementation fees

If the customer is paying for upfront implementation services over time, unpaid implementation fees (which may be an allocated portion of the CCA subscription fees being paid over the CCA term) at the cease-use date should already be accrued – i.e. they should have been accrued when the implementation services were provided. Therefore, no additional accrual for the unpaid portion of those fees is required at the cease-use date.

Example 6.2.40
Abandoning a module of a cloud-based solution subject to a CCA

Customer incurred costs to implement a cloud-based solution. Customer purchased, and the CCA covers, two modules: a payroll processing module and an HR module. Implementation was completed for both modules on the same date.

The following additional facts are relevant.

Hosting service fees:	$100 per month, paid in advance
Remaining term of the hosting arrangement at go-live date:	5 years
Go-live date:	January 1, Year 3
Capitalized implementation costs – payroll module:	$400
Capitalized implementation costs – HR module:	$300
Relative stand-alone prices (per month):	$50 (payroll module); $50 (HR module)
Credit-adjusted risk-free rate:	6%

On June 30, Year 4 (decision date), Customer commits to a plan to cease use of the payroll module effective March 31, Year 5 (cease-use date). This commitment is evidenced by a contract with a payroll outsourcing provider dated June 30, Year 4, for a new payroll solution expected to go live, after a period of parallel processing, on April 1, Year 5.

Customer has the following CCA implementation cost assets recorded at the decision date.

Payroll module:[1]	$120
HR module:[2]	$90

Notes:

1. Initial asset ($400) less amortized portion as of decision date ([$400 ÷ 5 years] × 3.5 years since go-live = $280).
2. Initial asset ($300) less amortized portion as of decision date ([$300 ÷ 5 years] × 3.5 years since go-live = $210).

Customer's accounting for the HR module and its capitalized implementation costs is unaffected by the decision to abandon the payroll module.

However, the decision to abandon the payroll module means the related CCA implementation cost asset should be amortized to $0 (no expected salvage value) by the cease-use date. Customer will amortize the remaining CCA implementation cost asset ($120) from the decision date to the cease-use date.

Customer has the following CCA implementation cost assets recorded as of December 31, Year 4.

Payroll module:[1]	$40
HR module:[2]	$60

Notes:

1. Carrying amount as of decision date ($120) less amortization since decision date ([$120 ÷ 9 months] × 6 months = $80).
2. Initial asset ($300) less amortized portion as of December 31, Year 4 (($300 ÷ 5 years) × 4 years since go-live = $240).

At the cease-use date, the carrying amounts of the CCA implementation cost assets are as follows. In addition, Customer records a Topic 420 liability for the remaining unpaid hosting service fees.

Payroll module:[1]	$ -
HR module:[2]	$45
Topic 420 liability – payroll module:[3]	$441

Notes:

1. Amortized to $0 by cease-use date.
2. Initial asset ($300) less amortized portion as of March 31, Year 5 (($300 ÷ 5 years) × 4.25 years since go-live = $255).
3. 9 remaining months' hosting service fees for the payroll module (9 × $50 = $450), discounted using the credit-adjusted risk-free rate of 6% (rounded) – note: there are no other associated costs without economic benefit to Customer to consider in this fair value liability measurement.

Question 6.2.155**

How should an entity account for unamortized software costs when an internal-use software license is converted to a CCA?

Background: Consider a scenario in which an entity previously purchased a software license. The entity capitalized the software license (see section 3.3 and section 3A.3) and significant costs to customize and implement the software (see section 3.2 and section 3A.2). The combined license and customization/implementation costs were recorded as an intangible asset (see section 7.2.10).

Later, the entity and the software vendor amend their contract to convert the on-premise software license to a CCA; that is, the vendor will take over hosting of the customized software and the entity will have no right to re-take possession of the software in the future (see section 2.5).

In this scenario, questions arise about how the entity should account for the unamortized intangible software license asset (comprising the capitalized license fees and customization/implementation costs).

Interpretive response: We believe the modified arrangement reflects a termination of the license such that the portion of the capitalized internal-use

software asset that relates to the software license fees should be treated as abandoned (or planned for abandonment if the migration to the cloud will occur in the future instead of immediately), as should any portion of the capitalized implementation costs that will not continue to benefit the entity post-migration (e.g. costs that were incurred to install the software to the entity's hardware).

By contrast, we believe it is appropriate to continue to recognize unamortized customization and implementation costs that will continue to benefit the entity post-migration. Such costs would generally have also qualified for capitalization as CCA implementation costs. Therefore, we believe it is appropriate to treat those costs as not having been abandoned.

Upon contract modification, the amortization period for remaining unamortized customization and implementation costs (as capitalized CCA implementation costs now instead of part of an intangible software license asset) should be based on the 'term of the hosting arrangement' (see section 6.2.20). This amortization period may be different from the remaining pre-modification 'useful life' of the internal-use software asset.

Upon migration, these costs, and amortization thereof, should begin to be presented in the financial statements in the same manner as any other CCA implementation costs (see section 7.2.20).

6.2.40 Impairment

Excerpt from ASC 350-40

35 Subsequent Measurement

General

> Impairment

35-1 Impairment shall be recognized and measured in accordance with the provisions of Section 360-10-35, which requires that assets be grouped at the lowest level for which there are identifiable cash flows that are largely independent of the cash flows of other groups of assets. The guidance is applicable, for example, when one of the following events or changes in circumstances occurs related to computer software being developed or currently in use indicating that the carrying amount may not be recoverable:

a. Internal-use computer software is not expected to provide substantive service potential.
b. A significant change occurs in the extent or manner in which the software is used or is expected to be used.
c. A significant change is made or will be made to the software program.
d. Costs of developing or modifying internal-use computer software significantly exceed the amount originally expected to develop or modify the software.

35-2 Paragraphs 360-10-35-47 through 35-49 requires that the asset be accounted for as abandoned when it ceases to be used.

35-3 When it is no longer probable that computer software being developed will be completed and placed in service, the asset shall be reported at the lower of the carrying amount or fair value, if any, less costs to sell. The rebuttable presumption is that such uncompleted software has a fair value of zero. Indications that the software may no longer be expected to be completed and placed in service include the following:

a. A lack of expenditures budgeted or incurred for the project.
b. Programming difficulties that cannot be resolved on a timely basis.
c. Significant cost overruns.
d. Information has been obtained indicating that the costs of internally developed software will significantly exceed the cost of comparable third-party software or software products, so that management intends to obtain the third-party software or software products instead of completing the internally developed software.
e. Technologies are introduced in the marketplace, so that management intends to obtain the third-party software or software products instead of completing the internally developed software.
f. Business segment or unit to which the software relates is unprofitable or has been or will be discontinued.

Pending Content

Transition Date: (P) December 16, 2027; (N) December 16, 2027 | Transition Guidance: 350-40-65-4

35-3 If the capitalization requirements in paragraphs 350-40-25-12 through 25-12A are ~~When it is~~ no longer met ~~probable that computer software being developed will be completed and placed in service~~ for software being developed, the asset shall be reported at the lower of the carrying amount or fair value, if any, less costs to sell. The rebuttable presumption is that such uncompleted software has a fair value of zero. Indications that the capitalization requirements in paragraphs 350-40-25-12 through 25-12A are ~~software may~~ no longer met ~~be expected to be completed and place in service~~ include the following: …

Implementation Costs of a Hosting Arrangement That Is a Service Contract

> Impairment

35-11 Impairment shall be recognized and measured in accordance with the provisions of Section 360-10-35 as if the capitalized implementation costs were a long-lived asset. That guidance requires that assets be grouped at the lowest level for which there are identifiable cash flows that are largely independent of the cash flows of other groups of assets. The guidance is applicable, for example, when one of the following events or changes in circumstances occurs related to the **hosting arrangement** that is a service contract indicating that the carrying amount of the related implementation costs may not be recoverable:

a. The hosting arrangement is not expected to provide substantive service potential.

b. A significant change occurs in the extent or manner in which the hosting arrangement is used or is expected to be used.
c. A significant change is made or will be made to the hosting arrangement.

An entity uses the long-lived assets impairment guidance in Topic 360 to determine whether capitalized internal-use software and CCA implementation costs are impaired, and if so, the amount of the impairment loss to recognize. The impairment loss is presented in the same manner in the income statement as an impairment loss recognized for any other long-lived asset. [350-40-35-1, 35-11]

Subtopic 350-40 gives the following example events or changes in circumstances that may indicate the internal-use software, or CCA implementation cost asset is not recoverable. [350-40-35-1, 35-11]

Internal-use software	CCA implementation cost assets
— Software is not expected to provide substantive service potential. — A significant change occurs in the extent or manner in which the software is used or expected to be used. — A significant change is made or will be made to the software program. — Costs of developing or modifying internal-use computer software significantly exceed the amount originally expected to develop or modify the software.	— The CCA is not expected to provide substantive service potential. — A significant change occurs in the extent or manner in which the CCA is used or expected to be used. — A significant change is made or will be made to the CCA.

Even if an internal-use software asset is part of an asset group that is not impaired under Topic 360, it may need to be written down. When the capitalization criteria are no longer met for *uncompleted* internal-use software, that software asset is reported at the lower of its carrying amount and fair value less costs to sell. There is a rebuttable presumption that uncompleted internal-use software has a fair value of zero. [350-40-35-3]

Subtopic 350-40 provides indicators that the software will no longer be completed and placed in service. [350-40-35-3]

Question 6.2.160

Is the liability for an acquired software license to be paid for over time included in the carrying amount of the asset group?

Background: The recoverability test for a held-and-used asset group (Step 1 of the impairment test) excludes:

— financial and nonoperating liabilities from the carrying amount of the asset group; and

— the cash flows attributable to the financial and nonoperating liabilities when determining the undiscounted future expected cash flows of the asset group – i.e. both the interest and principal components of the financial and nonoperating liabilities should be excluded.

In contrast, consistent with Example 1 to Topic 360, an entity includes operating liabilities (e.g. accrued liabilities and accounts payable) in the carrying amount of the asset group and the cash flows used in the recoverability test. [360-10-55-20 – 55-22]

Interpretive response: In general, no. The liability is generally excluded in determining the carrying amount of the asset group, and the interest and principal components of the liability excluded in determining the undiscounted future expected cash flows of the asset group in the recoverability test. An exception may arise if the asset group is a reporting unit and the lowest level of identifiable cash flows includes principal payments on debt. Question 5.3.30 of KPMG Handbook, Impairment of nonfinancial assets, discusses this exception in further detail and its effect on the recoverability test.

We observe the license liability is a financial liability that is generally consistent in character with a finance lease liability under Topic 842, including that it will incur interest expense, and the impairment guidance in Topic 360 excludes finance lease liabilities from Topic 360 asset groups (see Question 5.3.40 of KPMG Handbook, Impairment of nonfinancial assets).

Question 6.2.170

Does the recoverability test include fees for non-license elements of a software licensing arrangement?

Interpretive response: Yes, but how they are included in the Step 1 recoverability test depends on whether the fees are unpaid or prepaid.

Unpaid fees

Unpaid fees for non-license elements like post-contract customer support (PCS) or hosting services are not included in the license liability. Therefore, those fees are deducted from the undiscounted estimated future cash flows of the asset group to which the intangible software license asset belongs.

Prepaid fees

In contrast, an entity may prepay for multiple years of PCS or hosting services – e.g. prepays for a three-year term license with co-terminus PCS and/or hosting services. In this case, there are no unpaid fees and the prepaid asset is included in the carrying amount of the asset group that includes the intangible software license asset.

If an impairment exists, the long-term prepaid asset is in the scope of Topic 360 and will receive an allocation of the impairment loss. [360-10-15-4(a)(4)]

Questions 2.4.20 and 9.3.10 of KPMG Handbook, Impairment of nonfinancial assets, discuss the inclusion of assets in the recoverability test and allocation of impairment losses within an asset group, respectively.

Question 6.2.180

Does the recoverability test include CCA hosting service fees?

Interpretive response: Unpaid hosting service fees should be deducted from the undiscounted estimated future cash flows of the asset group to which a CCA implementation cost asset belongs.

In contrast, if an entity prepays for a multi-year CCA (e.g. prepays for a three-year SaaS subscription), the prepaid asset is included in the carrying amount of the asset group that includes the CCA implementation cost asset.

If an impairment exists, the long-term prepaid asset is in the scope of Topic 360 and will receive an allocation of the impairment loss. [360-10-15-4(a)(4)]

Questions 2.4.20 and 9.3.10 of KPMG Handbook, Impairment of nonfinancial assets, discuss the inclusion of assets in the recoverability test and allocation of impairment losses within an asset group, respectively.

Question 6.2.190

Does the recoverability test include variable payments?

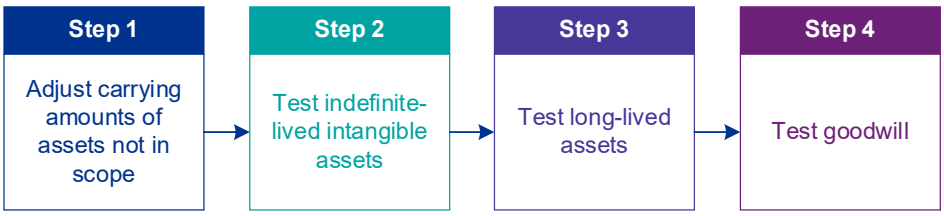
Interpretive response: Yes. An entity should include expected variable payments (e.g. usage or transaction-based fees) when estimating undiscounted future cash flows of the asset group in Step 1 of the Topic 360 impairment test.

The potential range of variability could affect the entity's decision of whether to base its estimates on a single best estimate or probability-weighted cash flows; see Question 7.2.30 in KPMG Handbook, Impairment of nonfinancial assets.

Question 6.2.200

Is internal-use software that is not expected to be completed written down before testing the related asset group for impairment?

Background: As a general principle, an entity performs impairment testing in the following order, based on the nature of the asset. [350-20-35-31 – 35-32, 360-10-35-27]



The entity adjusts carrying amounts for any resulting impairment losses before performing the next test. This sequencing is discussed in further detail in section 4.4 of KPMG Handbook, Impairment of nonfinancial assets.

Interpretive response: Yes. The carrying amount of an asset not in the scope of Subtopic 360-10 is adjusted based on the requirement(s) in other Topics before testing the asset group to which it belongs for impairment. Therefore, an internal-use software asset is written down if required under Subtopic 350-40 before the asset group to which it belongs is tested for impairment under Topic 360. [350-40-35-3, 360-10-35-27]

6.2.50 Internal-use software subsequently marketed

Excerpt from ASC 350-40

35 Subsequent Measurement

General

> Internal-Use Computer Software Subsequently Marketed

35-7 If, after the development of internal-use software is completed, an entity decides to market the software, proceeds received from the license of the computer software, net of direct incremental costs of marketing, such as commissions, software reproduction costs, warranty and service obligations, and installation costs, shall be applied against the carrying amount of that software.

35-8 No profit shall be recognized until aggregate net proceeds from licenses and amortization have reduced the carrying amount of the software to zero. Subsequent proceeds shall be recognized as **revenue** in accordance with Topic 606 on revenue from **contracts** with **customers** or recognized as a gain in accordance with Subtopic 610-20 on derecognition of nonfinancial assets if the contract is not with a customer.

35-9 If, during the development of internal-use software, an entity decides to market the software to others, the entity shall follow the guidance in Subtopic 985-20. Amounts previously capitalized under this Subtopic shall be evaluated at each balance sheet date in accordance with paragraph 985-20-35-4. Capitalized software costs shall be amortized in accordance with paragraphs 985-20-35-1 through 35-2.

35-10 A pattern of deciding to market internal-use software during its development creates a rebuttable presumption that any software developed by that entity is intended for sale, lease, or other marketing, and thus is subject to the guidance in Subtopic 985-20.

An entity may decide to market (e.g. sell or license) internal-use software after development is complete. Proceeds received from the sale or license of the software, net of direct incremental costs of marketing (e.g. commissions,

software reproduction costs, warranty and service obligations, installation costs) reduce the carrying amount of the internal-use software asset. [350-40-35-7]

After the carrying amount is reduced to zero, proceeds are recognized as revenue under Topic 606 (revenue from contracts with customers) or as a gain under Subtopic 610-20 (gains and losses from the derecognition of nonfinancial assets) if the contract is not with a customer. KPMG Handbook, Revenue for software and SaaS, includes guidance on applying Topic 606 and Subtopic 610-20 to software licensing arrangements. [350-40-35-8]

If an entity changes course and decides to externally market software previously intended only for internal use, any recorded software asset becomes subject to Subtopic 985-20 (costs of software to be sold, leased or marketed) from that point (see chapter 5 and section 6.4). [350-40-35-9]

A pattern of marketing internal-use software creates a rebuttable presumption that future software will also be marketed and therefore should not be accounted for under Subtopic 350-40. [350-40-35-10]

Question 6.2.210

What constitutes a 'pattern' of marketing internal-use software?

Interpretive response: It depends. We believe 'pattern' indicates more than a one-off instance, and also likely refers to more than only a couple of instances. However, an entity with more instances may have an established pattern to which the rebuttable presumption would apply. [350-40-35-10]

That said, reasonable judgment should be applied. For example, a smaller number of instances may establish a pattern if they occur within a shorter timeframe. In contrast, if an entity has been developing internal-use software for many years, the same number of instances, spread over that period, may not constitute a pattern.

As illustrated in the following examples, specific facts and circumstances may also affect whether a pattern is determined to exist.

— ABC Corp. decides to market internally developed internal-use software (Application Q) after it acquires Target Co., which has similar internal-use software that will be used in the combined entity's operations post-acquisition. In this example, the decision to externally market Application Q was created by a significant event that made the software redundant.

— DEF Corp. decides there is a market for its internal-use software (Application T) based on the benefits it has generated internally and can demonstrate, and that licensing it would not be competitively disadvantageous. This example might suggest DEF was open (and would be again) to revenue-enhancing opportunities that might arise for any of its internal-use software.

The first example might be given less weight in an analysis of whether there is a pattern (or emerging pattern) of marketing internal-use software externally than the second example.

Question 6.2.220

What evidence is required to overcome a rebuttable presumption that developed software will be marketed externally?

Interpretive response: It depends. As with any rebuttable presumption that exists in US GAAP, it can be overcome based on the facts and circumstances. However, the onus is on the entity to prove why it is overcome (rather than the converse – i.e. why the presumption should be sustained).

The entity's specific facts and circumstances will ultimately determine what and how much evidence is needed to overcome the presumption. However, an important piece of evidence may be demonstrating more recent (than the instances giving rise to the pattern creating the presumption) instances of internal-use software that has not, throughout its economic life (or a significant portion thereof to-date), been marketed externally; these instances may be sufficient to outweigh the pattern that created the presumption.

In general, with or without more recent experiential evidence, we would expect an entity to explain:

— what has changed since the instances creating the pattern occurred – e.g. in terms of the entity's policies/practices, business goals; and/or

— what is different about the new software under development as compared with the software that was marketed externally in the past – e.g. that the software is so specialized to the entity's operations or other software that selling or licensing it externally is not feasible.

Example 6.2.50

Rebuttable presumption exists

Scenario 1: Rebuttable presumption exists and cannot be overcome

DEF Inc. is developing a SaaS product offering for customers in the financial services and insurance industries. Customers of this SaaS offering will not have the contractual right to take possession of the software, and DEF does not presently have a substantive plan to market the software externally. However, in the past DEF has licensed software it developed with the initial intent to sell only through SaaS subscriptions to certain customers in these industries with special security and/or regulatory requirements.

DEF's past practice of licensing other software it developed principally for a SaaS offering creates a rebuttable presumption the new software will also be licensed externally, as well as sold on a SaaS basis.

Because DEF's new product is likely to be of interest to the same customers that have licensed its software products in the past, and in the absence of other evidence to rebut the presumption, DEF cannot overcome the rebuttable presumption created by its past practices.

Therefore, DEF accounts for its software development costs under Subtopic 985-20 even though it does not have a substantive plan to, and ultimately might not, license this new software.

Scenario 2: Rebuttable presumption exists but is overcome

Assume the same facts as in Scenario 1, except the new software product differs from DEF's previously offered products with respect to the types of personal and entity confidential information it will ingest and use, which is a key reason some customers have desired to license DEF's software in the past.

In addition, this is DEF's first new offering in a few years, during which time security and other features of DEF's public cloud hosting service provider have improved and become more widely used, including by previous DEF licensing customers and similar entities.

Lastly, DEF is implementing new business process controls that will require senior vice president of sales (or higher) approval to even enter into negotiations for a license to the new software.

In this scenario, DEF concludes it can overcome the presumption it will license the new software product externally. Consequently, DEF accounts for its software development costs under Subtopic 350-40.

Scenario 3: Rebuttable presumption does not exist

Assume the same facts as in Scenario 1 except DEF has only licensed its software once previously. This was despite meaningful past interest from some entities to license DEF's software, and DEF undertaking market feasibility studies to help it decide whether to license its software as well as offer it on a SaaS basis.

In that case, DEF was required to grant a license to the software underlying three of its seven core SaaS offerings to a single entity as part of a European regulatory order. Entering into those licenses satisfied the European regulator and there is no indication that DEF will be required to license those three offerings to other entities, or the European regulator will require DEF's new software to be licensed to that entity or any other entity.

In this scenario, DEF concludes the past regulatory action does not create a rebuttable presumption about the development of its SaaS offerings in general. Therefore, because there are no other factors indicating the software being developed for this new SaaS offering is not for internal use, DEF accounts for its software development costs under Subtopic 350-40.

Question 6.2.230

What are the accounting effects of a 'one-off' licensing arrangement for the vendor?

Background: An entity previously offered its software to customers *only* under hosting arrangements that do *not* give the customer the contractual right to take possession of the software (see section 2.5). Accordingly, and together with the fact that the entity has never had a substantive plan to license the software (see section 2.4.10), the entity has accounted for its software

development costs under Subtopic 350-40. The entity has capitalized significant development costs thereunder.

Now consider a scenario in which, to satisfy customer regulatory requirements, the entity subsequently enters into a customer contract under which the customer will host the software behind its own firewall on servers it owns or leases. Aside from this, the arrangement substantially mimics the entity's typical SaaS arrangements – i.e.:

— the customer will make subscription payments over the contract period; while

— the entity will manage installation (i.e. to the customer's instance) of the same software updates that it will install to its multi-tenant hosted instance of the software (i.e. to which the entity's SaaS customers obtain access) and provide technical support.

In this scenario, the question has arisen about whether this constitutes a licensing arrangement, rather than a SaaS arrangement; and if this is a licensing arrangement, the accounting consequences.

Interpretive response: First, the background scenario presented *does* constitute a licensing arrangement. Consistent with the response to Question 2.5.50, the customer *has* possession of the software from the arrangement's outset; as such, there is no hosting arrangement (much less one that does not include a license).

Second, it is unimportant that this is the first arrangement in which the entity has licensed this software. The entity is still required to apply the 'internal-use computer software subsequently marketed' guidance discussed in this section. Question 6.2.240 that follows discusses how to apply that guidance to a multiple-element licensing arrangement like that described in the background.

Third, until and unless the requirements in Question 2.4.30 (i.e. pertaining to external-use software becoming internal-use software) are met, any additional software development costs (e.g. developing upgrades or enhancements) are subject to Subtopic 985-20 from that point forward. On the basis, as in the background example, that any such upgrades or enhancements will be provided to the customer licensing the software, they would be considered *external-use* software projects. [350-40-15-2A(b), 25-8; 985-20-55-2]

Paragraphs 350-40-35-7 to 35-8 require the software asset's carrying amount to be reduced by the proceeds of the licensing arrangement (see Question 6.2.240). If the carrying amount is not reduced to zero by those proceeds (and any other licensing arrangements for the software), Question 6.2.250 addresses what guidance to apply to the remaining software asset.

Applying Question 2.4.30

In the context of the background scenario, even if the entity has no plans to license the software to other customers (i.e. the entity believes the licensing arrangement is a 'one-off' instance), we believe the requirements set out in Question 2.4.30 would not be met so long as (1) there is a reasonable possibility of the entity and the customer renewing the licensing arrangement or (2) it is more than remote that upgrades or enhancements to the licensed

software stemming from the entity's development/engineering efforts will be provided to the customer.

Question 6.2.240

What proceeds should reduce the carrying amount of the internal-use software asset?

Background: For purposes of this Question, consider the same scenario outlined in the background to Question 6.2.230. The entity is not only transferring a license to its software, but is also providing PCS (i.e. updates and technical support) and managed professional services. In this case, the question arises about whether the proceeds that will reduce the carrying amount of the capitalized software asset under paragraph 350-40-35-7 should include those from the PCS or managed professional services.

Interpretive response: We believe Subtopic 350-40 is not clear in this regard, and that the relevant guidance in paragraphs 350-40-35-7 and 35-8 can reasonably be interpreted in either of the following ways. Consequently, we would not object to either view (A or B) consistently applied.

- **View A: Only *license* proceeds should reduce the carrying amount of the software asset.** Under this view, "proceeds received from the *license* of the computer software…" and "No profit shall be recognized until aggregate net proceeds from *licenses*…" [emphasis added] should be read literally and in the context of Topic 606's guidance on identifying performance obligations (and regardless of whether the license is a *separate* performance obligation under Topic 606 – see chapter C of KPMG Handbook, Revenue for software and SaaS). Therefore, based on the Topic 606 guidance on allocating transaction price to performance obligations, proceeds attributable to goods or services other than the software license, such as PCS or professional services, are not applied against the carrying amount of the capitalized software. Instead, the entity recognizes these amounts in the same manner as if it had no capitalized software.
- **View B: *All* proceeds under the licensing arrangement should first reduce the carrying amount of the software asset.** Under this view, the proceeds earned from providing PCS or professional services are only possible *because* the related software was developed. Therefore, those proceeds should also, like those attributable to the software license, first be applied against the carrying amount of the software such that no profit from licensing the software *or from those derivative services* occurs until the carrying amount of the software is reduced to zero. Those who support this view believe it is consistent with AcSEC's stated intent for entities to apply a 'cost recovery method' when internal-use software is subsequently sold or licensed. In looking to AcSEC's stated intent in SOP 98-1, we observe that the relevant guidance in paragraphs 350-40-35-7 and 35-8 is unchanged from that in SOP 98-1. [SOP 98-1.89]

Question 6.2.250

What Subtopic governs the accounting for any remaining software asset if its carrying amount is not reduced to zero under paragraphs 350-40-35-7 and 35-8?

Background: For purposes of this Question, assume an entity licenses internal-use software to one or more third parties (i.e. customer or otherwise), but after reducing the software asset's carrying amount in accordance with paragraphs 350-40-35-7 and 35-8 there is a remaining carrying amount. Further assume the software does not meet the requirements in Question 2.4.30 to again be classified as internal-use software.

In that case, the question arises about whether the remaining software asset is subject to the subsequent measurement, presentation and disclosure requirements in Subtopic 985-20, or another Topic (e.g. it remains within the scope of Subtopic 350-40).

Note, any *additional* software development costs incurred (e.g. for upgrades or enhancements) are subject to Subtopic 985-20 until and unless the requirements in Question 2.4.30 (i.e. pertaining to external-use software becoming internal-use software) are met. See Question 6.2.230.

Interpretive response: In general, we believe it is appropriate for any remaining software asset to be accounted for under Subtopic 985-20. We believe it is logical:

- for *both* the existing capitalized software costs and any future upgrade/enhancement development costs related to that same software to be scoped consistently (i.e. both within Subtopic 985-20); and
- to scope the remaining software asset consistent with any other software the entity does or may license externally.

In addition, we observe that the guidance in Section 350-40-35 explicitly states that if, during development, an entity decides to sell or license the software to others, the entity follows the amortization and impairment guidance in Subtopic 985-20 for any software asset already capitalized. [350-40-35-9]

While paragraph 350-40-35-9 only explicitly refers to software under development, if software should be re-scoped (i.e. to Subtopic 985-20) based solely on an entity's *plan* to sell or license it to others, we believe an entity actually *doing so* – i.e. actually selling or licensing the software – should also result in re-scoping, regardless of the software's stage of completion (i.e. still under development or completed). We do not believe the explicit reference to "during development" in paragraph 350-40-35-9 reflects an intention otherwise.

Alternative view

The preceding notwithstanding, an alternative view is that Subtopic 350-40 is not clear about whether any *remaining* software asset becomes subject to Subtopic 985-20.

In this regard, paragraph 350-40-35-9 explicitly refers only to software that is under development, while paragraph 350-40-35-8 appears to contemplate

continued amortization of the software without any reference to a change in scope like that in paragraph 350-40-35-9. Given these points, we would accept this alternative view if it is applied consistently.

6.3 Website development

6.3.10 Before adopting ASU 2025-06

Subtopic 350-50 does not contain subsequent measurement guidance. Therefore, the following applies.

— Costs capitalized from applying Subtopic 350-40 (see sections 4.2.20 – 4.2.50) follow the subsequent measurement guidance applicable to internal-use software assets (see section 6.2)

— Costs capitalized under other Topics follow the subsequent measurement guidance in the Topic (or Subtopic) that applied when deciding which activity costs to capitalize. For example, purchased computer servers capitalized under Topic 360 are accounted for under the subsequent measurement guidance in Section 360-10-35.

6.3.20 After adopting ASU 2025-06**

ASU 2025-06 eliminates Subtopic 350-50. Therefore:

— Website development costs capitalized under Subtopic 350-40 follow the same subsequent measurement guidance as all other costs capitalized under that Subtopic.

— Costs capitalized under other Topics (e.g. equipment) follow the subsequent measurement guidance in the Topic (or Subtopic) that applied when deciding to capitalize the costs. For example, purchased computer servers capitalized under Topic 360 are accounted for under the subsequent measurement guidance in Section 360-10-35.

6.4 Costs of software to be sold, leased or marketed

Question 6.4.05

Is external-use software acquired in a business combination or acquisition of a not-for-profit entity accounted for under Subtopic 985-20 or Subtopic 350-30 post-acquisition?

Interpretive response: Based on discussions with the FASB staff, we believe it is acceptable, as an accounting policy election, to account for acquired external-use software post-acquisition under either Subtopic 985-20 (Approach A) or

Subtopic 350-30 (Approach B). US GAAP is not sufficiently clear to preclude either approach.

We have observed both approaches being applied in practice. Often, it is entities with significant capitalized internally developed external-use software that follow Approach A, while entities whose development processes do not result in material capitalized external-use software generally appear to follow Approach B.

6.4.10 Amortization

Excerpt from ASC 985-20

35 Subsequent Measurement

General

> Amortization of Capitalized Software Costs

35-1 Capitalized software costs shall be amortized on a product-by-product basis. The annual amortization shall be the greater of the amounts computed using the following:

a. The ratio that current gross revenues for a product bear to the total of current and anticipated future gross revenues for that product
b. The straight-line method over the remaining estimated economic life of the product including the period being reported on.

35-2 Because a net realizable value test, which considers future revenues and costs, must be applied to capitalized costs (see paragraph 985-20-35-4), amortization shall be based on estimated future revenues. In recognition of the uncertainties involved in estimating revenue, amortization shall not be less than straight-line amortization over the product's remaining estimated economic life.

35-3 Amortization shall start when the product is available for general release to customers.

55 Implementation Guidance and Illustrations

General

> Implementation Guidance

• > Amortization

• • > Changes in Estimates of Future Revenues or Remaining Economic Life

55-15 Estimates of future revenues or the remaining economic life for a product may change over the period in which the software product is being amortized. Amortization for any asset is based on estimates of future events, and software is no exception. The most recent information should be used to determine if changes to estimates should be made.

• • > Straight-Line Amortization

55-16 Paragraph 985-20-35-1 indicates that straight-line amortization of a software product is computed over the remaining estimated economic life of the product. As such, the unamortized cost of the product should be divided by its remaining life, including the current year.

• > Product Enhancements

• • > Useful Life of a Product Enhancement

55-19 The estimated useful life of a product enhancement is the estimated life of the enhancement. It is not the remaining life of the original product nor is it the remaining life of the original product for any costs of the original product included in the enhancement and the estimated life of the enhancement for all other costs. All costs of a product enhancement, including any costs carried over or allocated from the original product, should be amortized over the enhancement's estimated useful life.

Subtopic 985-20 requires the amortization of capitalized software production costs on a product-by-product basis. Amortization commences when the software product is available for general release to customers. [985-20-35-3]

Once amortization commences, the *annual* amount of amortization is the greater of: [985-20-35-1]

— the ratio of (1) gross current annual period revenues for the software product as compared to (2) gross current annual period revenues for the software product *plus* total expected future revenue for the software product (**ratio method**); and

— the amortization that would result for the period from amortizing the software costs on a straight-line basis from the beginning of the annual period over the software product's remaining economic life (**straight-line method**).

External-use software acquired in a business combination may be accounted for post-acquisition, including with respect to amortization, in accordance with either Subtopic 985-20 or Subtopic 350-30 (general intangibles other than goodwill). See Question 6.4.05.

Question 6.4.10

How is amortization recognized during interim periods?

Background: In general, annual 'ratio method' amortization cannot be calculated until the end of the entity's fiscal year. Before then, actual gross product revenues for the year are not known, and facts and circumstances may arise that affect estimated future gross product revenues.

Therefore, the entity may not know which method, straight-line or ratio, will produce greater amortization for the year. Even if it is virtually certain to be the ratio method, the exact amount will not be known.

As a result, the question arises about how an entity should record amortization in interim periods during its fiscal year.

Interpretive response: Entities reporting interim financial information on an interim basis under Topic 270 (interim reporting) frequently make estimates in assigning costs and expenses to interim periods so that interim period results reflect anticipated annual results. For example, entities frequently estimate certain employee bonuses, and accrue a proportion thereof during interim periods, even if the bonus will not be owed to the employee if they terminate employment or specified metrics are not met for the year. [270-10-45-4(b)]

An entity should make estimates about the amount of amortization it will recognize for the fiscal year, including the amortization method that will apply (straight-line or ratio), to record interim period amortization. The entity's intent with such estimates should not be biased toward recognizing more or less amortization in an interim period than can be rationally attributed thereto.

The entity should not restate previously recorded amortization amounts when its annual amortization estimates change. For example, first quarter amortization should not be adjusted for changes in estimates made in the second quarter. Additional or reduced first quarter amortization necessary to true-up first half (first and second quarters) amortization to revised estimates of annual amortization should be recognized entirely in the second quarter. [270-10-45-14]

Question 6.4.20

Is a change between straight-line and ratio method amortization a change in accounting principle?

Interpretive response: No. The method used to calculate annual amortization of capitalized software costs under Subtopic 985-20 is not an accounting principle under Topic 250. The method used to calculate amortization each year is dictated by the result each one produces for the year, which can change multiple times during a software product's estimated economic life. [985-20-35-1]

Because any change in amortization method is required rather than voluntary, an entity is not required to assess the preferability of such a change under Topic 250.

Question 6.4.30

Does the straight-line amortization method use the original or current carrying amount of the capitalized software costs?

Background: Assume an entity has $100 in capitalized software costs for a particular software product that has a four-year economic life. The entity recognizes $34 in Year 1 amortization based on the ratio method (versus $25 that would have resulted from the straight-line method).

In this example, the question arises about whether the Year 2 straight-line amortization equals $25 ($100 original carrying amount ÷ 4-year economic life) or $22 ($66 beginning of Year 2 carrying amount ÷ 3-year remaining economic life).

Interpretive response: The carrying amount of the software cost asset resets each year for purposes of calculating straight-line method amortization. Therefore, straight-line method amortization for Year 2 would be $22, rather than $25.

This is because the straight-line method description refers to the *remaining* economic life of the product. [985-20-35-1(b)]

Example 6.4.10

Amortization of capitalized software development costs – no change in estimates

ABC Corp. developed an application (Product G) that will be licensed to customers. Total capitalized software costs were $2,000,000. The application was available for general release on January 1, Year 1.

On January 1, Year 1, ABC anticipated $30,000,000 in total gross revenues over the five-year estimated economic life of Product G as follows.

	Year 1	Year 2	Year 3	Year 4	Year 5
Anticipated revenues	$3,000,000	$4,590,000	$8,964,000	$7,395,300	$6,050,700

ABC accounts for the capitalized software costs from January 1, Year 1, as follows. This example assumes no changes to gross revenue estimates or the economic life of Product G, and that actual revenues equal gross revenue estimates.

Year	Beg. balance	Actual revenues	Ratio	Ratio method[1]	Straight-line method[2]	Amort. recorded[3]
1	$2,000,000	$3,000,000	10%	$200,000	$400,000	$400,000
2	1,600,000	4,590,000	17%	272,000	400,000	400,000
3	1,200,000	8,964,000	40%	480,000	400,000	480,000
4	720,000	7,395,300	55%	396,000	360,000	396,000
5	324,000	6,050,700	100%	324,000	324,000	324,000

Notes:

1. Ratio method amortization: (Actual revenues ÷ Actual + remaining forecasted revenue) × Beg. balance.
2. Straight-line amortization: Beg. balance ÷ remaining economic life.
3. Annual amortization is the greater of the amount calculated under the ratio method or straight-line method. The beginning balance is adjusted by this amount for the following year.

Question 6.4.40

How does an entity account for changes to economic lives or anticipated revenues for software products?

Background: Entities should use the best available information to determine the amortization amount in each period. Using the best available information may result in changes to estimates, including changes to anticipated revenues and economic lives. [985-20-55-15]

Interpretive response: A change in accounting estimate is accounted for prospectively – i.e. in the period of change, and in future periods (if applicable). [250-10-45-17 – 45-18]

In most instances, changes to estimates used in the subsequent accounting for capitalized software development costs will impact the period in which the change occurred and future periods. Entities should not modify prior period amortization amounts.

Entities should consider appropriate disclosures related to the change in accounting estimate as required under Topic 250. See Question 3.4.50 in KPMG Handbook, Accounting changes and error corrections. [250-10-50-4 – 50-5]

Example 6.4.20

Amortization of capitalized software costs – change in estimated gross revenues

ABC Corp. developed a mobile application (Product U) that will be licensed to customers. Total capitalized software costs were $2,500,000. The mobile application was available for general release on January 1, Year 1.

On January 1, Year 1, ABC anticipated $20,000,000 in total gross revenues over the five-year estimated economic life of Product U as follows. The total economic life of Product U remains five years throughout this example.

	Year 1	Year 2	Year 3	Year 4	Year 5
Anticipated revenues	$3,000,000	$5,000,000	$5,000,000	$4,000,000	$3,000,000

At the end of Year 2, ABC revises its estimate of total gross revenues to $14,000,000 due to decreased customer demand, evidenced by Year 2 Product U revenues lower than initially forecast.

	Year 1 (Actual)	Year 2 (Actual)	Year 3 (Forecast)	Year 4 (Forecast)	Year 5 (Forecast)
Actual and anticipated revenues	$3,000,000	$3,960,000	$3,520,000	$2,112,000	$1,408,000

ABC accounts for the capitalized software costs from January 1, Year 1, as follows.

Year	Beg. balance[1]	Actual revenues	Ratio	Ratio method	Straight-line method	Amort. recorded[7]
1	$2,500,000	$3,000,000	15%	$375,000[2]	$500,000[3]	$500,000
2	2,000,000	3,960,000	36%	720,000[4]	500,000[5]	720,000
3	1,280,000	3520,000	50%	640,000[6]	426,667[5]	640,000
4	640,000	2,112,000	60%	384,000[6]	320,000[5]	384,000
5	256,000	1,408,000	100%	256,000[6]	256,000[5]	256,000

Notes:

1. The beginning balance for Years 2 – 5 equals the beginning balance for prior year less amortization recorded for prior year.
2. Actual revenues ($3,000,000) ÷ (Actual revenues ($3,000,000) + remaining forecasted revenues ($17,000,000)) × Beg. balance ($2,500,000).
3. Beg. balance for the year ($2,500,000) ÷ remaining economic life (5 years).
4. Actual revenues ($3,960,000) ÷ (Actual revenues ($3,960,000) + remaining forecasted revenues as revised ($7,040,000)) × Beg. balance ($2,000,000).
5. Years 2 – 5 straight-line method amortization calculated in same manner as Year 1 (see Note 3).
6. Years 3 – 5 ratio method amortization calculated in same manner as Year 2 (see Note 4).
7. The annual amortization expense is the greater of the amount calculated under the ratio or straight-line method.

When ABC changes its revenue estimates for Product U at the end of Year 2, it does not adjust Year 1 amortization expense. Instead, ABC accounts for the change in accounting estimate (i.e. Year 2 revenue actuals and the change in estimated future gross revenues) prospectively.

If ABC has issued interim financial statements during Year 2 before the change in accounting estimate (e.g. if Product U revenues in XYZ's first and second quarters were still trending consistent with the original forecast, so no change in estimate occurred during those periods), it does not adjust the amortization expense recorded in those interim periods.

Example 6.4.30

Amortization of capitalized software costs – change in estimated gross revenues and economic life

Assume the same facts as in Example 6.4.20, except the same factors that gave rise to the decreased customer demand for Product U also cause ABC Corp. to adjust the total economic life of Product U to four years (from five). The following table shows actual and estimated gross revenues for the revised economic life from the end of Year 2.

	Year 1 (Actual)	Year 2 (Actual)	Year 3 (Forecast)	Year 4 (Forecast)
Actual and anticipated revenues	$3,000,000	$3,960,000	$3,520,000	$3,520,000

ABC accounts for the capitalized software costs from January 1, Year 1, as follows.

Yr.	Beg. balance[1]	Actual revenues	Ratio	Ratio method	Straight-line method	Amort. recorded[8]
1	$2,500,000	$3,000,000	15%	$375,000[2]	$500,000[3]	$500,000
2	2,000,000	3,960,000	36%	720,000[4]	666,667[5]	720,000
3	1,280,000	3,520,000	50%	640,000[6]	640,000[7]	640,000
4	640,000	3,520,000	100%	640,000[6]	640,000[7]	640,000

Notes:

1. The beginning balance for Years 2 – 4 equals the beginning balance for prior year less amortization recorded for prior year.
2. Actual revenues ($3,000,000) ÷ Actual revenues ($3,000,000) + remaining forecasted revenues ($17,000,000)) × Beg. balance ($2,500,000).
3. Beg. balance ($2,500,000) ÷ remaining economic life (5 years).
4. Actual revenues ($3,960,000) ÷ (Actual revenues ($3,960,000) + remaining forecasted revenues as revised ($7,040,000)) × Beg. balance ($2,000,000).
5. Beg. balance ($2,000,000) ÷ remaining economic life as revised (3 years).
6. Years 3 – 4 ratio method amortization calculated in same manner as Year 2 (see Note 4).
7. Years 3 – 4 straight-line method amortization calculated in same manner as Year 2 (see Note 5).
8. The annual amortization is the greater of the amount calculated under the ratio or straight-line method.

When ABC changes its revenue and economic life estimates for Product U at the end of Year 2, it does not adjust Year 1 amortization expense. Instead, ABC accounts for the change in accounting estimates prospectively.

If ABC has issued interim financial statements during Year 2 before the change in accounting estimate (e.g. if Product U revenues in XYZ's first and second quarters were still trending consistent with the original forecast, so no change in estimate occurred during those periods), it does not adjust the amortization expense recorded in those interim periods.

Product enhancements

Section 5.3.50 discusses the accounting for capitalized production costs of a product enhancement depends on whether both the original product and the enhanced product will be sold (licensed) to customers. [985-20-55-18]

— If so, the unamortized production costs of the original product are allocated between an original product asset and an asset for the separately marketed enhanced product (see Question 5.3.90). [985-20-55-19]

  – The unamortized production costs allocated to the original product continue to be amortized over its remaining economic life.

  – The capitalized production costs of the enhanced product (i.e. those capitalized for the enhancement plus unamortized original product costs allocated to the enhanced product) are amortized over the economic life of the enhanced product.

— If not (i.e. only the enhanced product will be sold to customers), the unamortized production costs of the original product are combined with the capitalized production costs of the enhancement and amortized over the economic life of the enhanced product (see Question 5.3.80).

Amortization of an original or an enhanced product, or a product enhancement continues to follow the 'greater of' model outlined above. [985-20-35-1]

Example 6.4.40
Amortization – software product enhancements

ABC Corp. develops payroll software for large business entities. On January 1, Year 1, it released an upgraded version of its Product P solution. The upgraded version includes enhanced features and functionality that meet the definition of a product enhancement because ABC expects the upgraded version to both extend the economic life and the marketability of Product P.

ABC capitalized production costs incurred after the technological feasibility of the enhanced Product P software was established. The following additional factors are relevant.

— ABC originally capitalized $5,000,000 in Product P production costs. As of January 1, Year 1, $2,000,000 is unamortized.

— ABC has capitalized $3,000,000 in production costs for the Product P enhancements.

— The economic life of Product P was originally five years; this was not reassessed through January 1, Year 1. The remaining economic life of Product P without the product enhancements (original Product P) is two years at January 1, Year 1.

— The remaining economic life of enhanced Product P is five years from January 1, Year 1.

— The estimated economic life of the enhanced features of Product P is three years from January 1, Year 1.

Scenario 1: Original Product P will continue to be marketed

Original Product P will continue to be marketed to customers at a lower price than enhanced Product P (e.g. to customers that do not need the enhanced features). Because original Product P will continue to be licensed to customers, ABC, using a percentage of shared code allocation methodology, allocated 50%

of the software development costs related to original Product P to enhanced Product P (see Question 5.3.90).

The remaining original Product P unamortized costs ($1,000,000, equal to 50% of the unamortized original Product P costs) will continue to be amortized over the remaining economic life of original Product P (two years). The capitalized product enhancement costs ($3,000,000) plus the original Product P costs allocated to enhanced Product P ($1,000,000, equal to the other 50% of the unamortized original Product P costs) will be amortized over the remaining economic life of the enhanced product (five years).

ABC will amortize original Product P and enhanced Product P in the same manner as it would any other capitalized software costs subject to Subtopic 985-20 (see Examples 6.4.10 – 6.4.30).

Scenario 2: Original Product P will no longer be marketed

Original Product P will no longer be marketed to customers upon release of enhanced Product P.

The unamortized production costs of $2,000,000 are combined with those capitalized for the product enhancements (see Question 5.3.80). The combined $5,000,000 in capitalized production costs will be amortized over the remaining economic life of the enhanced Product P (five years).

ABC will, in effect, amortize the enhanced Product P as if it was a new software product from the date it is available for general release (see Examples 6.4.10 – 6.4.30).

6.4.20 Abandonment

Unlike Subtopic 350-40 applicable to internal-use software and CCA implementation cost assets, Subtopic 985-20 does not include guidance about abandoning a capitalized external-use software asset.

However, if an entity decides to cease use of external-use software (e.g. cease selling licenses to the software product), it is likely one or both of the following are affected:

— the economic life of the software product; and/or
— estimated future gross revenues of the software product.

It is possible that one or both of those preceded (and influenced) the entity's decision. However, even if not, it is likely one or both of those are reduced by the decision.

Under the Subtopic 985-20 amortization model (see section 6.4.10), shortening the economic life of the product or reducing future estimated gross revenues of the product will have the effect, similar to abandonment accounting under Subtopic 350-40, of prospectively accelerating amortization of the external-use software asset to the cease-use date.

6.4.30 Impairment

Excerpt from ASC 985-20

20 Glossary

Customer Support

Services performed by an entity to assist customers in their use of software products. Those services include any installation assistance, training classes, telephone question and answer services, newsletters, on-site visits, and software or data modifications.

Maintenance

Activities undertaken after the product is available for general release to customers to correct errors or keep the product updated with current information. Those activities include routine changes and additions.

35 Subsequent Measurement

General

> Net Realizable Value of Capitalized Software Costs

35-4 At each balance sheet date, the unamortized capitalized costs of a computer software product shall be compared to the net realizable value of that product. The amount by which the unamortized capitalized costs of a computer software product exceed the net realizable value of that asset shall be written off. The net realizable value is the estimated future gross revenues from that product reduced by the estimated future costs of completing and disposing of that product, including the costs of performing **maintenance** and **customer support** required to satisfy the entity's responsibility set forth at the time of sale. The reduced amount of capitalized computer software costs that have been written down to net realizable value at the close of an annual fiscal period shall be considered to be the cost for subsequent accounting purposes, and the amount of the write-down shall not be subsequently restored.

The following steps summarize the guidance in Subtopic 985-20.

Step 1	Determine the NRV of the software product
Step 2	Compare the NRV of the software product to the unamortized production costs
Step 3	Adjust the carrying amount of the unamortized production costs if the NRV is lower

Capitalized external-use software production costs are measured at the lower of unamortized cost and NRV on a product-by-product basis at each reporting date. [985-20-35-4]

The NRV of a software product is determined as follows.

NRV test impairments are not subsequently written-up for any reason, other than as discussed in Question 6.4.60. [985-20-35-4]

External-use software acquired in a business combination may be accounted for post-acquisition (including with respect to impairment) under either Subtopic 985-20 or Subtopic 350-30 (general intangibles other than goodwill). See Question 6.4.05.

Question 6.4.50

Does an entity perform the NRV test at both interim and annual reporting dates?

Interpretive response: Yes. The guidance in Subtopic 985-20 explicitly states the NRV test is to be performed 'at each balance sheet date'. [985-20-35-4]

Question 6.4.60

Are NRV impairments recoverable within a fiscal year?

Background: The reduced carrying amount (after NRV testing) of an external-use software asset 'at the close of an annual fiscal period' is its cost for subsequent accounting purposes; the writedown to NRV cannot be reversed. [985-20-35-4]

However, the reference to the annual reporting date gives rise to the question of whether an NRV writedown in an interim period may be reversed in a subsequent interim period in the same fiscal year.

Interpretive response: Yes. We believe NRV writedowns taken in an interim period may be reversed to the extent conditions improve before the end of the entity's fiscal year. For example, if an entity writes down an external-use software asset by $100 in Q1, the entity may restore up to that amount if circumstances change before the end of the fiscal year. This is because the guidance refers to the amount of the writedown of the asset at the close of an *annual* fiscal period being ineligible for recovery. [985-20-35-4]

In addition, the basis for conclusions to FAS 86 indicates the FASB considered the then-existing guidance on motion picture film costs when deciding on the NRV test for external-use software costs. The AICPA's Motion Picture Industry Accounting Guide (1973), Accounting for Motion Picture Films, in the context of assessing the NRV of a film, stated that adjustments within a fiscal year

"should be reflected on a cumulative basis from the beginning of the fiscal year." [FAS 86.47]

When a recovery during the fiscal year occurs, the amount thereof is recognized in the current interim period; the effect of the recovery on amortization of the asset is recognized prospectively. Prior interim period results are not adjusted. [270-10-45-14]

Question 6.4.70

Does an entity discount the estimated future gross revenues or costs used in the NRV test?

Interpretive response: No. The NRV test is performed on an undiscounted basis. Subtopic 985-20 does not discuss the discounting of estimated gross revenue or cost projections used in the NRV test. However, the FASB intended for the NRV test to be consistent with that applied to inventory, and the NRV of inventory under Subtopic 330-10 uses undiscounted estimated selling prices and predictable costs. [FAS 86.48, 330-10-35-1B, 330-10 Glossary]

Question 6.4.80

Are non-license revenues included when calculating a software product's NRV?

Background: Entities generate non-license, software-related revenue from most of the software products they license externally. For example, entities usually earn PCS revenue and may also earn revenues from providing product-related professional services (e.g. implementation, training) and/or hosting the software for customers. In addition, entities may also sell access to a software product they license on a SaaS basis.

The question arises about whether estimated future gross revenues used to calculate the NRV of a software product should include only those related to licensing the software, or also include software-related revenue that could not be earned by the entity if the software product were not developed and, in the case of PCS and license-related professional services, licensed to customers.

Interpretive response: Subtopic 985-20 is not specific in this respect, and in any event, given its issuance in 1985, did not contemplate scenarios such as entities providing hosting services, selling SaaS subscriptions or accounting for software license and software-related revenues as combined units of account under Topic 606 (i.e. single, combined performance obligations – see chapter C of KPMG Handbook, Revenue for software and SaaS). Therefore, there is likely some diversity in practice.

Acknowledging that, we believe it is likely most appropriate to include software license and software-related revenues, including SaaS subscription revenues, in the NRV calculation for a software product when those revenues would not arise without the software product having been developed.

With respect to SaaS subscription revenues, excluding those revenues could give rise to an uneconomic NRV writedown, resulting only because the software product is also licensed to customers. This is because if the software product was only sold on a SaaS basis, and therefore was subject to Subtopic 350-40 instead of Subtopic 985-20, those SaaS revenues would be included when assessing the capitalized software costs for impairment (see section 6.2.40). Thus, a software product with SaaS and license revenues could be impaired under the Subtopic 985-20 NRV test when the same product, having *only* the same SaaS revenue (i.e. less *total* forecasted revenues), might not be impaired under Subtopic 350-40.

Further, excluding revenues for software-related services that are inseparable (i.e. not distinct) from product licenses under Topic 606 would require entities to separately identify revenues for purposes of the NRV test that Topic 606 concludes are not separately identifiable.

Related costs

An NRV calculation for a software product that includes software-related revenues should also include all costs expected to be incurred to fulfill the performance obligations underlying those estimated revenues. We believe Subtopic 340-40 provides relevant guidance on what costs should be included; see chapter H of KPMG Handbook, Revenue for software and SaaS. [340-40-25-7 – 25-8]

Question 6.4.90

Are projected sales- and usage-based revenues included in a software product's NRV?

Interpretive response: Yes. We believe gross product revenues include all projected revenues directly attributable to the software product, including sales- and usage-based license fees.

Question 6.4.100

Are estimated hardware revenues included in a software product's NRV if the software is firmware?

Background: As discussed in section 5.2.30, firmware refers to software that is integral to a larger product or process. Firmware may be sold *only* embedded within another product or it may also be licensed separately (e.g. to other manufacturers for use in their products).

Interpretive response: If the NRV of developed firmware based on separate sales (license and software-related revenues – see Question 6.4.80) is less than its carrying amount, we believe it would be appropriate to consider forecasted sales of the hardware product(s) in which the firmware is embedded, net of the reasonably predictable costs of those sales.

Example 6.4.50
NRV test for capitalized software production costs

ABC Corp. has capitalized software production costs for two software products (A and B). As of the reporting date, ABC assesses the NRV for each product as follows.

	Product A	Product B
Estimated future gross revenues[1]	$20,000,000	$8,000,000
Estimated future costs to complete the software product[2]	1,500,000	0
Estimated future costs to fulfill[3]	8,000,000	4,500,000
Unamortized costs	1,500,000	4,000,000
NRV writedown[4]	**0**	**500,000**

Notes:

1. Includes revenues from licensing the software, PCS, and professional services to implement the software (see Question 6.4.80).
2. These include the remaining expected production costs to be incurred before the product is available for general release.
3. Includes costs to fulfill the forecasted revenues in Note 1 (see Question 6.4.80).
4. Product A's NRV exceeds its carrying amount by $9 million ($20 million – $1.5 million – $8 million – $1.5 million); therefore, there is no NRV writedown. In contrast, Product B's carrying amount exceeds its NRV by $500,000 ($8 million – $4.5 million – $4 million); Product B's carrying amount is written down accordingly.

Assuming the assessment date is the end of ABC's fiscal year, instead of an interim reporting date (see Question 6.4.60), the writedown of Product B cannot be reversed in the future. This applies even if there is a subsequent change to ABC's estimates of future gross revenues or related costs to fulfill the performance obligations underlying those gross revenues.

7. Presentation

Detailed contents

New item added in this edition: **
Item significantly updated in this edition: #

7.1 How the standards work

Internal-use software and cloud computing arrangements

Subtopic 350-40 provides only limited guidance about financial statement presentation. It specifies only the:

— balance sheet presentation of acquired licenses to internal-use software (i.e. as intangible assets); and
— balance sheet, income statement and statement of cash flows presentation of cloud computing arrangement (CCA) implementation costs.

Consequently, other Topics govern other matters related to an entity's financial statement presentation – e.g. Topic 360 for presentation of impairment losses on internal-use software assets. [360-10-45-5]

Website development#

Subtopic 350-50 does not contain presentation guidance, and is eliminated by ASU 2025-06. Therefore, both before and after adopting ASU 2025-06, we believe the financial statement presentation related to website development costs is governed by other Topics.

Costs of software to be sold, leased or marketed

The financial statement presentation requirements of Subtopic 985-20 include:

— capitalized software production costs are presented as an amortizable intangible asset;
— capitalized production costs amortization is presented in cost of sales (or similar); and
— the presentation requirements of Topic 350 (goodwill and intangible assets) apply for capitalized production costs.

Like Subtopics 350-40 and 350-50, where Subtopic 985-20 does not provide specific requirements, general financial statement presentation principles and Topics (e.g. Topic 230 on the statement of cash flows) apply.

7.2 Internal-use software and cloud computing arrangements

7.2.10 Internal-use software

Excerpt from ASC 350-40

25 Recognition

> Capitalization of Cost

25-17 Entities often license internal-use software from third parties. A software license within the scope of this Subtopic (see paragraphs 350-40-15-1 through 15-4C) shall be accounted for as the acquisition of an intangible asset and the incurrence of a liability (that is, to the extent that all or a portion of the software licensing fees are not paid on or before the acquisition date of the license) by the licensee. The intangible asset acquired shall be recognized and measured in accordance with paragraphs 350-30-25-1 and 350-30-30-1, respectively.

Subtopic 350-40 does not prescribe financial statement presentation for internally developed or acquired internal-use software. However, it does prescribe an acquired internal-use software license is an intangible asset. [350-40-25-17]

Observation
Licensed internal-use software

The basis for conclusions to ASU 2016-19, which added paragraph 350-40-25-17 to Subtopic 350-40 in 2016, is clear the Board intentionally decided to require entities to account for acquired internal-use software licenses as acquired intangible assets. [ASU 2016-19.BC5]

This accounting and presentation for acquired internal-use software licenses may reflect a change for some entities from their historical accounting and reporting pre-ASU 2016-19.

Observation**
Cash paid for capitalized software costs

In the proposed ASU that eventually became ASU 2025-06, the FASB suggested requiring entities to separately present cash paid for capitalized internal-use software in the investing section of the statement of cash flows. However, the FASB ultimately decided not to carry forward this requirement to the final ASU based on stakeholder feedback.

Given that some entities may be familiar with the proposed ASU, we believe it is important to clarify that this aspect of the proposed ASU was not ultimately adopted. [ASU 2025-06.BC96, BC100, BC102]

Question 7.2.10

Should the initial recognition of an internal-use software license and license fees liability be disclosed as a noncash transaction?

Interpretive response: Yes. See Question 8.2.40 in KPMG Handbook, Statement of cash flows.

Question 7.2.15**

How should an entity classify its non-capitalizable internal-use software development costs in the income statement?

Interpretive response: There is no explicit guidance in US GAAP regarding the income statement classification of non-capitalizable development costs for internal use software. Specifically, Subtopic 350-40 does not address this issue, unlike Subtopic 985-20.

We believe entities should consider the intended use of the internal-use software when determining the appropriate income statement classification.

Revenue producing software (e.g. SaaS platforms)

The Basis for Conclusions to ASU 2025-06 observes that entities often classify and disclose non-capitalized costs of internal-use software they will sell customers access to via CCAs as R&D expenses. In this regard, we believe many of those entities do so by analogy to Subtopic 985-20, which requires costs incurred before technological feasibility is established for external-use software to be expensed as incurred and classified as R&D expense. [985-20-25-1, ASU 2025-06.BC47]

We believe this presentation is reasonable given the nature of these costs is generally consistent with those incurred by external-use software vendors; in addition, we observe that the FASB did not express concern, either in the basis for conclusions to ASU 2025-06 or in the public deliberations preceding the ASU's issuance, about the practice described in the preceding paragraph.

That said, AcSEC and the FASB, each, intentionally did not specify income statement classification of non-capitalizable internal-use software development costs (i.e. in SOP 98-1 and ASU 2025-06, respectively). Therefore, there may be other acceptable income statement classifications; in that case, we encourage entities to discuss their intended classification(s) with their auditors or other accounting advisors. [SOP 98-1.54, ASU 2025-06.BC47]

Non-revenue producing software (e.g. ERP systems)

As noted in the preceding paragraph, neither AcSEC, nor the FASB, chose to prescribe an income statement presentation for non-capitalizable internal-use software development costs.

In practice, we believe most entities classify non-capitalizable costs of non-revenue generating internal-use software based on the functional line-item the software will support. For example:

- HR system costs may be included in a G&A line-item; and
- Customer relationship management (CRM) system costs may be included in a sales and marketing line-item.

We believe this a reasonable and appropriate practice, consistent with both:

- AcSEC's observation in SOP 98-1 that it would generally expect costs for software developed for selling, administrative or general business purposes (such as ERP systems) *not* to be classified as R&D expenses; and [SOP 98-1.52]
- the FASB's decision *not* to prescribe R&D classification for these costs because the costs may relate to operational or administrative software, for which R&D expense classification would generally be a change in practice and inconsistent with stakeholders' views of those costs. [ASU 2025-06.BC47]

Like for revenue producing software, because there is not explicit guidance around the income statement classification of non-revenue producing internal-use software development costs, there may be alternative approaches and some diversity in practice. If an entity wants to classify such costs on a basis other than that described herein, we encourage them to discuss their intended classification with their auditors or other accounting advisors.

Section 4.6 of KPMG Handbook, Financial statement presentation, provides guidance on income statement expense classification in general.

Question 7.2.16**

Is amortization of internal-use software 'depreciation' or 'amortization'?

Interpretive response: Although Subtopic 350-40 refers to 'amortization' of internal-use software costs, rather than 'depreciation', we believe there is diversity in practice about how entities classify internal-use software cost amortization; this was acknowledged by the FASB in the basis for conclusions to ASU 2024-03, *Income Statement—Reporting Comprehensive Income—Expense Disaggregation Disclosures* (Subtopic 220-40). [ASU 2024-03.BC100]

We believe this diversity results from diversity in how entities classify internal-use software assets on the balance sheet. As observed earlier in this section, Subtopic 350-40 does not prescribe financial statement presentation for internally developed or acquired internal-use software; some entities classify such software as PP&E, others as an intangible asset. However, an acquired internal-use software license *must* be presented as an intangible asset. [350-40-25-17]

In our view, entities' income statement classification as amortization or depreciation should align with its balance sheet presentation of the internal-use software asset as PP&E or an intangible asset. If the internal-use software asset is presented as PP&E, the amortization thereof should be presented and disclosed as depreciation; if the internal-use software asset is presented as an intangible asset, the amortization thereof should be presented and disclosed as amortization.

7.2.20 Cloud computing arrangements

Excerpt from ASC 350-40

45 Other Presentation Matters

Implementation Costs of a Hosting Arrangement That Is a Service Contract

> Amortization

45-1 An entity shall present the amortization of implementation costs described in paragraph 350-40-35-13 in the same line item in the statement of income as the expense for fees for the associated **hosting arrangement**.

> Statement of Financial Position

45-2 An entity shall present the capitalized implementation costs described in paragraph 350-40-25-18 in the same line item in the statement of financial position that a prepayment of the fees for the associated hosting arrangement would be presented.

> Statement of Cash Flows

45-3 An entity shall classify the cash flows from capitalized implementation costs described in paragraph 350-40-25-18 in the same manner as the cash flows for the fees for the associated hosting arrangement.

Subtopic 350-40 prescribes the financial statement presentation for CCA implementation costs.

Financial statement presentation – CCA implementation costs	
Balance sheet	An entity presents capitalized CCA implementation costs in the same line item as a prepayment of the hosting service fees (see Questions 7.2.20 and 7.2.30). [350-40-45-2]
Income statement	The amortization of capitalized CCA implementation costs is presented in the same line item as the hosting service fees paid to the cloud service provider (see Question 7.2.40). [350-40-45-1]
Cash flow statement	Cash payments for CCA implementation costs are classified consistent with how the hosting service fees are classified (see Question 7.2.50). [350-40-45-3]

Observation

Consistency between CCA hosting service fees and implementation costs

The financial statement presentation requirements reflect the preference of most EITF members to ensure a consistent link between the capitalized implementation costs and the fees associated with the CCA itself. [ASU 2018-15.BC12]

Subtopic 350-40 results in consistent presentation of CCA implementation costs and CCA hosting service fees throughout the financial statements.

Observation

CCA implementation cost amortization and EBITDA

EBITDA is a non-GAAP measure, therefore Subtopic 350-40 does not address whether amortization of capitalized CCA implementation costs should be excluded in the calculation of EBITDA.

However, the FASB staff explained 'amortization' is used in Subtopic 350-40 for recognizing capitalized CCA implementation costs as expense in the same general sense it is used in Topic 340 (other assets and deferred costs) and elsewhere in US GAAP to describe the expense attribution of non-tangible assets. [EITF Issue Summary No. 1, Supplement No. 4, May 24, 2018]

During the public EITF deliberations of ASU 2018-15, the FASB staff further indicated the term 'amortization' for the recognition of the capitalized costs is not intended to suggest it is appropriate to increase EBITDA by excluding the amortization of CCA implementation cost assets.

See KPMG Issues In-Depth, Non-GAAP financial measures, for additional guidance on non-GAAP reporting.

Question 7.2.20

Where are prepaid hosting service fees presented on the balance sheet?

Interpretive response: CCAs are service arrangements. Therefore, prepaid hosting service fees are presented consistent with how the entity presents other prepaid service fees. [350-40-15-4C, ASU 2018-15.BC2]

When a classified balance sheet is presented, such fees are presented on a classified basis (i.e. current and noncurrent portions) in balance sheet line items such as 'prepaid expenses and other current assets' and 'other assets'.

Question 7.2.30

Are capitalized CCA implementation costs bifurcated into current and noncurrent during the implementation phase?

Background: The implementation phase of a CCA may cross one or more financial reporting dates; an entity may commence implementing a cloud-based solution in one reporting period, but not complete it during that same period.

Consequently, the question arises about how an entity should bifurcate capitalized CCA implementation costs into current and noncurrent portions at reporting dates during the implementation period.

For example, a calendar year-end entity (customer) commences a CCA implementation on November 1 but does not complete it by December 31. The entity incurs $100 of a total expected $300 in implementation costs through December 31. The entity's operating cycle is no longer than 12 months.

Interpretive response: We believe each implementation activity will usually relate to the CCA as a whole. For example, a particular configuration, unless subsequently changed, will affect how the entity uses the cloud-based solution throughout the term of the hosting arrangement. Therefore, the costs of that activity cannot be assigned to any discrete period within the term.

Consequently, we believe each dollar of incurred cost that exists at a reporting date before go-live should be allocated between current and noncurrent based on how much of that dollar the entity expects will be amortized during the next 12 months (or operating cycle, if longer).

Using the background example to illustrate, at December 31 the entity estimates the cloud-based solution will go live on April 1 and the term of the hosting arrangement to be three years. Therefore, on the balance sheet it presents:

- $25 of the $100 as current: 9 months' amortization over the 12 months from December 31; and
- $75 as noncurrent: 27 months' amortization more than 12 months from December 31.

Question 7.2.40

How is the expense for hosting services presented in the income statement?

Interpretive response: We believe hosting services expense should be presented within income from continuing operations; the precise line item(s) may vary from entity to entity and will generally depend on the functional nature of the cloud-based solution.

For example, hosting service fees for an HR solution may be presented in a G&A expense line-item, while hosting service fees for a customer relationship

management solution may be presented in a sales and marketing expense line-item.

Question 7.2.50

How are cash payments for CCA hosting service fees and implementation costs classified?

Interpretive response: In general, both types of payments should be classified as cash flows from operations.

The cash payments for the hosting service fees in a CCA are cash payments to a supplier for a service, and therefore are generally cash outflows for operating activities. [230-10-45-17(b), 350-40-15-4C]

Because the cash payments for CCA implementation costs are required to be classified consistent with how the hosting service fees are classified, they must also generally be classified as cash outflows for operating activities. [350-40-45-3]

Amortization of capitalized CCA implementation costs is presented as a reconciling item in the reconciliation of net income to net cash flows from operating activities.

7.3 Website development

7.3.10 Before adopting ASU 2025-06

Subtopic 350-50 does not contain financial statement presentation guidance, therefore the following applies.

- Costs capitalized from applying Subtopic 350-40 (see sections 4.2.20 to 4.2.50) follow the presentation guidance applicable to internal-use software assets (see section 7.2).
- Costs capitalized under other Topics should follow the presentation guidance in the applicable Topic (or Subtopic). For example, the requirements in Topic 360 apply to computer servers acquired for website development.

7.3.20 After adopting ASU 2025-06**

ASU 2025-06 eliminates Subtopic 350-50. However, because Subtopic 350-50 did not contain financial statement presentation guidance, the guidance in section 7.3.10 continues to apply.

7.4 Software to be sold, leased or marketed

Excerpt from ASC 985-20

45 Other Presentation Matters

General

45-1 Because amortization expense of capitalized software costs relates to a software product that is marketed to others, the expense shall be charged to cost of sales or a similar expense category.

45-2 In an entity's balance sheet, capitalized software costs having a life of more than one year or one operating cycle shall be presented as an other asset because the costs are an amortizable intangible asset.

45-3 The accounting requirements of Topic 350 do not apply to capitalized software costs. However, the presentation and disclosure requirements of that Topic do apply to capitalized software costs.

55 Implementation Guidance and Illustrations

General

> Implementation Guidance

• > Purchased Computer Software

• • > Software Purchased Before Technological Feasibility Established

55-14 An entity may purchase software before technological feasibility has been established. For example, an entity purchases software for $100,000 that can be resold for $75,000. The amount of $25,000 would be charged to research and development, and $75,000 would be capitalized. If the software product reached technological feasibility, the $75,000 would be included in the cost of the software product. If the technological feasibility of the software was never established, the $75,000 would be classified as inventory.

Capitalized production cost amortization is presented in cost of sales (or similar, such as cost of revenue). [985-20-45-1]

Subtopic 985-20 states: [985-20-45-2 – 45-3]

— capitalized production costs are an amortizable intangible asset; and
— the presentation requirements of Topic 350 (goodwill and intangible assets) apply to capitalized production costs.

Consequently, entities present such costs as an intangible asset on the balance sheet – with other intangible assets or as a separate line item. In our experience, entities with significant capitalized production costs often present such costs as a separate line item. [350-30-45-1]

Consistent with the presentation of other amortizable intangible assets, this means capitalized production costs are generally not bifurcated between current and noncurrent portions, and generally would be classified as

noncurrent unless the original economic life of the related software product is one year (or the entity's operating cycle) or less.

Question 7.4.10

Where is purchased software presented on the balance sheet?

Interpretive response: Purchased software to be integrated into another software product (or other product or process) should be presented together with other capitalized software production costs if acquired after technological feasibility is established for the related product or process (see section 5.2.40). [985-20-25-8, 55-14]

If that same purchased software has an alternative future use, it should be classified in accordance with its alternative future use if it is acquired before technological feasibility is established. [985-20-55-14]

If technological feasibility is later established, the capitalized costs of the purchased software are reclassified to be presented together with the capitalized production costs of the software product into which it is integrated. [985-20-55-14]

If technological feasibility of the software product into which it was intended to be integrated is never established, the purchased software continues to be presented consistent with its alternative future use. [985-20-55-14]

Question 7.4.20

How is the amortization of capitalized production costs allocated in the income statement when there are multiple cost of sales line items?

Background: Software entities frequently present multiple revenue and cost of sales (or cost of revenue) line items in their income statements. For example, a software entity may have some or all the following revenue line items with corresponding cost line items (not exhaustive):

- license or product revenue;
- maintenance and support revenue;
- professional services revenue; and/or
- hosting and subscription services revenue.

In this case, the question arises about whether software production costs amortization should be allocated among multiple cost of sales line items, and if so, which ones.

Interpretive response: In general, we believe software costs amortization should be allocated in a manner consistent with the entity's gross revenue estimates used in the NRV test (see Question 6.4.80).

That is, if an entity is including a revenue stream (e.g. PCS revenue) in its estimated gross revenues for the product in NRV testing, the entity has de

facto concluded the software product is integral to the entity's ability to earn those revenues. Accordingly, the cost of those revenues should include an allocation of the amortization of the software product's capitalized costs for the period.

We believe an appropriate basis for allocation would be a relative allocation based on the current period (for which amortization is being recognized) product-related revenues for each revenue stream, and that an entity should disclose its accounting policy in this regard if material. [235-10-50-3]

8. Disclosure

Detailed contents

New item added in this edition: **
Item significantly updated in this edition: #

8.1 How the standards work

All three subtopics addressed in this publication include relatively limited disclosure guidance.

— Subtopic 350-40 generally refers to other Topics (e.g. Topic 360 on PP&E or Topic 730 on R&D) for its disclosure requirements; it requires only a limited number of specific disclosures related to CCAs.

— Subtopic 350-50 (eliminated by ASU 2025-06) does not include disclosure guidance. However, entities undertaking website development are still required to make disclosures required by other Topics (e.g. Subtopic 350-40 or Topic 360) stemming from website development activities.

— Subtopic 985-20 requires entities to make disclosures about unamortized software production costs, periodic amortization and any software cost impairments (i.e. NRV write-downs). In addition, it refers entities to specific disclosure requirements in other Topics (i.e. Topic 275 on risks and uncertainties, Subtopic 350-30 on intangible assets and Topic 730).

Effects of ASU 2024-03 (DISE)**

On adoption of ASU 2024-03, Income Statement—Reporting Comprehensive Income—Expense Disaggregation Disclosures, also known colloquially as the 'DISE' standard, *public business entities (PBEs)* will be required to disaggregate 'relevant' income statement captions (e.g. cost of goods sold) into prescribed natural expense categories in the notes to the financial statements. Those natural expense categories are purchases of inventory; employee compensation; depreciation, depletion, and amortization of certain oil and gas producing activities; *depreciation (recorded under Subtopic 360-10); and intangible asset amortization (recorded under Subtopic 350-30).* [220-40-50-6, ASU 2024-03.BC97]

ASU 2024-03 is effective for business entities for annual periods beginning after December 15, 2026, and interim periods within annual periods beginning after December 15, 2027. Early adoption is permitted. [220-40-65-1]

The basis for conclusions to ASU 2024-03 discusses presentation under Subtopic 220-40 of external- and internal-use software amortization. For additional discussion of this ASU and considerations specific to the amortization of external- and internal-use software assets, as well as amortization of CCA implementation cost assets, see KPMG Issues In-Depth, Disaggregation of income statement expenses – DISE, specifically Questions 3.6.10 and 3.6.30.

8.2 Internal-use software and cloud computing arrangements

Excerpt from ASC 350-40

50 Disclosure

General

50-1 The General Subsection of this Subtopic does not require any incremental disclosures. Disclosure shall be made in accordance with existing authoritative literature including the following:

a. Topic 275
b. Subtopic 730-10
c. Topic 235
d. Subtopic 360-10.

Pending Content

Transition Date: (P) December 16, 2027; (N) December 16, 2027 | Transition Guidance: 350-40-65-4

50-1 The disclosure requirements in Subtopic 360-10 on property, plant, and equipment apply to capitalized costs accounted for under this Subtopic, regardless of how those costs are presented in the financial statements. For purposes of applying those disclosure requirements, any disclosures in Subtopic 360-10 related to property, plant, and equipment shall be applied to internal-use software costs and related amortization. ~~The General Subsection of this Subtopic does not require any incremental disclosures.~~ Additionally, disclosure ~~Disclosure~~ shall be made in accordance with existing authoritative literature including the following:

a. Topic 275
b. Subtopic 730-10
c. Topic 235
d. Subparagraph superseded by Accounting Standards Update No. 2025-06.~~Subtopic 360-10.~~

Implementation Costs of a Hosting Arrangement That Is a Service Contract

50-2 An entity shall disclose the nature of its **hosting arrangements** that are service contracts.

50-3 The disclosure requirements in the General Subsection of this Section are applicable to the capitalized implementation costs of hosting arrangements that are service contracts. An entity shall make the disclosures in Subtopic 360-10 as if the capitalized implementation costs were a separate major class of depreciable asset.

Pending Content

Transition Date: (P) December 16, 2026; (N) December 16, 2026 | Transition Guidance: 220-40-65-1

50-3 The disclosure requirements in the General Subsection of this Section are applicable to the capitalized implementation costs of hosting arrangements that are service contracts. An entity shall make the disclosures in Subtopic 360-10 as if the capitalized implementation costs were a separate major class of depreciable asset. See paragraphs 220-40-50-21 through 50-25 for additional disclosure requirements.

Excerpt from ASC 360-10

50 Disclosure

General

50-1 Because of the significant effects on financial position and results of operations of the depreciation method or methods used, all of the following disclosures shall be made in the financial statements or in notes thereto:

a. Depreciation expense for the period
b. Balances of major classes of depreciable assets, by nature or function, at the balance sheet date
c. Accumulated depreciation, either by major classes of depreciable assets or in total, at the balance sheet date
d. A general description of the method or methods used in computing depreciation with respect to major classes of depreciable assets.

Impairment or Disposal of Long-Lived Assets

> Impairment of Long-Lived Assets Classified as Held and Used

50-2 All of the following information shall be disclosed in the notes to financial statements that include the period in which an impairment loss is recognized:

a. A description of the impaired long-lived asset (asset group) and the facts and circumstances leading to the impairment
b. If not separately presented on the face of the statement, the amount of the impairment loss and the caption in the income statement or the statement of activities that includes that loss
c. The method or methods for determining fair value (whether based on a quoted market price, prices for similar assets, or another valuation technique)
d. If applicable, the segment in which the impaired long-lived asset (asset group) is reported under Topic 280.

8.2.10 Before adopting ASU 2025-06

Before ASU 2025-06, entities are not required by Subtopic 350-40 to make specific disclosures independent of what is required by other Topics, except as follows for CCA implementation cost assets: [350-40-50-1 – 50-3]

- disclose the nature of the entity's CCAs; and
- make disclosures as if CCA implementation cost assets were its own major class of depreciable asset.

The second bullet in the preceding paragraph means entities with CCA implementation cost assets should disclose:

- the gross carrying amount of CCA implementation cost assets;
- CCA implementation cost asset amortization expense for the period (see Question 8.2.20);
- accumulated amortization of CCA implementation cost assets (see Question 8.2.20); and
- the method(s) for computing amortization of CCA implementation cost assets.

8.2.20 After adopting ASU 2025-06**

ASU 2025-06 introduces an explicit requirement to provide the property, plant and equipment (PP&E) disclosures in Subtopic 360-10 for all internal-use software costs capitalized under Subtopic 350-40, regardless of how those costs are presented in the financial statements (e.g. as PP&E or an intangible asset – see section 7.2.10) or how the software was acquired (e.g. internally developed or licensed from a third party). [350-40-50-1, ASU 2025-06.BC90, BC94]

Importantly, the amendments also specify that the intangible asset disclosures in paragraphs 350-30-50-1 – 50-3 are not required for software costs capitalized under Subtopic 350-40. This may be an important clarification for entities given that *licensed* internal-use software is required to be presented as an intangible asset rather than PP&E (see section 7.2.10). [350-40-25-17, ASU 2025-06.BC90]

Question 8.2.05**

How do entities comply with the Subtopic 360-10 disclosure requirements for internal-use software assets classified as intangible assets?

Background: Section 7.2.10 addresses balance sheet presentation (i.e. as PP&E or intangible assets) of internal-use software assets.

Interpretive response: We believe entities may need to consider providing at least certain of the required Subtopic 360-10 disclosures *separately* from those it provides for assets classified as PP&E (including any internal-use software) to prevent them from being confusing or misleading.

For example, paragraph 360-10-50-1(a) requires disclosing *total* depreciation expense for the period; however, because intangible asset internal-use

software will not be presented together with PP&E on the balance sheet, financial statement users may be unable to reconcile this amount to changes in accumulated depreciation.

Additionally, entities may also need to consider whether to provide these required disclosures for intangible asset internal-use software in the intangible assets note – i.e. instead of in a PP&E note – to align with how these assets are classified on the balance sheet and where financial statement users may look for such disclosures.

Question 8.2.06**

Do the Subtopic 360-10 or Subtopic 350-30 disclosures apply to internal-use software acquired in a business combination?

Interpretive response: We believe that either set of disclosures may be appropriate to provide (so long as one set is provided and an entity applies that one set consistently to similar circumstances), given some may view there to be conflicting guidance as outlined in the two views that follow.

We observe, however, that the disclosure requirements in paragraphs 360-10-50-1 to 50-2 and those that would apply to internal-use software in paragraphs 350-30-50-1 to 50-3 are substantially similar, such that the differences between an entity's disclosures if it provides one versus the other may not be material.

View 1: Subtopic 350-30 disclosures apply

While paragraph 350-40-50-1 requires providing the Subtopic 360-10 disclosures for all costs 'accounted for under' Subtopic 350-40, Subtopic 350-30 (which includes the disclosure requirements in Section 350-30-50) – instead of Subtopic 350-40 – governs the accounting for intangible assets recognized under Subtopics 805-20 (business combinations) and 958-805 (not-for-profit entities – business combinations). Internal-use software acquired in a business combination is recognized under one of those Subtopics. [350-30-15-3(b)]

Further, only internal-use software costs 'recognized' under Subtopic 350-40 are *excluded* from the scope of Subtopic 350-30. [350-30-15-4(f)]

View 2: Subtopic 360-10 disclosures apply

The Subtopic 360-10 disclosures apply because ASU 2025-06 specifically amended paragraph 350-30-15-3 to specify that "the disclosure requirements of paragraphs 350-30-50-1 through 50-3 also apply to capitalized software costs *related to software to be sold, leased, or marketed that an entity recognizes in accordance with Subtopic 985-20*" [emphasis added]. This specific amendment suggests that the FASB intended for ASU 2025-06 to clarify that the Subtopic 350-30 disclosures only *ever* apply to external-use software. [350-30-15-3]

8.2.30 Before and after adopting ASU 2025-06**

Question 8.2.10

Is the list of other Topics in paragraph 350-40-50-1 a complete list?

Interpretive response: No. We believe the language in that paragraph is not exhaustive. As an example, Subtopic 350-40 specifies acquired internal-use software licenses are intangible assets. Therefore, we believe entities with acquired internal-use software license assets should consider the disclosure requirements in Subtopic 350-30 even though that Subtopic is not mentioned in paragraph 350-40-50-1. [350-30-50-1 – 50-5]

Question 8.2.20

Are entities required to disclose CCA implementation cost amortization expense and accumulated amortization separately?

Interpretive response: Yes. Paragraph 350-40-50-3 states entities must make disclosures as if CCA implementation cost assets were a major class of depreciable asset under Subtopic 360-10, and paragraph 360-10-50-1 permits disclosure of depreciation expense and accumulated depreciation in total, instead of by major class of depreciable asset.

However, CCA implementation cost assets are amortized, rather than depreciated, and are not PP&E assets (see section 7.2.20). Therefore, even if the entity typically discloses depreciation expense and accumulated depreciation of its PP&E in the aggregate, we believe an entity must disclose CCA implementation cost amortization and accumulated amortization separately.

Question 8.2.30

Are the internal-use software and CCA disclosures required for both interim and annual financial reporting periods?

Interpretive response: No. There are no internal-use software disclosures required in interim periods. However, the need for additional interim disclosures should be evaluated under the requirements of Topic 270 (interim reporting).

8.3 Website development#

Subtopic 350-50 does not contain disclosure guidance and is eliminated by ASU 2025-06. Therefore, both before and after entities adopt ASU 2025-06, we believe:

— costs capitalized from applying Subtopic 350-40 (see chapter 4) should follow the disclosure guidance applicable to internal-use software assets (see section 8.2); and

— costs capitalized under other Topics should follow the disclosure guidance in the applicable Topic (or Subtopic). For example, the disclosures in Section 360-10-50 would apply to purchased computer servers capitalized under Topic 360 as part of website development.

8.4 Software to be sold, leased or marketed

Excerpt from ASC 985-20

50 Disclosure

General

50-1 Both of the following shall be disclosed in the financial statements:

a. Unamortized computer software costs included in each balance sheet presented.
b. The total amount charged to expense in each income statement presented for both of the following:

   1. Amortization of capitalized computer software costs
   2. Amounts written down to net realizable value.

   The amortization and write-down amounts may be combined with only the total of the two expenses being disclosed.

50-2 Paragraph 350-30-15-3 requires that an entity apply the disclosure requirements of paragraphs 350-30-50-1 through 50-3 to capitalized software costs. Paragraph 730-10-50-1 requires that disclosure be made in the financial statements of the total research and development costs charged to expense in each period for which an income statement is presented and states that such disclosure shall include research and development costs incurred for a computer software product to be sold, leased, or otherwise marketed.

55 Implementation Guidance and Illustrations

> Illustrations

• > Example 1: Disclosure of Risks and Uncertainties Related to Capitalized Software Costs

55-23 This Example illustrates the application of the disclosure requirements of Topic 275 to risks and uncertainties related to capitalized software costs. This Example has the following assumptions.

55-24 Software, Inc. develops and markets computer programs. In 20X3, it acquired a software entity. A significant portion of the purchase price was allocated to (capitalized) Product A (present net book value of $5 million), the most significant and profitable software program currently being marketed by the acquired entity. Only nominal amounts of other software costs have been capitalized. Software, Inc. expects Product A and its derivatives to be among its most significant products over the next several years. However, a competitor has recently released a new product designed to compete directly with Product A. Software, Inc. amortizes the capitalized software costs of Product A by the greater of the following:

a. The ratio that current gross revenues for a product bear to the total of current and anticipated future gross revenues for that product
b. The straight-line method over the remaining estimated economic life of the product including the period being reported on, pursuant to this Subtopic.

55-25 The amount of the amortization computed for the year 20X4 was equal to 20% of the beginning-of-the-year capitalized amount and was a significant component of cost of sales.

55-26 The segment of the computer software industry in which Software, Inc. operates is characterized by sales of products occurring primarily on the basis of customers' perceptions of the relative technical merits of competing products. Those perceptions are greatly influenced by product reviews in technical journals and advertising, and they can change rapidly. Innovative products have been introduced in recent years that have reduced quickly and significantly the volume of sales of preexisting products in the same market niche. While management of Software, Inc. believes its estimates of future gross revenues and the estimated economic life of Product A used in the determination of the amortization of capitalized software costs are reasonable, new products introduced by its competitors, such as the one discussed in paragraph 985-20-55-24, could have a significant near-term negative effect on such estimates. As a result, the amount of periodic amortization could increase in the near term in amounts that could be material to the entity's financial statements.

55-27 Software, Inc. would make the following disclosure.

Software, Inc.'s policy is to amortize capitalized software costs by the greater of the following:

a. The ratio that current gross revenues for a product bear to the total of current and anticipated future gross revenues for that product
b. The straight-line method over the remaining estimated economic life of the product including the period being reported on.

It is reasonably possible that those estimates of anticipated future gross revenues, the remaining estimated economic life of the product, or both will be reduced significantly in the near term [due to competitive pressures]. As a result, the carrying amount of the capitalized software costs for Product A ($5 million) may be reduced materially in the near term.

55-28 In this Example, the entity acknowledges that the carrying amount of its capitalized software costs is subject to significant uncertainty. The uncertainty relates to estimates of future years' revenues and useful lives that are made at the date of the financial statements, and the entity is aware that circumstances

exist that could cause such estimates to change in the near term. The entity's disclosure makes clear that it is at least reasonably possible that the carrying amount could be reduced in the near term.

55-29 If the amortization policy in the preceding illustrative disclosure is already disclosed elsewhere in the notes, it need not be repeated. The reference in brackets to competitive pressures in the preceding illustrative disclosure is an example of voluntary disclosure of factors that cause the estimate to be sensitive to change that is encouraged by paragraph 275-10-50-9.

Subtopic 985-20 requires disclosing all the following for capitalized external-use software production costs.

- Unamortized costs for each balance sheet presented. [985-20-50-1(a)]
- Amortization expense and amounts written down to NRV in each income statement period. The entity can elect to disclose these separately from each other or as a single amount. [985-20-50-1(b)]
- R&D costs charged to expense in each income statement period for software that has not yet reached technological feasibility. [730-10-50-1, 985-20-50-2]

In addition, the disclosure requirements in 350-30-50-1 through 50-3 apply to capitalized software production costs. [985-20-50-2]

The implementation guidance in Subtopic 985-20 illustrates an example disclosure under Topic 275 for capitalized software production costs. The example emphasizes the importance of disclosures about the uncertainty and variability inherent in the entity's estimates that drive amortization and NRV of the entity's capitalized software production costs. [985-20-55-23 – 55-29]

9. ASU 2025-06 effective dates and transition**

Detailed contents

New chapter added in this edition: **

9.1 How the standard works

The FASB issued ASU 2025-06 in September 2025. The effective date for all entities is outlined in the table below.

Effective dates	All entities
Annual and interim periods – Fiscal years beginning after	Dec. 15, 2027
Early adoption permitted	Yes, in any interim or annual period for which an entity's financial statements have not been issued (or made available for issuance) as of the beginning of the entity's fiscal year

Entities may transition to the guidance in ASU 2025-06 using one of the following transition approaches.

Transition approaches	
Prospective	Apply the ASU to software costs incurred for (1) new internal-use software projects commencing on or after the adoption date and (2) in-process internal-use software projects for which capitalization has not begun before the adoption date.
Modified prospective	Like the prospective approach, apply the ASU to software costs incurred for (1) new internal-use software projects commencing on or after the adoption date and (2) in-process internal-use software projects for which capitalization has not begun before the adoption date. Unlike the prospective approach, *also* apply the ASU to any capitalized software costs for in-process projects as of the adoption date. Record a cumulative-effect adjustment to the opening balance of retained earnings if, as of the adoption date, it is not yet probable that the in-process project will be completed and the software used to perform its intended function (i.e. the amended probable-to-complete threshold – see section 3A.2.30 – has not bet met).
Retrospective	Apply the ASU as if it had always been effective (including in comparative periods).

An entity discloses the nature of, and reason for, the change in accounting principle, and its transition method. It also qualitatively describes the financial statement line items affected by the accounting change.

In addition, retrospective adopters must also quantitatively describe the effects of the accounting change.

Before the effective date, SEC registrants are required to disclose the potential material effects of issued accounting standards that have not yet been adopted. When assessing whether the effect of a new or updated standard is material, registrants must consider the full scope of the standard, including recognition, measurement, presentation and disclosure requirements. KPMG Hot Topic, SAB 74 reminders, outlines the requirements of, and highlights important reminders about, SAB 74 for ASUs issued but not yet adopted. [SAB Topic 11.M]

9.2 Effective dates

Excerpt from ASC 350-40

65 Transition and Open Effective Date Information

General

> Transition Related to Accounting Standards Update No. 2025-06, Intangibles—Goodwill and Other—Internal-Use Software (Subtopic 350-40): Targeted Improvements to the Accounting for Internal-Use Software

65-4 The following represents the transition and effective date information related to Accounting Standards Update No. 2025-06, Intangibles—Goodwill and Other—Internal-Use Software (Subtopic 350-40): Targeted Improvements to the Accounting for Internal-Use Software:

Effective date and early adoption

a. All entities shall apply the pending content that links to this paragraph for annual reporting periods beginning after December 15, 2027, and interim reporting periods within those annual reporting periods.
b. Early adoption of the pending content that links to this paragraph is permitted in an interim or annual reporting period in which financial statements have not yet been issued or made available for issuance. If an entity adopts the pending content that links to this paragraph in an interim reporting period, it shall adopt the pending content as of the beginning of the annual reporting period that includes that interim reporting period.

See section 9.1 for a table summarizing the effective dates of ASU 2025-06.

Question 9.2.10

If an entity early adopts ASU 2025-06 in an interim period, must it apply the guidance from the start of its fiscal year?

Interpretive response: Yes. While ASU 2025-06 permits early adoption during an interim period, it expressly requires the entity's adoption date to be the beginning of the fiscal year to which the interim period belongs. For example, if a calendar-year entity early adopts the ASU during its third fiscal quarter of 2026, it must do so *as of* January 1, 2026. [350-40-65-4(b)]

9.3 Transition

Excerpt from ASC 350-40

65 Transition and Open Effective Date Information

General

> Transition Related to Accounting Standards Update No. 2025-06, Intangibles—Goodwill and Other—Internal-Use Software (Subtopic 350-40): Targeted Improvements to the Accounting for Internal-Use Softwar**e**

65-4 The following represents the transition and effective date information related to Accounting Standards Update No. 2025-06, Intangibles—Goodwill and Other—Internal-Use Software (Subtopic 350-40): Targeted Improvements to the Accounting for Internal-Use Software:

Transition methods

c. An entity shall apply the pending content that links to this paragraph using one of the following transition methods:

   1. Prospectively to new software costs incurred for all projects, including costs incurred for in-process projects, during annual reporting periods (and interim reporting periods within those annual reporting periods) beginning after the date that the pending content that links to this paragraph is adopted.
   2. On a modified transition approach, as follows:

      i. Prospectively to new software costs incurred (for all projects, including costs incurred for in-process projects), excluding those described in (c)(2)(ii), during annual reporting periods (and interim reporting periods within those annual reporting periods) beginning after the date that the pending content that links to this paragraph is adopted.

      ii. For an in-process project that, as of the date that the pending content that links to this paragraph is adopted, the entity determines does not meet the capitalization requirements in paragraphs 350-40-25-12 through 25-12A but had met the capitalization requirements before that date, the entity shall derecognize capitalized costs for that in-process project through a cumulative-effect adjustment to the opening balance of retained earnings (or other appropriate components of equity or net assets in the statement of financial position) as of the beginning of the annual reporting period in which the pending content that links to this paragraph is adopted.

   3. Retrospectively through a cumulative-effect adjustment to the opening balance of retained earnings (or other appropriate components of equity or net assets in the statement of financial position) as of the beginning of the first period presented.

Transition disclosures

d. An entity that applies the pending content that links to this paragraph prospectively in accordance with (c)(1) shall provide the transition disclosures required by paragraph 250-10-50-1(a) in both the interim reporting period (if applicable) and the annual reporting period of the change.
e. An entity that applies the pending content that links to this paragraph using a modified transition approach in accordance with (c)(2) shall provide the transition disclosures required by paragraph 250-10-50-1(a) and the cumulative effect of the change on retained earnings (or other components of equity or net assets in the statement of financial position) as of the beginning of the annual reporting period in which the pending content that links to this paragraph is adopted in both the interim reporting period (if applicable) and the annual reporting period of the change.
f. An entity that applies the pending content that links to this paragraph retrospectively in accordance with (c)(3) shall provide the transition disclosures required by paragraph 250-10-50-1(a) through (b)(1), (b)(2) for any prior periods retrospectively adjusted, and (b)(3) and (c)(2) in both the interim reporting period (if applicable) and the annual reporting period of the change.

Question 9.3.10

If uncompleted software is not probable of completion at the adoption date, is it de facto impaired under paragraph 350-40-35-3 when using the prospective transition approach?

Background: When the capitalization criteria (see section 3A.2.30) are no longer met for *uncompleted* internal-use software, the in-process software asset is reported at the lower of its carrying amount and fair value less costs to sell. There is a rebuttable presumption that uncompleted internal-use software has a fair value of zero. [350-40-35-3]

Therefore, the question arises if a software project had capitalized costs before the adoption date, but the project is not probable of completion under the amended Subtopic 350-40 guidance as of the adoption date, are the previously capitalized costs automatically impaired given the guidance in the preceding paragraph and, if so, isn't that, in effect, requiring the modified prospective transition approach?

Interpretive response: Consistent with the final part of the question posed at the end of the background, we believe the prospective transition option would be effectively nullified if entities (1) apply paragraph 350-40-35-3 to the existing capitalized costs, and (2) conclude they cannot overcome that paragraph's rebuttable presumption of a zero fair value for the in-process software. We do not believe it was the FASB's intent to offer a null set transition option.

To that end, we believe it is reasonable to conclude that paragraph 350-40-35-3 does not apply when transitioning to ASU 2025-06 using the prospective approach. We believe that when adopting ASU 2025-06 for the first time, an

entity has not previously assessed its existing software projects 'under the capitalization requirements [as amended] in paragraphs 350-40-25-12 – 25-12A'. Therefore, the software projects in question did not *previously* meet *those* requirements (i.e. they met the *pre-ASU 2025-06* requirements in paragraph 350-40-25-12); accordingly, those projects cannot '*no longer*' meet the amended requirements and paragraph 350-40-35-3 does not come into play.

Example 9.3.10
ASU 2025-06 adoption – transition methods

Background

ABC Corp. is developing an analytics module (AM) for its proprietary Customer Relationship Management (CRM) platform. The AM will consolidate its global sales, marketing and customer service operations. The AM includes a predictive analytics feature that will use machine learning to forecast customer churn that contains novel and unproven elements. The core CRM functionalities were completed and ready for their intended use on July 1, Year 4, such that the AM reflects its own, discrete 'software project'.

As of July 1, Year 5, the AM is determined to be in the application development stage under legacy Subtopic 350-40. In addition, ABC's board of directors has approved the multi-year AM project budget and, after a feasibility study, Management has concluded it is probable the AM will be successfully developed and completed based on the skills of its developers and consultants.

Due to management's 'probable' conclusion, ABC begins capitalizing associated software development costs on July 1, Year 5 under the pre-ASU 2025-06 provisions in Subtopic 350-40. ABC capitalizes $1.5 million and $2.0 million for the fiscal years ending June 30, Year 6, and Year 7, respectively ($3.5 million in total). ABC has not yet ceased capitalization by June 30, Year 7, because the software is not yet substantially complete and ready for its intended use.

Transition

ABC adopts ASU 2025-06 for its fiscal year *beginning* July 1, Year 7 (i.e. fiscal Year 8). As of that date, ABC is continuing to develop AM. Contrary to the pre-ASU 2025-06 probable conclusion reached as of July 1, Year 5, *post-ASU 2025-06*, ABC concludes that it does not meet the probable-to-complete threshold as of its July 1, Year 7 adoption date because significant development uncertainty in the form of the novel predictive analytic functionalities has not yet been resolved through coding and testing, Therefore the software project does not meet the criteria for capitalization under ASU 2025-06 as of the adoption date.

ABC considers each of the available transition approaches and application as follows.

— **Prospective transition:** ABC ceases capitalizing incremental development costs for AM incurred on or after July 1, Year 7 until the significant development uncertainty related to the novel predictive module functions and features has been resolved. The previously capitalized balance of $3.5 million remains on the balance sheet, consistent with Question 9.3.10.

— **Modified prospective transition:** ABC derecognizes the capitalized AM costs on the basis that the AM project is not yet probable of completion under paragraphs 350-40-25-12 and 25-12A as of the adoption date. It records a cumulative-effect adjustment of $3.5 million to the opening balance of retained earnings as of July 1, Year 7.

— **Retrospective transition:** ABC makes a cumulative-effect adjustment to retained earnings to derecognize its CRM project as of the beginning of the first period presented in its fiscal Year 8 financial statements (for ABC, who is presenting only two years, this is July 1, Year 6). The fiscal Year 7 financial statements presented in the fiscal Year 8 financial statements will be recast as if ABC had been applying the amendments enacted by ASU 2025-06 all along.

The following summarizes the relevant journal entries under each transition approach.

Account	Debit	Credit
Prospective transition		
No entry is recorded as of the date of adoption. The $3.5 million in capitalized costs remains on the balance sheet. Future costs are expensed as incurred until the project meets the post-ASU 2025-06 capitalization criteria. The core CRM platform's capitalized costs remain unchanged as they are unrelated to the AM software project and continue to be amortized over their useful life.		
Modified prospective transition		
Opening retained earnings as of July 1, Year7	$3,500,000	
Capitalized software costs (AM)		$3,500,000
ABC derecognizes previously capitalized costs for the AM project it determines does not meet the capitalization requirements in paragraph 350-40-25-12 – 25-12A as of the adoption date. The core CRM platform's capitalized costs remain unchanged as they are unrelated to the AM software project and continue to be amortized over their useful life.		
Retrospective transition		
Beginning retained earnings as of July 1, Year 6	$1,500,000	
Capitalized software costs (AM)		$1,500,000
Apply the ASU as if it had always been effective. An entry is recorded to retrospectively adjust retained earnings for capitalized AM software costs derecognized as of the beginning of the comparative period. For the years ended June 30, Year 7 and June 30, Year 8, the financial statements for each period will reflect applying ASU 2025-06 as if ABC had done so all along (e.g. costs capitalized in fiscal Year 7 pre-adoption will be reflected as expense in the fiscal Year 7 income statement presented in the fiscal Year 8 financial statements). Consistent with the other two transition approaches, the core CRM platform's capitalized costs remain unchanged as they are unrelated to the AM software project and continue to be amortized over their useful life.		

Subtopic 350-40 Glossary

Excerpt from Topic 350-40

20 Glossary

Contract

An agreement between two or more parties that creates enforceable rights and obligations.

Customer

A party that has contracted with an entity to obtain goods or services that are an output of the entity's ordinary activities in exchange for consideration.

Hosting Arrangement

In connection with accessing and using software products, an arrangement in which the customer of the software does not currently have possession of the software; rather, the customer accesses and uses the software on an as-needed basis.

Not-for-Profit Entity

An entity that possesses the following characteristics, in varying degrees, that distinguish it from a business entity:

a. Contributions of significant amounts of resources from resource providers who do not expect commensurate or proportionate pecuniary return
b. Operating purposes other than to provide goods or services at a profit
c. Absence of ownership interests like those of business entities.

Entities that clearly fall outside this definition include the following:

a. All investor-owned entities
b. Entities that provide dividends, lower costs, or other economic benefits directly and proportionately to their owners, members, or participants, such as mutual insurance entities, credit unions, farm and rural electric cooperatives, and employee benefit plans.

Pending Content

Transition Date: (P) December 16, 2027; (N) December 16, 2027 | Transition Guidance: 350-40-65-4

Performance Requirements

Performance requirements are what an entity needs the software to do (for example, functions or features).

Preliminary Project Stage

When a computer software project is in the preliminary project stage, entities will likely do the following:

a. Make strategic decisions to allocate resources between alternative projects at a given point in time. For example, should programmers

develop a new payroll system or direct their efforts toward correcting existing problems in an operating payroll system?
b. Determine the performance requirements (that is, what it is that they need the software to do) and systems requirements for the computer software project it has proposed to undertake.
c. Invite vendors to perform demonstrations of how their software will fulfill an entity's needs.
d. Explore alternative means of achieving specified performance requirements. For example, should an entity make or buy the software? Should the software run on a mainframe or a client server system?
e. Determine that the technology needed to achieve performance requirements exists.
f. Select a vendor if an entity chooses to obtain software.
g. Select a consultant to assist in the development or installation of the software.

Pending Content

Transition Date: (P) December 16, 2027; (N) December 16, 2027 | Transition Guidance: 350-40-65-4

Preliminary Project Stage

Paragraph superseded by Accounting Standards Update No. 2025-06.

Probable

The future event or events are likely to occur.

Public Business Entity

A public business entity is a business entity meeting any one of the criteria below. Neither a **not-for-profit entity** nor an employee benefit plan is a business entity.

a. It is required by the U.S. Securities and Exchange Commission (SEC) to file or furnish financial statements, or does file or furnish financial statements (including voluntary filers), with the SEC (including other entities whose financial statements or financial information are required to be or are included in a filing).
b. It is required by the Securities Exchange Act of 1934 (the Act), as amended, or rules or regulations promulgated under the Act, to file or furnish financial statements with a regulatory agency other than the SEC.
c. It is required to file or furnish financial statements with a foreign or domestic regulatory agency in preparation for the sale of or for purposes of issuing securities that are not subject to contractual restrictions on transfer.
d. It has issued, or is a conduit bond obligor for, **securities** that are traded, listed, or quoted on an exchange or an over-the-counter market.
e. It has one or more securities that are not subject to contractual restrictions on transfer, and it is required by law, contract, or regulation to prepare U.S. GAAP financial statements (including notes) and make them publicly available on a periodic basis (for example, interim or annual periods). An entity must meet both of these conditions to meet this criterion.

An entity may meet the definition of a public business entity solely because its financial statements or financial information is included in another entity's filing

with the SEC. In that case, the entity is only a public business entity for purposes of financial statements that are filed or furnished with the SEC.

Revenue

Inflows or other enhancements of assets of an entity or settlements of its liabilities (or a combination of both) from delivering or producing goods, rendering services, or other activities that constitute the entity's ongoing major or central operations.

Security

A share, participation, or other interest in property or in an entity of the issuer or an obligation of the issuer that has all of the following characteristics:

a. It is either represented by an instrument issued in bearer or registered form or, if not represented by an instrument, is registered in books maintained to record transfers by or on behalf of the issuer.
b. It is of a type commonly dealt in on securities exchanges or markets or, when represented by an instrument, is commonly recognized in any area in which it is issued or dealt in as a medium for investment.
c. It either is one of a class or series or by its terms is divisible into a class or series of shares, participations, interests, or obligations.

Standalone Price

The price at which a customer would purchase a component of a **contract** separately.

Useful Life

The period over which an asset is expected to contribute directly or indirectly to future cash flows.

Subtopic 985-20 Glossary

Excerpt from Topic 985-20

20 Glossary

Acquiree

The **business** or **businesses** that the **acquirer** obtains control of in a **business combination**. This term also includes a nonprofit activity or business that a not-for-profit acquirer obtains control of in an **acquisition by a not-for-profit entity**.

Acquirer

The entity that obtains control of the **acquiree**. However, in a **business combination** in which a **variable interest entity** (VIE) is acquired, the primary beneficiary of that entity always is the acquirer.

Acquisition by a Not-for-Profit Entity

A transaction or other event in which a not-for-profit acquirer obtains control of one or more nonprofit activities or businesses and initially recognizes their assets and liabilities in the acquirer's financial statements. When applicable guidance in Topic 805 is applied by a **not-for-profit entity**, the term **business combination** has the same meaning as this term has for a for-profit entity. Likewise, a reference to business combinations in guidance that links to Topic 805 has the same meaning as a reference to acquisitions by not-for-profit entities.

Business

Paragraphs 805-10-55-3A through 55-6 and 805-10-55-8 through 55-9 define what is considered a business.

Business Combination

A transaction or other event in which an **acquirer** obtains control of one or more **businesses**. Transactions sometimes referred to as true mergers or mergers of equals also are business combinations. See also **Acquisition by a Not-for-Profit Entity**.

Coding

Generating detailed instructions in a computer language to carry out the requirements described in the detail program design. The coding of a computer software product may begin before, concurrent with, or after the completion of the detail program design.

Contract

An agreement between two or more parties that creates enforceable rights and obligations.

Customer

A party that has contracted with an entity to obtain goods or services that are an output of the entity's ordinary activities in exchange for consideration.

Customer Support

Services performed by an entity to assist customers in their use of software products. Those services include any installation assistance, training classes, telephone question and answer services, newsletters, on-site visits, and software or data modifications.

Detail Program Design

The detail design of a computer software product that takes product function, feature, and technical requirements to their most detailed, logical form and is ready for coding.

Hosting Arrangement

In connection with accessing and using software products, an arrangement in which the customer of the software does not currently have possession of the software; rather, the customer accesses and uses the software on an as-needed basis.

Legal Entity

Any legal structure used to conduct activities or to hold assets. Some examples of such structures are corporations, partnerships, limited liability companies, grantor trusts, and other trusts.

Maintenance

Activities undertaken after the product is available for general release to customers to correct errors or keep the product updated with current information. Those activities include routine changes and additions.

Not-for-Profit Entity

An entity that possesses the following characteristics, in varying degrees, that distinguish it from a business entity:

a. Contributions of significant amounts of resources from resource providers who do not expect commensurate or proportionate pecuniary return
b. Operating purposes other than to provide goods or services at a profit
c. Absence of ownership interests like those of business entities.

Entities that clearly fall outside this definition include the following:

a. All investor-owned entities
b. Entities that provide dividends, lower costs, or other economic benefits directly and proportionately to their owners, members, or participants, such as mutual insurance entities, credit unions, farm and rural electric cooperatives, and employee benefit plans.

Product Design

A logical representation of all product functions in sufficient detail to serve as product specifications.

Product Enhancement

Improvements to an existing product that are intended to extend the life or improve significantly the marketability of the original product. Enhancements normally require a product design and may require a redesign of all or part of the existing product.

Revenue

Inflows or other enhancements of assets of an entity or settlements of its liabilities (or a combination of both) from delivering or producing goods, rendering services, or other activities that constitute the entity's ongoing major or central operations.

Testing

Performing the steps necessary to determine whether the coded computer software product meets function, feature, and technical performance requirements set forth in the product design.

Variable Interest Entity

A **legal entity** subject to consolidation according to the provisions of the Variable Interest Entities Subsections of Subtopic 810-10.

Working Model

An operative version of the computer software product that is completed in the same software language as the product to be ultimately marketed, performs all the major functions planned for the product, and is ready for initial customer testing (usually identified as beta testing).

Index of changes

This index lists the significant additions and changes made in this edition to assist you in locating recently added or updated content. The following symbols are used throughout this Handbook to indicate the types of revisions made in this edition for chapters, sections, Questions, Examples and other items:

** new item added in this edition
# item significantly updated in this edition

KPMG Financial Reporting View

Delivering guidance and insights, KPMG Financial Reporting View is ready to inform your decision making. Stay up to date with us.

Defining Issues

Our collection of newsletters with insights and news about financial reporting and regulatory developments, including Quarterly Outlook and FRV Weekly.

Handbooks and Hot Topics

Our discussion and analysis of accounting topics – from short Hot Topics that deal with a topical issue, to our in-depth guides covering a broad area of accounting.

CPE opportunities

Register for live discussions of topical accounting and financial reporting issues. CPE-eligible replays also available.

Financial Reporting Podcasts

Tune in to hear KPMG professionals discuss major accounting and financial reporting developments.

Visit **Financial Reporting View**
and **sign up** for news and insights

Access our US Handbooks

As part of Financial Reporting View, our library of in-depth guidance can be accessed here, including the following Handbooks.

- Accounting changes and error corrections
- Accounting for economic disruption
- Asset acquisitions
- Bankruptcies
- Business combinations
- Business combinations (SEC reporting)
- Climate risk in the financial statements
- Consolidation
- Contingencies, commitments and guarantees
- Credit impairment
- Debt and equity financing
- Derivatives and hedging
- Discontinued operations and held-for-sale disposal groups
- Earnings per share
- Employee benefits
- Equity method of accounting
- Fair value measurement
- Financial statement presentation
- Foreign currency
- GHG emissions reporting
- Going concern
- IFRS® compared to US GAAP
- Impairment of nonfinancial assets
- Income taxes
- Internal control over financial reporting
- Inventory
- Investment companies
- Investments
- Leases
- Long-duration contracts
- Reference rate reform
- Research and development
- Revenue recognition
- Revenue: Real estate
- Revenue: Software and SaaS
- Segment reporting
- Service concession arrangements
- Share-based payment
- Software and website costs
- Statement of cash flows
- Tax credits
- Transfers and servicing of financial assets

Learn about us:  **kpmg.com**