Grant Thornton

July 2020

Viewpoint

Accounting for software costs

Contents

Introduction

Companies today rely on software in countless ways to help forge a competitive edge, whether it's using basic email and smartphones or pioneering new ways to harness artificial intelligence. Savvy businesses routinely use software to streamline operations like payroll, manufacturing, or financial reporting, while a growing number of executives are deploying data analytics to grow sales and inform smart business decisions.

The myriad ways that companies currently leverage software are matched by the number of ways that software can be developed or otherwise obtained. Companies can develop software internally, externally, or jointly with a third party. Software can be purchased off-the-shelf and used directly as a stand-alone product or customized to meet a company's specific needs. Software can also be embedded into an existing product or process, or it can be accessed directly online via a hosting arrangement that is provided by a third party.

While this variety bodes well for growing a business, accounting for the costs of software can be somewhat of a challenge. The FASB's Codification features no fewer than five Topics that offer guidance on how to account for the costs of developing, purchasing, and implementing software. This guidance is nuanced depending on how a company either obtains or develops, and how it ultimately uses, the software.

For example, software that is used exclusively for internal purposes, whether it is developed internally or acquired from an outside party, is accounted for using the guidance in ASC 350-40, *Internal-Use Software*—except for certain costs that are incurred when internal software is used in research and development, which are accounted for under ASC 730, *Research and Development*.

In contrast, software that is sold, leased, or marketed as a stand-alone product, or as an integral component of another product or process, is accounted for using the guidance in ASC 985-20, *Software – Costs of Software to Be Sold, Leased or Marketed*, regardless of whether it is developed internally or purchased from a third party.

Determining which guidance applies to a hosting arrangement depends not only on whether the company is a customer or a vendor, but also on whether the customer ultimately takes control of the software or can only access it. Hosting arrangements are generally treated as service contracts unless the customer takes or can take possession of the software. Some of a customer's costs to implement such an arrangement may be capitalized under the guidance in ASC 350-40.

This publication unravels the FASB's guidance on accounting for software costs in ASC 350-40, ASC 730, and ASC 985-20, by using direct citations from the Codification, examples created to illustrate the FASB's guidance, and insights based on our experience with clients and conversations with colleagues and standard-setters.

This publication will be updated periodically to reflect new guidance and practice issues that develop, and is written to reflect the adoption of the following recently issued Accounting Standards Updates that impact the accounting for software costs:

- ASU 2018-15, *Customer's Accounting for Implementation Costs Incurred in a Cloud Computing Arrangement That Is a Service Contract (a consensus of the FASB Emerging Issues Task Force)*
- ASC 606, *Revenue from Contracts with Customers*

1. Scope

Entities today obtain software in many ways. Sometimes, software is a standard off-the-shelf product with broad applicability that can be easily purchased and is ready to use without any customization or complex installation. Other times, an entity might engage a third-party specialist or use its own internal specialist to develop customized software designed to meet its specific needs. Somewhere between these two options, an entity might purchase base software from a third party and then customize it internally or with the help of a third party for its own purposes. Finally, an entity may enter into a hosting arrangement whereby a third party makes software available for the entity to access online as needed.

Entities also use software in many different ways. An entity may use software internally to run its business, or it may sell or provide access to software in contracts with customers. Some entities purchase or develop software to use as part of research and development activities that are focused on developing new products or services.

How the entity obtains and uses software will impact the accounting for a particular software product. The following table outlines the various FASB guidance that might apply to accounting for software costs and indicates when that guidance should be applied.

Figure 1.1 Summary of guidance for software development costs

Guidance	Applicability
ASC 350-40, *Intangibles – Goodwill and Other: Internal-Use Software*	Applies when there is no intention to sell the software; rather, it will be used solely in operating an entity's business, including • *For a vendor* – When the software will be used by the vendor in providing a cloud computing service arrangement where the customer does not take possession of the software and cannot host it on its own. • *For a customer* – When the customer can take possession of the software and host it on its own.
ASC 985-20, *Software: Costs of Software to Be Sold, Leased, or Marketed*	Applies to software development costs for a software product that will either be sold or embedded in a product that will subsequently be sold, leased, or otherwise marketed.
ASC 730, *Research and Development*	Applies to costs incurred to internally develop software to be used in research and development.

Guidance	Applicability
ASC 720-45, *Other Expenses: Business and Technology Reengineering*	Provides guidance on the costs associated with business process reengineering and information technology transformation projects.
ASC 350-50, *Website Development Costs*	Discusses the accounting for costs incurred in the five stages of website development, which are outlined in this guidance.

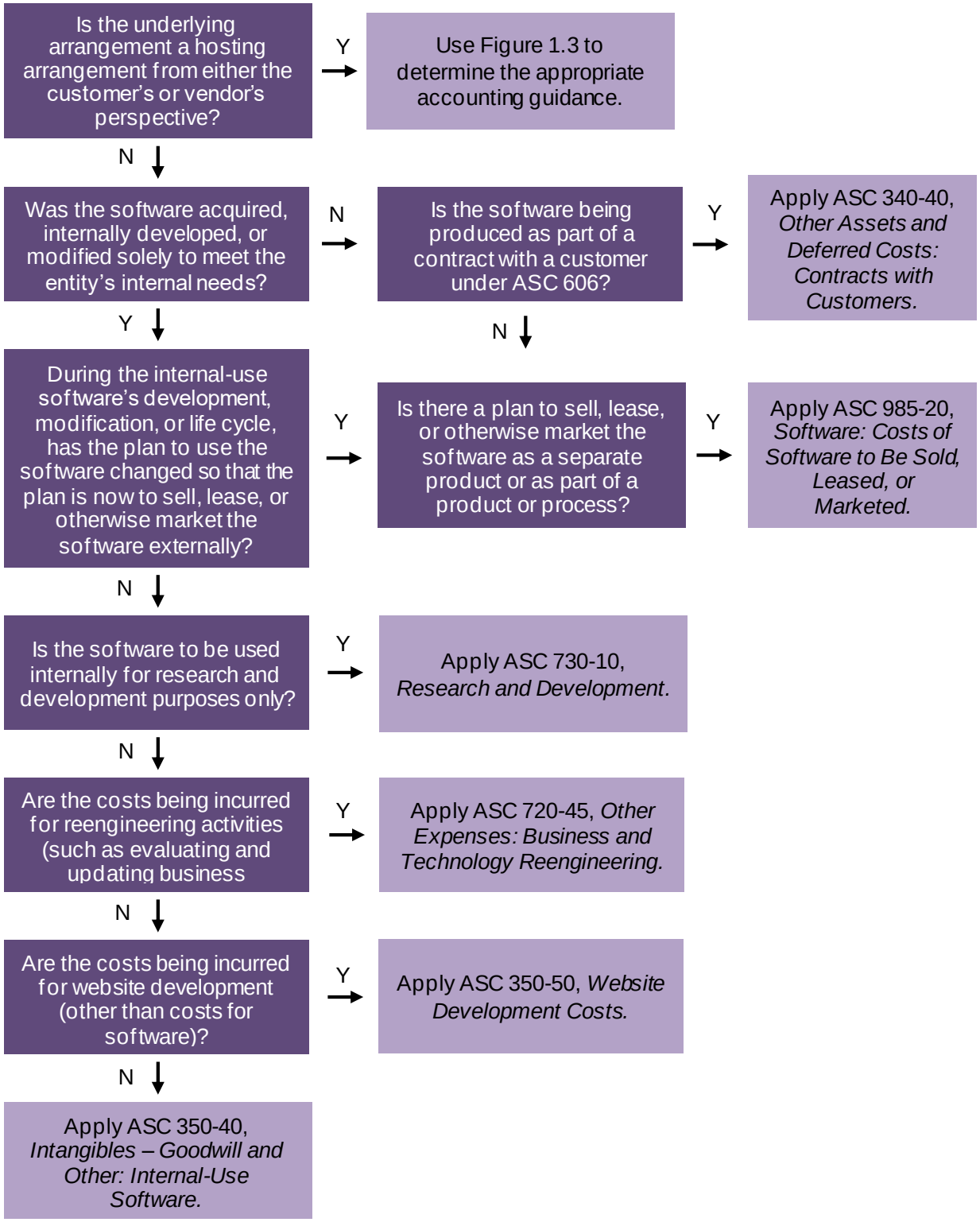
Determining which guidance applies to the costs of developing, purchasing, or implementing software requires an understanding of the intended use of the software product, the types of costs involved, and the product's stage of development. Broadly, the first step is for an entity to determine if the software being developed or implemented is only for internal use, such as a customized inventory tracking system, or if it is being designed to sell to customers either as a stand-alone product or as an integral component of another product. An entity must also determine if the arrangement is a hosting arrangement and, if so, whether the customer can take possession of the software underlying the hosting arrangement and is able to host the software on its own or with another third party.

The guidance in this Viewpoint is written from the perspective of an entity that has adopted ASU 2018-15, *Customer's Accounting for Implementation Costs Incurred in a Cloud Computing Arrangement That Is a Service Contract,* which is effective for public business entities for fiscal years beginning after December 15, 2019, and interim periods within those fiscal years, and for all other entities in annual reporting periods beginning after December 15, 2020, and interim periods beginning after December 15, 2021. An entity may elect to early adopt the guidance, including in interim periods, and may apply the guidance prospectively or retrospectively.

This Viewpoint is also written from the perspective of an entity that has adopted ASC 606, *Revenue from Contracts with Customers.*

The following figure illustrates how an entity should evaluate which guidance to apply to particular software-related costs. The scope of each type of software-development accounting guidance in the figure is discussed in the rest of this section.

Figure 1.2: Determining the appropriate accounting to apply to software related costs



1.1 Internal-use software

Although internal-use software is generally considered to be an intangible asset, it has characteristics that are similar to property, plant, and equipment: it is specifically identifiable, it may have a useful life of several years or more, it is not intended for sale in the ordinary course of business, and it is either acquired or developed with the intention of being used by the entity. The guidance used to account for internal-use software under ASC 350-40 is likewise similar to the guidance in ASC 360 *Property, Plant, and Equipment*, in that it accounts for internal-use software using a cost accumulation model and amortizes the asset over its useful life.

However, the guidance on accounting for internal-use software in ASC 350-40, which includes software that is used as all or a part of a service offering (see Section 1.1), differs from the guidance on accounting for software that an entity intends to sell during the ordinary course of business, which falls under ASC 985-20, *Software: Costs of Software to Be Sold, Leased, or Marketed*. For software to be considered internal-use software, it must be used solely to meet an entity's internal needs, and the entity must not have a substantive plan to market the software externally. Internal-use software may either be developed internally or acquired from a third party. If there is a substantive plan to market the software externally, regardless of whether it already exists or is being developed, the software does not qualify as internal-use software. In other words, if the initial primary purpose of a specific software product is for the entity's own use but its secondary purpose is to be sold or marketed to outside parties, the software does not meet the definition of internal-use software. The software must be created and intended *solely* for internal use to qualify as internal-use software under ASC 350-40.

Internal-use software is accounted for using the guidance in ASC 350-40, as discussed in Section 2. Software that is developed to be marketed, sold, or leased is accounted for using the guidance in ASC 985-20, as discussed in Section 1.3 and Section 4.

ASC 350-40-15-2A

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.

1.1.1 Substantive plan to market software externally

ASC 350-40 includes a list of conditions that an entity should consider when determining whether it has a "substantive" plan to market software externally. First, implementation of the plan must be "reasonably possible" in order for a plan to be considered "substantive." The implementation would be considered reasonably possible if the chance of it occurring is greater than remote. However, the chance that the implementation would occur does not need to be likely in order for it to be considered reasonably possible. Therefore, activities like engaging in a market feasibility study, or entering into a joint arrangement to share costs with another entity to develop software that both entities plan to use

internally, would not qualify as a substantive plan to market the software externally. On the other hand, any one of the following conditions might indicate that a plan to market software externally is substantive:

- Selecting a marketing channel,
- Identifying specific promotional activities.
- Developing a plan for delivery, billing, and support of the software product.

ASC 350-40-15-2B

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.

Determining whether a marketing plan is 'substantive' under ASC 350-40

The following examples illustrate how an entity might determine whether it has a substantive plan to market software externally.

Substantive marketing plan

A bank develops software that allows customers to transfer cash from their account to another individual's account using a mobile application. The software is initially developed as a feature to be accessed only by the bank's customers as software as a service (SaaS). While development of the app is still in the "preliminary project stage," as discussed in Section 2.1, the bank conducts exploratory market research and discovers there is a market to sell the software to other banks. The bank decides to pursue this option but does not intend to market the software to other banks immediately after completing the app, because it expects that offering the application exclusively to its own retail banking customers might grow its customer base. The bank establishes a committee of individuals comprising software developers, marketing personnel, and finance staff to identify other banks as potential customers and to develop the infrastructure necessary to support the product when it is released for sale to other banks in the future.

The bank concludes that it has a substantive plan to market the software externally because implementation of the plan is reasonably possible, and accounts for the costs incurred to develop the software in accordance with ASC 985-20.

No substantive marketing plan

Assume the same facts in the previous example, except that, as the outcome of the exploratory market research, the bank instead decides that the benefits of offering the SaaS software exclusively to its customers outweigh the marketing costs and potential profit from selling the software to its competitors.

The bank therefore decides not to move forward with a plan to market the software externally to other banks and accounts for the software under the internal-use guidance in ASC 350-40.

The guidance in ASC 350-40 also requires entities to consider their past practices when evaluating whether a software product is designed only for internal use. If an entity has both used software internally and sold the same software externally in the past, then there is a rebuttable presumption that any software the entity develops is intended for sale, as explained in ASC 350-40-15-2C.

ASC 350-40-15-2C

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.

Grant Thornton insights: Considering an entity's past practices

An entity with a history of subsequently marketing and selling software products initially developed for internal use must carefully consider whether any newly developed software may subsequently be leased or sold in determining how to account for software development costs. The FASB has stated that a past practice of selling internally used software creates a rebuttable presumption that any software developed by the entity will be marketed, sold, or leased. Under ASC 350-40, an entity is required to present a preponderance of evidence to overcome this presumption.

Drawing on the previous example in "Determining whether a marketing plan is substantive under ASC 350-40" with no substantive marketing plan above, if the bank reverses its decision not to market or sell the software and subsequently markets the internal-use software to other banks, management should evaluate whether it has established a practice of both using and selling computer software. If so, a rebuttable presumption that the bank intends to market or sell future software projects would then exist. The bank could overcome the rebuttable presumption if a preponderance of the evidence indicates that the software will not be marketed or sold.

For example, assume management determined that a rebuttable presumption exists. The bank begins developing a new software product to provide to its customers as part of its services, similar to the cash transfer software from the previous example. The software is an app that the customers can download to assist in splitting a check among patrons at a restaurant. The bank does not have plans to sell or market the app to other banks. However, because of the bank's history of selling software initially intended to be solely for its customers, it might conclude that it is unable to overcome the rebuttable presumption that it would sell this software. In that case it would be required to account for the development of this software in accordance with the guidance in ASC 985-20, unless a preponderance of evidence exists to overcome the presumption that the software is intended for sale.

On the other hand, if the bank begins developing a customized software product to interface with its highly customized financial reporting software by pulling data for strategic analysis, it might be able to

provide sufficient evidence that the software will not be transferred to other entities in the future, since the software could not feasibly interface with other entities' financial reporting software. Additionally, the type of financial analysis conducted by the software may be key to the bank's overall strategy for generating profit, in which case, sharing the strategy could significantly impair the bank's ability to remain competitive in the marketplace. If the bank can provide compelling evidence that the software will never be sold, leased, or otherwise marketed, it may be able to overcome the rebuttable presumption that the software will be sold, leased, or marketed externally. If so, the bank should account for the software under the guidance for internal-use software in ASC 350-40.

1.1.2 Identifying internal-use software

To help entities correctly identify internal-use software, the FASB has provided lists of examples that both would and would not qualify as internal-use software accounted for under ASC 350-40. While the lists provide specific examples, there are common elements among the items on the lists that may be helpful for an entity trying to determine whether its software is internal-use.

Common elements of internal-use software include software that will be used:

- As part of a manufacturing process. This could include software that runs machines or equipment used by the entity to produce its goods, or software used to control other operations within a manufacturing plant or distribution center.
- By the entity's employees to provide services to customers.
- By the entity in its internal operations, such as in the accounting, finance, or payroll departments, for storage of information, or internal communications.

Common elements of software that is not intended for internal use include software that:

- Is integrated into or operates a good manufactured or sold by the entity.
- Creates a digital version of an analog item that is then sold to customers.
- Meets any of the previously discussed scope exceptions (software that an entity has a plan to sell, software developed under a contract with a customer, software to be used for research and development).

ASC 350-40-55-1

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.

ASC 350-40-55-2

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.

1.2 Software to be used in research and development

Certain costs associated with software developed for internal research and development activities are within the scope of ASC 730, *Research and Development,* rather than ASC 350-40. The accounting for software used in research and development differs, depending on whether the entity purchases, leases, or internally develops the software.

When an entity purchases or leases software from a third party to use in research and development activities, it should consider whether the software could have an alternative future use beyond the existing research and development project. If the software has a future internal use, an entity should account for the costs in accordance with the guidance in ASC 350-40. Otherwise, the costs associated with purchasing or leasing the software should be expensed in accordance with ASC 730.

If the software is developed internally for a specific research and development project, an entity should account for the costs under ASC 730, regardless of whether the software has an alternative future use. Furthermore, any costs associated with developing software for a "pilot project," meaning a project that is not executed on a scale that is economically feasible for commercial production, should also be accounted for as research and development under ASC 730.

The accounting for software costs under ASC 730 is discussed in more detail in Section 3.

ASC 350-40-15-7

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.

1.3 Software to be sold, leased, or marketed

Software that is developed or purchased by an entity that will be sold, leased, or marketed is accounted for under ASC 985-20. This guidance applies to software that will be sold as a stand-alone product as well as to software that will be sold as part of another product or process, regardless of whether it is internally developed or purchased from a third party. An entity that has a substantive plan to externally market software that has been developed internally should account for the software as a product to be sold under ASC 985-20. (Refer to Sections 1.1.1. and 1.1.2 for a discussion of the criteria to consider when determining whether software will be sold externally or is designed solely for internal use.)

Software that is developed for internal use or for research and development purposes, along with software that is produced, modified or customized as part of a contract with a customer, does not fall within the scope of ASC 985-20. Accounting for software that will be sold, leased, or marketed is discussed in more detail in Section 4.

ASC 985-20-15-2

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.

ASC 985-20-15-3

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).

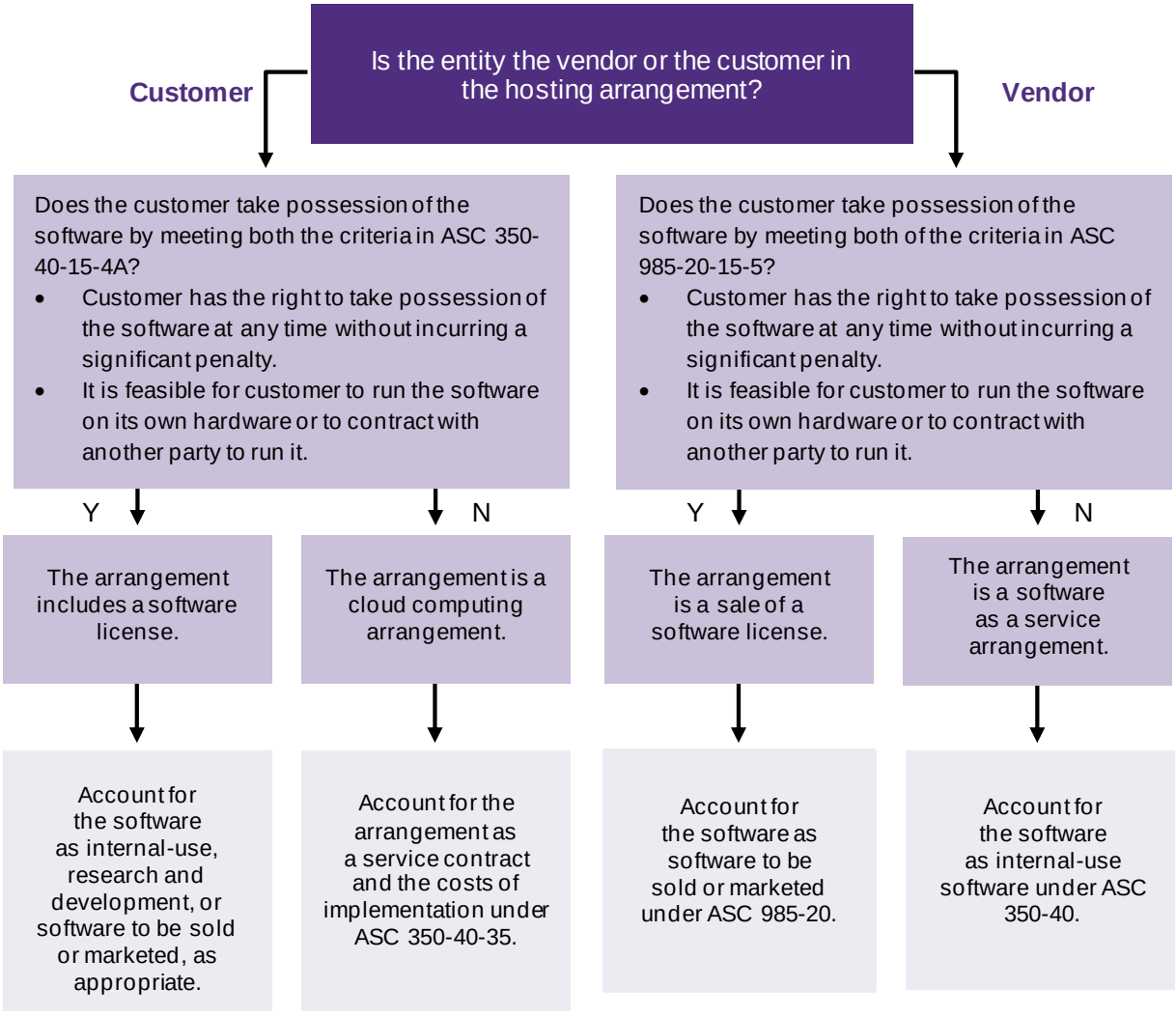
1.4 Hosting arrangements

According to the Codification's Master Glossary, the term "hosting arrangement" refers to a transaction entered into by a vendor to provide a customer with access to software that the customer can use, but does not own or possess.

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.

The phrase “hosting arrangement” can describe the overall transaction from either the customer’s or the vendor’s perspective. In this publication, however, the term “cloud computing arrangement” describes a hosting arrangement from the customer’s perspective, whereas the term “software as a service” describes the same transaction from the vendor’s perspective. See Figure 1.3 below for a summary of the accounting for a hosting arrangement.

Figure 1.3: Accounting for a hosting arrangement



1.4.1 Hosting arrangement as a cloud computing transaction

In this publication, a “cloud computing arrangement” is a hosting arrangement from a customer’s perspective in which the customer cannot take possession of the hosted software. If the customer can take possession of the hosted software, the contract both the service of hosting and a license. If the customer cannot take possession, the contract is accounted for as a service only contract, that is, a cloud computing arrangement. The customer can take possession of the software only if both of the following criteria in ASC 350-40 are met:

a. It has the right to take possession of the software at any time during the hosting period without incurring a significant penalty.

b. It can feasibly run the software on its own hardware or contract with a third party unrelated to the software vendor to host the software.

If one or both of these criteria are not met, the customer does not take possession of the software.

Under the guidance in ASC 350-40, two factors must be considered when evaluating whether a customer would incur a significant penalty if it takes possession of software in a hosting arrangement:

- Will the customer incur a significant cost when it takes possession of the software?
- Will the software become significantly less useful or valuable if it were used separately from the hosting service?

ASC 350-40-15-4A

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.

ASC 350-40-15-4B

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.

If the customer does not have the right to take possession of the software, it must account for the contract as a cloud computing arrangement. Because the vendor is only allowing the customer to access the software as needed, the arrangement is a service contract rather than a contract to purchase or license software. Therefore, the customer actually accounts for the *service* it receives from the vendor hosting the software, and not the software itself. However, it must also account for the costs of implementing the cloud computing arrangement, as outlined in ASC 350-40. These set-up and implementation costs are capitalized or expensed, and then subsequently accounted for, as discussed in Section 5.

If the customer has the ability to take possession of software under a hosting arrangement, it should account for the arrangement as internal-use software, as discussed in Section 2, and not as a cloud computing arrangement.

ASC 350-40-15-4C

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.

ASC 350-40-15-4D

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.

ASC 350-40-25-18

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.

At the crossroads: Accounting for implementation costs in a cloud computing arrangement that is a service contract

In August 2018, the FASB issued ASU 2018-15, *Customer's Accounting for Implementation Costs Incurred in a Cloud Computing Arrangement That Is a Service Contract (a consensus of the FASB Emerging Issues Task Force),* which provides guidance on accounting for costs incurred by a customer when implementing a cloud computing arrangement that is considered a service contract. Prior to the amendments in the ASU, there was diversity in practice because no explicit GAAP existed on how a customer should account for these implementation costs. The amendments require an entity that incurs costs to implement a cloud computing arrangement to account for those costs in a manner similar to costs for internal-use software.

Public business entities should apply the amendments in ASU 2018-15 in fiscal years beginning after December 15, 2019, including interim periods within those fiscal years. All other entities should apply the amendments in fiscal years beginning after December 15, 2020 and in interim periods within fiscal years beginning after December 15, 2021. Early adoption is permitted for all entities in any annual or interim period if the financial statements have not already been issued or made available for issuance.

Entities should apply the amendments either (1) retrospectively, recognizing the cumulative effect of applying the amendments in the financial statements in the opening retained earnings of the earliest period presented, or (2) prospectively to costs for activities performed on or after the adoption date of the ASU.

1.4.2 Hosting arrangement as software as a service transaction

In this publication, a "software as a service (SaaS) arrangement" is a hosting arrangement from a vendor's perspective in which the customer cannot take possession of the hosted software. When an entity develops software to be used in a SaaS arrangement, it must determine whether to account for the software as internal-use software or as software that will be sold or marketed externally. An entity should

account for software used in a SaaS arrangement as software that will be sold if both of the following two criteria are met:

- The customer has a contractual right to take possession of the software at any time during the hosting period without incurring a significant penalty.
- The customer can either run the software on its own hardware or can contract with a party unrelated to the vendor to host the software.

In determining whether a penalty that would be incurred to take possession of hosted software is "significant," an entity must consider if the customer can both (1) take possession of the software without incurring significant costs, and (2) use the software separately from the hosting arrangement without significantly decreasing the software's value or usefulness.

If the criteria are expected to be met at any time during the arrangement, the costs of developing the software are accounted for as software that will be sold, leased, or marketed under ASC 985-20, as discussed in Section 4.

If both of these criteria are not expected to be met during the arrangement, the software is used to provide a service (SaaS) rather than being sold, leased, or marketed. The costs to develop the software used in the SaaS contract are accounted for as internal-use software in ASC 350-40, as discussed in Section 2.

ASC 985-20-15-5

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.

ASC 985-20-15-6

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

ASC 985-20-15-7

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).

1.5 Other costs of technology

U.S. GAAP also includes specific guidance for two other types of technology-related costs that entities might incur: business and technology reengineering costs, and website development costs. While neither of these types of costs is directly related to software, they are connected with the entity's technology environment and may include software-related costs that should be accounted for under other ASC Topics.

1.5.1 Business process reengineering and technology transformation

Entities often perform information technology transformation projects as well as business process reengineering projects to better align their businesses with their existing technology or to update both their business processes and their existing technology. An entity may either hire a consultant with specialized skills in the area being transformed or reengineered or perform these projects in-house if the expertise exists. A consultant may provide services that are limited to business process reengineering, or may provide services that encompass an entire project, including internally developing or purchasing software as well as updating hardware and other fixed assets. When a consultant assumes an entire project, an entity must track and differentiate between reengineering costs and costs incurred during other phases of the project.

Examples of activities that may be involved in a business process reengineering or an information technology transformation project include

- Preparing a request for proposal
- Assessing and documenting the current state of business processes and information technology, including mapping, developing an "as-is" baseline, flow charting, and determining the current business process structure
- Reengineering the entity's business processes to increase their efficiency or effectiveness
- Restructuring the workforce to determine the number and types of employees that will be necessary for the reengineered business processes

Costs of the reengineering process itself are accounted for under ASC 720-45 and are generally expensed as incurred, regardless of whether they are provided by a third party or internally. The guidance on accounting for these costs is outside of the scope of this publication.

1.5.2 Website development costs

An entity might incur a variety of costs to develop a website. Costs relating to developing or obtaining software as a part of website development are accounted for as software costs, following the guidance for internal-use software or software to be sold or marketed, as appropriate (see Section 1.1 and 1.3 for more information).

Other costs of developing a website, including costs for registering an internet domain, developing graphics and content, and operating the website, are accounted for under ASC 350-50. These costs are capitalized or expensed based on the purpose of the activity and the stage of development of the website, as outlined in the guidance in ASC 350-50. Website development costs are outside of the scope of this publication.

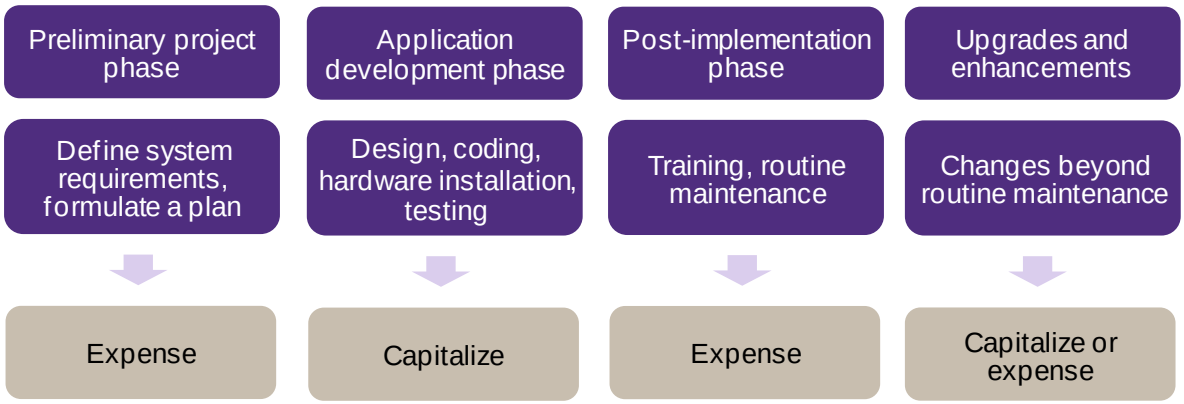
2. Internal-use software

Costs incurred in developing internal-use software are either capitalized or expensed, depending on both the nature of the costs and the phase of development in which they are incurred (Figure 2.1 outlines the stages of internal-use software development and the accounting for each phase, as defined in ASC 350-40.) Costs incurred for implementation activities during the preliminary and post-implementation phases of a project, for instance, are expensed as incurred, while costs incurred during the application development phase are generally capitalized. Entities that incur costs to upgrade or enhance existing software are required to capitalize these costs if the changes result in additional functionality, but to expense costs if the software's functionality is not improved.

When it is no longer probable that software being developed will be used, capitalization should cease and the asset should be evaluated for impairment as discussed in Section 2.5.

Capitalization of development costs should cease, and amortization of those costs should begin, when the software is ready for its intended use. The costs are amortized over the estimated useful life of the software as discussed in Section 2.4.

Figure 2.1: Accounting for costs incurred to develop internal-use software



Entities should consider the activities being performed when determining the software project's stage of development. The guidance in ASC 350-40-55-3 outlines activities and processes that fall into each stage of development. For example, a software development project would be in the preliminary phase while the entity determines the performance and system requirements for the software as well as evaluates potential methods for meeting those identified requirements.

ASC 350-40-55-3

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.

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.

c. Postimplementation–operation stage:
    1. Training
    2. Application maintenance.

ASC 350-40-55-4

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.

Grant Thornton insights: Evaluating costs when project stages are unclear

Many entities conduct multiple project stages simultaneously when developing internal-use software. For instance, a software project might revert back to a previous stage due to various developmental issues, which creates complexities when applying the guidance in ASC 350-40.

ASC 350-40-55-3 outlines which processes fall into each stage of product development, but the FASB has acknowledged that these stages and processes do not always occur linearly in the order presented in the guidance. When project stages fluctuate or are not clearly distinguishable, we believe that entities should focus on the type of activity being conducted rather than on the project stage.

For example, an entity that commits to funding a software project from start to finish and selects a specific project from among alternatives would consider the project to be in the application

development stage, and would capitalize development costs incurred after that point. In contrast, an entity that commits to funding a project before evaluating project alternatives and selecting a specific solution would account for the costs incurred during the selection phase as part of the preliminary project stage, based on the nature of those costs, even though the entity had already committed to the project.

2.1 Preliminary project stage

The preliminary project stage is the period during which an entity determines the performance and system requirements for the proposed internal-use software and evaluates potential methods for meeting the project's requirements with the resources available. In general, an entity is assessing its needs and evaluating various alternatives in the preliminary project stage. This stage includes defining and creating a plan to develop and / or implement the proposed internal-use software, as well as selecting vendors if the software will be purchased or consultants will assist with implementing the plan. Activities undertaken in the preliminary project stage are analogous to research and development activities. Entities should expense all costs incurred during this stage.

The ASC Master Glossary outlines the types of activities an entity may perform during the preliminary project stage.

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.

The following example shows the types of costs incurred during the preliminary stage of a project.

Accounting for costs incurred during the preliminary project stage

A trade association tracks membership data using a database stored on its internal network. While this system is functional, the association determines that members would benefit greatly from online access to the data. The association performs initial outreach to obtain information about the cost of providing its members online access. Based on the initial data collected, the association decides to move forward with the project and allocates money for the project in its annual budget.

The association sets up a committee comprising certain members of management as well as members of the association to develop a list of requirements and desired functionality for the proposed online membership system. The committee evaluates existing off-the-shelf products as well as proposals to develop a customized solution from outside vendors. After several months, the committee selects a vendor that specializes in online membership software programs to provide a customized interface for members to access the system.

All expenses incurred during this preliminary project period would be expensed as incurred. Even after the association committed to the project and allocated a portion of its budget to the changes, the nature of the expenses incurred by the committee related to determining the performance and system requirements for the proposed software and to evaluating potential methods for meeting those requirements. As a result, all costs incurred in this example are considered preliminary-project-stage costs and are expensed as incurred.

2.2 Application development stage

Typically, when the preliminary project stage is completed and an entity decides to move forward with a software project and commits to funding the project through completion, the project enters the application development stage. The application development stage includes (1) designing the development path, including software configuration and software interfaces, (2) coding, (3) installation and (4) testing, including parallel processing.

During this stage, the entity is required to capitalize internal and external costs to develop internal-use software. Costs to develop or obtain software that provides access to or conversion of old data by new systems should also be capitalized. In order to capitalize costs, management must authorize the project and be sufficiently committed to the point that it is probable that the software will be fully developed and used to perform its intended function. Authorization to develop the software may be explicit, as in the case of an executed contract with a third party, or it may be implicit, such as budgeting for expenses that will be incurred to develop the software.

ASC 350-40-25-12

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.

As previously noted, in determining whether to capitalize or expense costs for developing internal-use software, an entity must consider not only the project phase, but also the nature of the expenses incurred and whether they relate to developing the software. Entities should capitalize any costs that directly relate to the development of the software, including direct costs of materials or services provided externally, and payroll and payroll-related costs for employees' time spent directly on the development of the software. The amounts capitalized should include interest costs incurred while developing the software in accordance with ASC 835-20, *Interest: Capitalization of Interest*.

ASC 350-40-25-2

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

ASC 350-40-25-3

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

ASC 350-40-25-4

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

ASC 350-40-25-5

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.

While a project is in the application development stage, the entity should carefully review the nature of the costs incurred, as costs that do not directly relate to the development of the software are not capitalizable. For example, entities should expense all costs to train employees on how to use the software, even if the costs are incurred during the application development stage.

Transferring data from existing software to newly developed or purchased software is an activity that usually happens during the application development phase. If the data is converted between systems using manual processes, the cost to do so should be expensed as incurred. However, if the data will be converted using software, the cost of developing or purchasing software to perform the conversion should be capitalized as software development costs. All costs to manually convert data from the old system, including purging existing data, reconciling data between the systems, or importing old data into the new system are expensed as incurred.

Costs of converting software

Entity A and Entity B are both developing internal-use software to replace existing software. Both projects are in the application development phase. Both entities need to convert data from their existing software to the software being developed.

Entity A needs to convert data from its existing inventory tracking system to the new software it has developed. Entity A tasks some of its inventory management staff to manually type the data from the old system into the new system. The cost of the employees who manually enter the data, as well as the employees who review the data once input, is an expense in the period incurred. Although the project is in the application development phase, the activity of manually converting data is not a captializable cost.

Entity B needs to convert data from its existing customer data management system to the new software it is developing. Entity B tasks its project team to build a software program that will automatically extract the data from the existing system, reformat it, and import it into the new system. The cost to develop the conversion software would be capitalized subject to the guidance in ASC 350-40. Any manual processes performed in addition to the development of the software, such as reconciling between the old and new system once the software has completed the conversion, are expensed as incurred.

ASC 350-40-30-1

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.

Only direct costs of the internal-use software are eligible for capitalization. General and administrative or overhead costs are expensed as incurred. Examples of such costs include depreciation of the computers or an allocation of rent to the space used by the computer programmers.

ASC 350-40-30-3

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

The following example illustrates how an entity would account for costs incurred during the application development stage in an internal-use software project.

Accounting for costs incurred during the application development stage

A manufacturing entity purchases equipment to use in its manufacturing process and authorizes an internal team of engineers to customize the software embedded in the equipment.

The entity has previously expensed the costs incurred to determine the performance and system requirements for the project, as well as costs involved in evaluating whether to customize the software in-house or hire a third-party consultant. Management commits to moving forward with the project using the internal team by including the associated costs in its budget.

During the current year, the entity incurs $1 million in total payroll and benefits costs for the software engineers, but only 25 percent, or $250,000, of that amount is directly related to working on the software project. The entity incurs $50,000 in costs to train employees on how to operate the customized equipment. The entity has also allocated a portion of overhead for rent and utilities, totaling $100,000, to the software engineering group based on the square footage of the engineering space divided by the total square footage of the leased property. Of that amount, the entity allocates the same proportion as payroll and benefits for the engineers (25%) to the application development stage.

The entity capitalizes payroll and benefits related to the employees' time spent directly working on the project. In other words, the software engineers are required to track their time, and their salary is allocated proportionately to this development stage based on the time spent coding and testing the software. The entity expenses training and overhead costs, including the allocated rent and utilities, as incurred.

The table below summarizes the costs incurred in the application development phase during the year, as well as the accounting treatment for each type of cost.

Type of cost	Amount allocated	Capitalize or expense?
Engineering payroll	$250,000	Capitalize
Training	$ 50,000	Expense
Allocated overhead for rent and utilities	$ 25,000	Expense

2.2.1 Acquiring a software license for internal use

An entity may license software for internal use from a third party. When this happens, an entity should first look to the guidance in Section 1.4 for identifying hosted software to determine if the license is an asset or a service. If it is not a service (not hosted software), the software license is recognized at cost as an intangible asset that is amortized over its useful life. If the entire license fee is not paid at license inception, for example, if it will be paid over the license term, the asset should be recorded at total contractual cost (or allocated cost in a multiple-element arrangement, see Section 2.2.2), and a liability is recognized for future payments under the license agreement. The initial measurement guidance in ASC 350-40-25 refers to the general intangible assets guidance in ASC 350-30, *Intangibles: General Intangibles Other Than Goodwill,* for recognition and measurement of software licensed from third parties. The intangible assets guidance then refers to the asset acquisition guidance in ASC 805, *Business Combinations,* specifically 805-50-15-3 and 805-50-30-1 to 30-4, which requires an acquired intangible asset to be recognized at cost.

ASC 350-40-25-17

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.

Licensing software for internal use

An entity purchases a two-year software license from a vendor for inventory tracking software for $120,000. The entity pays $60,000 when it enters into the license and will pay the remaining $60,000 at the end of the first year. At inception of the license term the entity records the following entry to recognize the license and the liability for payment:

DR: Software license	$120,000	
CR: Cash		$60,000
CR: Software license payable		$60,000

Each month the entity records entries to amortize the software license:

DR: Amortization of software license	$10,000	
CR: Software license		$10,000

2.2.2 Multiple elements included in purchase price

An arrangement to purchase internal-use software or a cloud computing arrangement may include multiple elements, such as a license, implementation services, training, and maintenance. An entity that purchases software or hosting services from a third party in a multiple-element arrangement should

allocate the consideration paid to the various elements based on their relative stand-alone selling price. The entity should capitalize only the costs that meet the capitalization criteria in ASC 350-40. An entity should not default to allocating the total cost to the various elements based on each element's stated price within the multiple-element contract.

ASC 350-40-30-4

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.

Grant Thornton insights: Estimating stand-alone selling price

In some cases, the elements of a multiple-element arrangement are not sold on a stand-alone basis or if sold on a stand-alone basis, the customer does not have access to the stand-alone selling prices. When observable stand-alone selling prices are not available, the customer in a must estimate the stand-alone selling prices of each of the elements. While ASC 606, *Revenue from Contracts with Customers,* includes guidance on how a *vendor* should estimate the stand-alone selling price, U.S. GAAP does not include explicit guidance on how a *customer* should estimate these stand-alone selling prices. This does not mean that an entity can default to using the prices stated in the contract. Rather, an entity must apply judgment to estimate the stand-alone selling price of each element in the contract and allocate the costs based on those estimates.

We believe it would be reasonable to consider the guidance in ASC 606 when determining the stand-alone selling price of elements in a multi-element arrangement. That guidance requires an entity to consider all information reasonably available, and to maximize observable inputs. It also cautions that while a stated selling price, such as a list price, may be the stand-alone selling price, an entity should not presume that it is without considering other available information.

The following example demonstrates the application of the guidance for allocating the purchase price.

Allocating the purchase price

An entity enters into a contract with a third-party vendor to license a software product and to purchase one year of post-contract customer support (PCS) and 20 hours of training, all in exchange for $20,000. The software is for internal-use only. The vendor does not sell the software license or training on a stand-alone basis but offers PCS at a renewal rate of $4,000 per year. PCS services offered by

competitors with similar software products are also similarly priced, so the entity concludes that $4,000 is a reasonable estimate of the stand-alone selling price of the PCS.

Although the third-party vendor does not sell training on a stand-alone basis, the entity identifies several other companies that offer similar training for the licensed software at stand-alone prices ranging from $90 to $110 per hour. Based on this data, the entity concludes that $100 per hour is an appropriate estimate of the stand-alone selling price for the training.

The entity then uses the residual method to estimate the stand-alone selling price of the software, allocates the transaction price to each of the contractual elements on a relative stand-alone selling price basis, and accounts for each element based on the applicable accounting guidance, as shown below.

Product	Allocated transaction price	Accounting treatment
PCS	$ 4,000	Expense over the contract term
Training	2,000	Expense as incurred
Software	14,000[1]	Capitalize and amortize[2]
Total	$ 20,000	

[1] This amount is the residual value of the $20,000 contract price minus the $4,000 stand-alone selling price for PCS and the $2,000 stand-alone selling price for training.

[2] The software license is accounted for at cost and amortized over its useful life, in a manner consistent with the acquisition of an intangible asset, as discussed in Section 2.2.1.

2.2.3 Impact of new software development activities on existing software

When an entity undertakes to develop or purchase new software that will replace existing software, it should consider whether a change in the useful life of the existing software is required. If the entity determines that the useful life of the existing software has changed, it should account for that change under ASC 250, *Accounting Estimates and Error Corrections,* as a change in accounting estimate. That is, it should update its estimate of useful life and revise the amortization going forward only.

ASC 350-40-25-15

New software development activities shall trigger consideration of remaining useful lives of software that is to be replaced. When an entity replaces existing software with new software, unamortized costs of the old software shall be expensed when the new software is ready for its intended use.

The development of new software may indicate that there is a change in the useful life of the existing software that is being replaced, for example, if the new software will be ready for use before the end of the existing software's remaining useful life. In some cases, an entity might need to reassess the life of the asset for the existing software. Any resulting change in useful life should be accounted for prospectively as a change in accounting estimate under ASC 250. If there are any unamortized costs associated with the old software when the new software is ready to be used, the entity must expense those costs at that time, as demonstrated by the following example.

Developing new software to replace existing software

Entity A uses internally developed software to process certain transactions. As of December 31, 20X1, the software has a remaining life of three years. Entity A is working on making significant changes to its operations which will change the way these transactions are processed. As a result, the entity determines that the existing software would require significant modifications in order to continue to be used under the modified operations. In the third quarter of 20X1, the entity decides to hire a third party to design and develop a new software product to meet the requirements of the revised operating processes, which must be ready by the time the new processes are implemented, January 1, 20X3. As a result, the entity revises its estimate of the remaining useful life for the existing software to one year as of December 31, 20X1. If the new software is implemented before the existing software's remaining useful life expires, the entity would expense any remaining unamortized costs.

2.3 Post-implementation operation stage

Once all substantial testing on a software project is completed and the software is ready to be used, the development stage ends, and the post-implementation operation stage of the project begins. Entities may no longer capitalize costs incurred related to the software, such as training and maintenance in the post-implementation stage, but instead must expense them as incurred. However, upgrades and enhancements that meet certain criteria may be eligible for capitalization as software development costs if they meet the application development stage criteria.

ASC 350-40-25-14

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.

2.3.1 Training and maintenance

Consistent with the requirements for capitalizing property, plant and equipment, only costs incurred to bring an asset to its "intended use" are capitalized in the case of software development. Therefore, costs incurred in the post-implementation period are generally expensed as incurred, regardless of whether they are internally developed or provided by a third-party.

One typical cost in the post-implementation phase is the cost of routine maintenance. Routine maintenance activities include keeping the software up to date, correcting errors, and other small changes to ensure it is functioning as intended. Another cost often incurred in the post-implementation phase is the cost of training employees on how to use the software.

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.

ASC 350-40-25-6

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

2.3.2 Upgrades and enhancements

Sometimes an entity makes changes to internal-use software that go beyond routine maintenance. When this occurs, the entity must understand the extent of the changes to determine whether the software has been enhanced or improved. If it is probable that the improvements will result in additional functionality, the entity should capitalize costs incurred for the upgrades or enhancements. Changes or improvements result in additional functionality if, after the modification, the software can perform tasks that it could not previously perform. Entities are required to segregate costs incurred to enhance or upgrade functionality from costs of routine maintenance, which must be expensed as incurred, to avoid improperly capitalizing costs related to maintaining the software.

ASC 350-40-25-7

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.

For internal costs incurred to modify software, entities should develop a reasonable, cost-effective way of separating costs for maintenance from the costs for upgrades and enhancements. If it is not possible to reasonably separate these costs, the entity should expense the costs as incurred for minor upgrades and enhancements as well as for maintenance costs.

For example, if the individuals responsible for software maintenance are also responsible for minor upgrades and enhancements, it may be impracticable or cost prohibitive for them to accurately track time spent on upgrades and enhancements. It might also be challenging to identify whether work done for a minor upgrade will result in additional functionality, particularly if upgrades are often started but not completed, which might call into question whether the effort would result in additional functionality.

ASC 350-40-25-10

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.

Grant Thornton insights: 'Minor' upgrades and enhancements

The practicality of separating internal costs is only a factor when considering how to account for relatively minor upgrades or enhancements, and should not be used to justify the lack of appropriate processes and procedures to track the necessary data when developing a more significant upgrade.

For example, if a team of developers is spending a majority of their time developing a new version of an existing software that will significantly expand its functionality, the entity is not exempt from capitalizing application development costs, even if the developers are also responsible for maintenance. Instead, the entity must develop procedures to capture the individuals' time spent on the project in order to capitalize costs incurred to develop the new version of the software.

When a single contract with an external software developer includes both software maintenance and upgrades or enhancements, an entity should first determine whether the upgrades and enhancements are either "specified," meaning the entity has knowledge of, or expects, a future upgrade or enhancement, or if they are "unspecified," meaning the entity is entitled to any upgrades when and if they are available but does not know the timing or nature of the upgrades.

If the upgrade right is specified, the entity should allocate the cost between the maintenance services and the specified upgrades based on their relative stand-alone selling price. Costs allocated to the specified upgrades should be evaluated for capitalization using the guidance for internal-use software, by project stage and activity. Any costs capitalized for the upgrade are amortized over the contract period. Costs allocated to maintenance services should be expensed over the contract period.

If the contract includes unspecified upgrades delivered on a when-and-if-available basis, then the entity should expense both the upgrade and maintenance costs over the contract period on either a straight-line basis or another systematic and rational basis that more accurately represents the pattern of delivery of the services.

ASC 350-40-25-11

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.

Grant Thornton insights: Determining whether upgrades and enhancements are specified

When purchasing software from a third party, an entity may need to use significant judgment to determine whether the vendor has provided a specified upgrade since the right to an upgrade is often not explicitly stated in the arrangement. The right to a specified upgrade may be implied based on the information about future product enhancements that the vendor communicates to the entity while negotiating the arrangement. If the vendor provides sufficient details about the features and functionality of an enhancement, the entity may have an expectation that the enhancement will be provided. If the entity has knowledge of, or expects, a future upgrade or enhancement, the upgrade right is specified. However, if the entity determines that it does not need and will not use the specified upgrades or enhancements, it may choose not to take advantage of the upgrade. In such cases, the entity should expense the entire amount paid to the vendor as maintenance costs.

The following example illustrates how an entity might account for external costs incurred for a specified upgrade to an internal-use software product.

Software upgrade is specified

Entity B contracts with Vendor C to purchase a perpetual license of Software 2.0 for internal use, one year of PCS, and the right to receive an upgrade to Software 3.0 when the upgrade is released, for total consideration of $125,000. The Software 3.0 upgrade is released one year after Entity B enters into the agreement. Entity B expects to use the software for four years.

Because the arrangement specifies that Entity B will receive three specific elements, the total consideration must be allocated among those elements based on their stand-alone selling prices, as shown in the following table. As discussed in Section 2.2.1, Entity B determines the stand-alone selling prices consistent with the guidance in ASC 606, maximizing observable inputs.

	Stand-alone selling price	Allocation percentage	Consideration allocated
Software 2.0	$ 96,000	64%	$ 80,000
Upgrade to Software 3.0	$ 36,000	24%	$ 30,000
PCS	$ 18,000	12%	$ 15,000
	$150,000	100%	$125,000

Entity B capitalizes $80,000 for Software 2.0 and $30,000 for the upgrade, resulting in a total of $110,000 capitalized as internal-use software. The costs capitalized related to Software 2.0 are amortized beginning on the date it is delivered to the entity. Since the upgrade to Software 3.0 is not delivered until a year later, Entity B does not start amortizing the amount capitalized for the upgrade until it receives the upgrade a year after the agreement is signed. These two capitalized elements therefore have different amortization periods. Software 2.0 will be amortized over four years, and the upgrade to Software 3.0 will be amortized over three years.

Entity B allocates $15,000 of the transaction price to PCS, which is expensed over the one-year PCS period.

The following example illustrates how an entity might account for unspecified software upgrades.

Unspecified software upgrades

Entity D purchases software and one year of PCS from Vendor E. Included in the PCS is the right to receive all future upgrades when and if the vendor releases the upgrades, as long as the customer has a current arrangement for PCS. As of the contract renewal date, Entity D is not aware of any specific features or functionality that will be added to the software. As a result, the right to receive unspecified upgrades on a when-and-if available basis is not considered a specified upgrade or enhancement and the PCS is accounted for as a single element.

Accordingly, Entity D allocates the purchase price between the software and PCS on a relative stand-alone selling price basis. Entity D capitalizes the software under ASC 350-40, and expenses the amount allocated to the PCS ratably over the one-year term.

At the end of the first year, Entity D renews the PCS services for an additional year in exchange for additional consideration. As of the renewal date, Entity D determines that it is unaware of any specific features or functionality to be added to the software in any pending upgrade. Because the right to receive unspecified upgrades on a when-and-if-available basis is not considered a specified upgrade or enhancement, Entity D ratably expenses the amount of consideration over the one-year service period.

2.4 Amortization of capitalized internal-use software costs

Capitalized internal-use software development costs should be amortized on a straight-line basis, unless another systematic and rational basis better represents how the entity expects to benefit from using the software. If an entity identifies a significant change to the expected pattern of use of the software, it should update the amortization to reflect that change prospectively, in accordance with the guidance on a change in accounting estimate in ASC 250.

ASC 350-40-35-4

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.

Estimating the useful life over which the internal-use software costs will be amortized is analogous to estimating the amortization or depreciation period for other intangible and tangible assets. ASC 350-40-35-5 outlines factors that an entity should consider when assessing the estimated useful life of internal-use software.

ASC 350-40-35-5

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.

Grant Thornton insights: Determining the amortization period

The process of estimating a period to use for amortizing the costs incurred to develop internal-use software is subjective and requires entities to evaluate the particular facts and circumstances of each situation. Because software is generally more prone to obsolescence and changing technology than many other tangible or intangible assets, entities need to exercise judgment when estimating the amortization period. The pace of change in the software and technology industry often results in a

relatively short useful life for most software.

If the expected use of the software changes after the amortization period is established, an entity may need to adjust the useful life of the software. These changes might include modifying the software, developing or purchasing new software that will replace the existing software, or ceasing to use the software altogether.

An entity should consider the unit of account when determining the point when amortization should begin. Each component of the software, or "module" should be considered separately as its own unit of account, unless the functionality of a single module depends on the completion of other modules. If a module is ready for its intended use, testing is complete, and is not interdependent with other modules, the entity should begin amortization of the costs of that module. If the functionality of a single module entirely depends on completing another module or modules, amortization should begin only when the interdependent modules are completed and ready for the intended use.

ASC 350-40-35-6

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.

2.5 Impairment of capitalized internal-use software cost

Given the constant evolution of software products, software generally has a relatively short useful life before it becomes obsolete. The unexpected introduction of new software and technology can also have a dramatic impact on the usefulness of existing software, including software still in development. As a result, entities should continuously evaluate whether events or changes in circumstances might trigger an impairment of any internal-use software programs. The guidance in ASC 350-40-35-1 outlines the following four factors that might indicate an entity may not recover the carrying amount of capitalized software either in use or in development:

- The software being used or developed is no longer expected to be of use to the entity.
- A significant change occurs in the extent or manner in which the software is being used or is expected to be used if the software is in development.
- A significant change is planned or is made to the software.
- Costs to develop or modify the software significantly exceed the amount originally expected.

If one of these conditions exists, the capitalized software costs should be assessed for impairment using the measurement and impairment guidance for property, plant, and equipment in ASC 360-10-35. An entity should (1) compare the expected cash flows from the asset to its carrying value to determine whether the carrying amount is recoverable and, if not recoverable, (2) recognize an impairment loss for the amount of carrying value that is greater than fair value.

ASC 350-40-35-1

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.

ASC 360-10-35-17

An impairment loss shall be recognized only if the carrying amount of a long-lived asset (asset group) is not recoverable and exceeds its fair value. The carrying amount of a long-lived asset (asset group) is not recoverable if it exceeds the sum of the undiscounted cash flows expected to result from the use and eventual disposition of the asset (asset group). That assessment shall be based on the carrying amount of the asset (asset group) at the date it is tested for recoverability, whether in use (see paragraph 360-10-35-33) or under development (see paragraph 360-10-35-34). An impairment loss shall be measured as the amount by which the carrying amount of a long-lived asset (asset group) exceeds its fair value.

2.5.1 Abandoned or suspended projects

If an entity stops using internal-use software, the software should be accounted for as abandoned following the guidance for long-lived assets to be abandoned in ASC 360. Under that guidance, an asset is abandoned when it ceases to be used. An entity that decides to abandon an internal-use software asset before the end of its useful life treats the decision as a change in estimate of the useful life. That is, it updates the useful life based on the planned abandonment date and updates the amortization of the software prospectively.

ASC 350-40-35-2

Paragraphs 360-10-35-47 through 35-49 requires that the asset be accounted for as abandoned when it ceases to be used.

Sometimes there are indicators during development that the software will not be completed. This could occur due to, for example, budget constraints, changes in the business, or changes in the assessment of

the feasibility of the project. It could also be a result of problems encountered in executing the project plan, such as difficulties in developing the software to meet the system requirements, incurring costs in excess of budget, or increased estimates to purchase the software from a third party. Or, the entity might choose to go another direction and as a result, make a decision to abandon the project before completion.

If there are indicators that it is probable that the software project will be abandoned, or if the entity makes the decision to abandon the software during development, before capitalization has ceased, the entity should stop capitalizing costs and assess whether previously capitalized costs might be impaired. The capitalized costs of the abandoned project should be written down to the lower of the software's carrying amount or fair value, if any, minus any costs that would be incurred to sell the in-process software. ASC 350-40-35-3 includes a rebuttable presumption that incomplete software has a fair value of zero, so that abandoning a project generally results in writing off all previously capitalized costs related to that project.

ASC 350-40-35-3

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.

Grant Thornton insights: Costs incurred for software not completed

Software projects can be complex and often involve several iterations that are started but are not developed to completion. For example, an entity may begin the application development phase for developing software to track its customer relationships, only to subsequently discover a new product on the market that performs the same function. Or, an entity may begin the application development phase of a software project only to find that the project needs to revert back to the planning phase. In fact, it is not uncommon for an entity to revert back to the planning phase several times before arriving at a software application that is viable. Entities need to track the costs of developing each software

project and capitalize those costs based on the guidance for each development phase as a project progresses. If an entity determines that the software being developed will not be completed, the costs incurred to date in the development of that software should be evaluated and written down to the lower of carrying amount or fair value less cost to sell in accordance with ASC 350-50-35-3. If the costs to date are incurred for activities that do not have use to a future project, it is likely that their value is zero.

In other cases, an entity might suspend development activities related to a software project only for a time, with the intention of completing the project at a later date. An entity that can reasonably assert that the project will still be completed may retain the capitalized costs as an asset. However, if the entity is capitalizing interest related to the project it must stop interest capitalization until it resumes developing the software.

ASC 350-40-30-2

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.

2.6 Presentation and disclosure

ASC 350-40 does not provide any presentation or disclosure requirements for internal-use software. Instead, the presentation and disclosure requirements for internal-use software are primarily covered by the general disclosure requirements found in other Topics of the Codification, including the following:

- ASC 275, *Risks and Uncertainties* – Estimated useful life of intangible assets, including software, if a change in the estimate of the useful life would be material to the financial statements
- ASC 730-10, *Research and Development* – Total research and development costs expensed in each period, including research and development costs incurred for a computer software product to be sold, leased, or otherwise marketed
- ASC 235, *Notes to the Financial Statements* – Accounting policies, including the policy for the amortization of intangibles
- ASC 360-10, *Property, Plant and Equipment* – Amortization expense for the period, balance of major classes of depreciable assets, accumulated amortization at the balance-sheet date, and a general description of the method or methods used in computing depreciation and amortization with respect to major classes of depreciable and amortizable assets

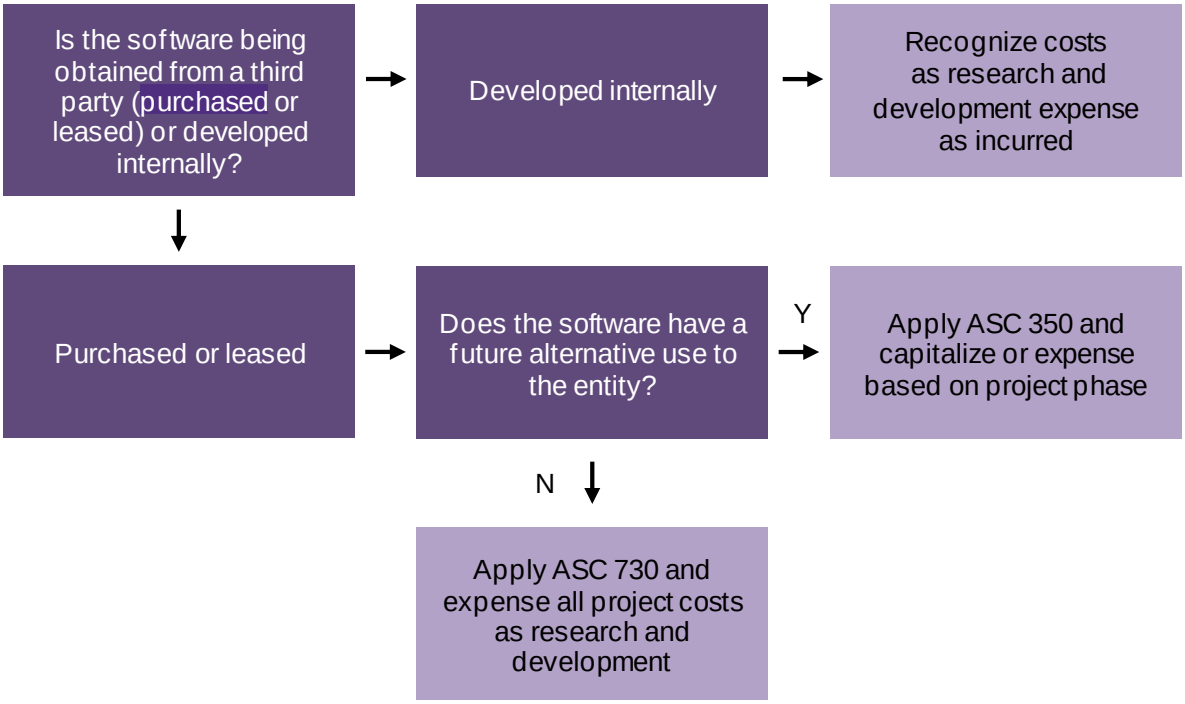
3. Research and development software

The accounting for software used for research and development is based on whether the entity obtains the software from a third party or develops it internally. If it is obtained from a third party, the software is accounted for based on whether it has an alternative use in the future beyond the existing research and development project.

To determine whether software falls within the scope of the guidance on research and development in ASC 730, refer to the discussion in Section 1.2.

The following figure provides a summary of the accounting for software used in research and development.

Figure 3.1: Summary of accounting for software used in research and development



3.1 Software that is purchased or leased

The accounting for software that an entity purchases or leases to be used in research and development depends on whether or not the software has an alternative future use to the entity. Software that has an alternative future use beyond the current research and development project should be accounted for as internal-use software under ASC 350-40, as discussed in Section 2. Amortization of the capitalized software should be recognized as a research and development cost on the income statement.

The costs of purchasing or leasing software that does not have an alternative future use, either as part of another research and development project or in some other capacity, should be expensed as incurred following the guidance in ASC 730.

ASC 730-10-25-2

Elements of costs shall be identified with research and development activities as follows…:

c. Intangible assets purchased from others. The costs of intangible assets that are purchased from others for use in research and development activities and that have alternative future uses (in research and development projects or otherwise) shall be accounted for in accordance with Topic 350. The amortization of those intangible assets used in research and development activities is a research and development cost. However, the costs of intangibles that are purchased from others for a particular research and development project and that have no alternative future uses (in other research and development projects or otherwise) and therefore no separate economic values are research and development costs at the time the costs are incurred.

ASC 730-10-25-3

When software for use in research and development activities is purchased or leased, its cost shall be accounted for as specified by (c) in the preceding paragraph and paragraph 730-10-25-1. That is, the cost shall be charged to expense as incurred unless the software has alternative future uses (in research and development or otherwise).

3.2 Software that is developed internally

The costs of internally developing software that is used for research and development should be recognized as research and development expense as the costs are incurred. ASC 730 offers no exception for entities to capitalize these costs. The alternative future use test that applies to costs incurred to purchase research and development software (see Section 3.1) does not apply when an entity develops software internally. Costs of developing research and development software internally are expensed, without regard to the phase of the software development project.

ASC 730-10-25-4

Development of software to be used in research and development activities includes costs incurred by an entity in developing computer software internally for use in its research and development activities, are research and development costs and, therefore, shall be charged to expense when incurred. The alternative future use test does not apply to the internal development of computer software; paragraph 730-10-25-2(c) applies only to intangibles purchased from others. This includes costs incurred during all phases of software development because all of those costs are incurred in a research and development activity.

3.3 Disclosure

Under the guidance in ASC 730, an entity is required to disclose total expense recognized for research and development activities for each period included in the financial statements. Amounts related to research and development costs incurred for software to be sold are disclosed as a part of the total research and development expense.

ASC 730-10-50-1

Disclosure shall 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. Such disclosure shall include research and development costs incurred for a computer software product to be sold, leased, or otherwise marketed.

4. Software to be sold

Costs to produce or purchase software that an entity plans to sell, lease, or market are accounted for under the guidance in ASC 985-20. ASC 985-20 covers costs to internally develop as well as costs to purchase software when the software will be sold, leased or otherwise marketed. This includes software to be sold either as a separate product or as part of a product or process. The guidance in ASC 985-20 is also applied to costs for software enhancements. Software that is part of a cloud computing arrangement is accounted for under ASC 985-20 if both of the following criteria are met:

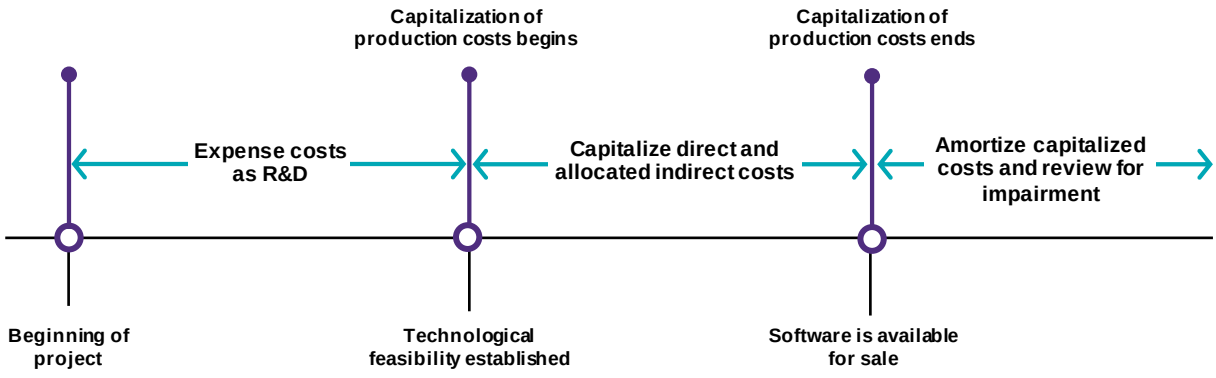
- The customer has a contractual right to take possession of the software without a significant penalty.
- The customer can feasibly either run the software on its own or contract with a third party to host the software.

The scope of ASC 985-20 excludes arrangements with customers to deliver software or a software system with significant production, modification, or customization. The costs of those arrangements are accounted for as costs to fulfill a contract under ASC 340-40.

Under ASC 985-20, the costs incurred to establish the technological feasibility (Section 4.1) of software that will be sold, leased, or marketed, either on its own or as part of another product, should be expensed as research and development when incurred, regardless of whether the software is developed internally or purchased from a third party. Once technological feasibility of the software is achieved, the entity capitalizes the remaining costs incurred to develop the software for sale, including costs of coding and testing the software. An entity should continue to capitalize costs until the product is ready to be sold or marketed to customers, at which time, amortization of the capitalized costs begins and the costs are subsequently reported at the lower of amortized cost or net realizable value.

The following figure illustrates the stages of a project to produce software that an entity plans to sell, lease, or market.

Figure 4.1: Stages of development of a software project



4.1 Technological feasibility

All costs of developing software prior to establishing its technological feasibility are research and development costs and are expensed as incurred.

Technological feasibility is achieved when an entity has completed all planning, designing, coding, and testing activities necessary to establish that the software product can be produced to meet its design specifications, including functions, features, and technical performance requirements. As described in ASC 985-20-25-1, this can be achieved through the use of either

- A detail program design
- The combination of a product design and working model, which have been confirmed for completeness by testing

The activities that an entity must perform to show that the software has reached technological feasibility vary, based on whether or not software development follows a "detail program design," as defined in the Codification's Master Glossary.

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**.**

ASC 985-20-25-1

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.

ASC 985-20-25-2

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.

When a software product contains multiple modules that are not individually saleable, technological feasibility is established at the level of the product as a whole, not for the individual modules. In order for capitalization to begin, the detail program design or the working model of the entire product must be complete.

ASC 985-20-55-7

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.

The criteria laid out in ASC 985-20-25-2 for technological feasibility are intended to provide an objective determination of when research and development activities end and production activities begin such that an entity should capitalize the costs subsequently incurred to develop software. Because ASC 985-20-25-2 sets forth an objective point that technological feasibility is established, an entity should follow those criteria rather than impose additional, more restrictive criteria.

ASC 985-20-55-6

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.

Grant Thornton insights: Determining if technological feasibility has been established

The determination of whether technological feasibility has been established requires a detailed understanding of the software being produced, the design plan, and the current stage in the development process. Generally, an entity's accounting department alone will not have adequate knowledge of the product and process to make these determinations without acquiring additional information. Therefore, discussions with developers are critical throughout the process to determine the appropriate accounting treatment at each phase of the project and to develop the appropriate processes and controls over the costs that will be expensed or capitalized during the development process. As a result, accounting for the costs incurred to develop software that is marketed or sold often requires a good deal of coordination and communication between the software developers and the entity's accounting department.

4.1.1 A detail program design exists

When an entity uses a detail program design for a software development project, technological feasibility has been established only after an entity has performed all of the following activities:

- Complete both the product design and the detail program design and established that the necessary skills, hardware, and software technology are available to produce the product
- Confirm the completeness of the detail program design and its consistency with the product design by documenting and tracing the detail program design to the product specifications
- Review the detail program design for high-risk development issues (for example, novel, unique, unproven functions and features or technological innovations), and resolved through coding and testing any uncertainties related to identified high-risk development issues

Sometimes, when an entity employs a detail program design, the three criteria used to confirm the existence of technological feasibility outlined above are not met until a working model is completed. When this is the case, the entity should establish that a working model is completed (as discussed in Section 4.1.3) before capitalization begins.

The Master Glossary offers definitions of the terms coding and testing to assist entities in applying this guidance.

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.

ASC 985-20-55-4

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).

ASC 985-20-55-5

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).

4.1.2 A detail program design does not exist

If an entity does not use a detail program design to develop a software product, it must perform both of the following actions to establish technological feasibility:

- Complete a product design and a working model of the software product.
- Confirm by testing the completeness of the working model and its alignment with the product design

As discussed in Section 4.1.1, an entity may need to evaluate these conditions if the criteria for establishing technological feasibility under a detail program design aren't met until a working model is completed. In this case, the criteria in this section would need to be met to establish technological feasibility.

Working model

A working model is a functional version of the proposed software product that has been made up to demonstrate how the product will perform. To meet the requirements of establishing technological feasibility, a working model must include all of the following characteristics:

- Operative
- Written in the same language as the product that will be marketed
- Complete with all major functions that were planned for the product
- Ready for initial customer testing

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).

The software industry also uses other definitions of a working model to describe a prototype that has critical parts that are coded or written in a different language than the product that will be sold, or in pseudocode. A model written in pseudocode lacks the key characteristics listed above, specifically the requirement that it be written in the same language as the product that will be marketed, and therefore does not meet the definition of a working model that is used to establish technological feasibility.

ASC 985-20-55-8

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.

ASC 985-20-55-9

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.

4.1.3 Development issues after technological feasibility is established

If a high-risk development issue is discovered after technological feasibility has been established, an entity should charge to research and development all previously capitalized costs incurred to develop the software, as well as any additional costs incurred to return the product back to the point where it meets the technological feasibility criteria once again. These costs are accounted for as a change in accounting estimate using the guidance in ASC 250, since the discovery of a high-risk development issue after the entity concludes that technological feasibility exists qualifies as new information. The entity should charge these costs to research and development expense until technological feasibility is established once more using the criteria in Section 4.1.1 or Section 4.1.2, as appropriate.

ASC 985-20-55-10

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.

4.2 Production costs and post-production costs of computer software

Costs to produce software that are incurred after technological feasibility has been established are capitalized, including coding and testing costs.

Costs of producing of software that will be an integral part of a product or a process should not be capitalized until two conditions are met:

- The entity has established technological feasibility has been established for the project
- The entity has completed all research and development activities for the other components of the project.

ASC 985-20-25-4

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.

An entity should capitalize the direct costs of producing a product master that are incurred after technological feasibility has been established, including costs related to coding and testing the software. The Master Glossary defines a “product master” as a completed version of the entire software product that can be reproduced, including documentation and training materials.

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.

Certain indirect costs that are incurred in developing software after technological feasibility has been established may also be capitalized, such as an allocated amount of the cost of the overhead costs related to the programmers, such as employment taxes, health insurance, workers’ compensation, and paid time off, as well as the cost of the facilities where they work. Entities should not capitalize general and administrative expenses, as they are period costs that should be expensed as incurred.

Grant Thornton insights: Costs for capitalization

While identifying the direct costs of developing a product master may be straightforward, it can be easy to miss some of the indirect costs that should also be capitalized. One such indirect cost is the interest on borrowings that directly fund the production of the software. Interest should be capitalized in line with the capitalized interest guidance in ASC 835-20, *Interest: Capitalization of Interest*.

Employee benefits are another category of often overlooked costs involved in developing software that should be capitalized. The majority of the direct costs of developing software include the salaries of employees coding the software, but indirect costs, such as fringe benefits, taxes, and bonuses are also eligible for capitalization. In order to capture these indirect costs for capitalization, an entity may need to develop an estimation methodology such as hourly overhead rates.

Under the guidance in ASC 985-20, an entity should continue to capitalize the costs incurred in developing software until the product is ready for general release to customers. After the general release, ongoing costs associated with the software, such as maintenance and customer support, are charged to expense at the earlier of when the costs are incurred or the related revenue is recognized.

ASC 985-20-25-3

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.

ASC 985-20-25-5

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.

ASC 985-20-25-6

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.

4.2.1 Costs of customer support and maintenance

When selling software, entities often promise to assist customers in using the software and to keep it functioning and up to date after purchase. These services may be part of the sale of the software to the customer or may be available at the customer's option. These customer support and maintenance activities take place after the software is available for sale and should be charged to expense at the earlier of when the costs are incurred or when the related revenue is recognized.

Customer support includes a range of services provided by the entity that are designed to help customers use the software product. This could include assisting customers with installing the software, or training them on how to use the software, providing telephone or internet support to answer questions and address issues, providing ongoing information on the use of the software through newsletters or other communications, and other similar activities.

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.

The Master Glossary defines “maintenance” as activities that keep the software functioning as it was intended when it was sold. They include correcting errors, keeping a product up to date, and making routine changes. Maintenance activities do not create new features or functionality in the software, but instead ensure that the software continues to function as intended.

On the other hand, a “product enhancement” is an improvement to an existing software product that either extends its life, or significantly improves its marketability. Enhancement activities generally include redesigning some or all of the software product, and therefore go beyond the scope of maintenance activities, which are intended only to maintain the software’s existing functionality. Costs incurred in connection with a product enhancement are capitalized or expensed based on whether technological feasibility of the enhancement has been established, as discussed in Section 4.2.2.

Determining whether any given activity represents maintenance, customer support, or a product enhancement requires judgment, and any determinations must be made based on the specific facts and circumstances of the product and the activity.

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.

Product Enhancements: 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.

ASC 985-20-55-11

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.

ASC 985-20-55-12

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.

4.2.2 Costs of product enhancements

An enhancement is a change to an existing software product that improves its functionality or future marketability. These changes are more extensive than routine maintenance, and often involve redesigning some or all of the software product. Therefore, an entity must establish technological feasibility of the product enhancement before capitalizing any of the associated costs. Establishing technological feasibility for a project is discussed in Section 4.1.

Technological feasibility for a product enhancement can often be established earlier in the development process than might be the case for a new software product. This is because the enhancement modifies an existing product for which technological feasibility has already been established. Sometimes software is "ported," or adapted to allow it to function in a different computing environment than it was originally programmed in, for example, on a different piece of hardware or a different operating system. A new detail program design may not be necessary, and technological feasibility may be established after resolving any high-risk issues in the development process.

ASC 985-20-55-20

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.

ASC 985-20-55-21

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.

ASC 985-20-55-22

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.

Before technological feasibility is established, an entity should recognize the costs of a product enhancement as research and development expense when incurred.

After technological feasibility is established, an entity should capitalize direct costs and allocated indirect costs of completing the enhancement in the same manner as for production of a new software product to be sold or marketed, as discussed in Section 4.2.

If the enhancement replaces the original product to the extent that the original product will no longer be marketed, the remaining capitalized cost of the original product should be added to the cost of the enhancement for purposes of determining the net realizable value and amortization for the enhancement. If the enhancement and the original product will be marketed separately, an entity should allocate the unamortized cost of the original product between the enhancement and the original product.

ASC 985-20-55-18

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.

Costs of original product when enhancement is made

Entity has been selling Version 1 of its software to customers for three years, for which $1,500 of unamortized capitalized costs remain. Entity begins a product enhancement which will add several substantial features to Version 1. Entity incurs $100 of costs before technological feasibility is established, and those costs are expensed as R&D. Entity incurs $900 of direct project costs to complete the enhancement after technological feasibility is established, which costs are capitalized. When the enhancement is complete, Entity releases the enhanced software for sale to its customers as Version 2.

If Entity ceases to sell Version 1 to its customers and only sells Version 2 going forward, the $1,500 of capitalized costs associated with the original software are added to the $900 capitalized for the enhancement. Therefore, $2,400 total costs are used to calculate the amortization and the net realizable value of the capitalized software costs for Version 2.

If Entity continues to sell Version 1 to its customers in addition to selling Version 2, it must allocate the $1,500 of capitalized cost for Version 1 between the original software and the upgrade. Each version will be amortized separately, since they will be sold separately. Using judgment, Entity determines that 33% of the costs of the original software should be allocated to Version 1, and the remaining 66% should be allocated to Version 2. Therefore, the unamortized costs of Version 1 are $500 and the capitalized costs of Version 2 are $1,900, made up of the $1,000 of costs allocated from Version 1 and the $900 from the product enhancement. The capitalized software costs of Version 1 and Version 2 will be accounted for separately.

4.3 Purchased software

If an entity purchases software to be sold or marketed rather than developing the software internally, the accounting for that software depends on whether it has an alternative future use to the entity. The alternative use does not need to be in the same capacity for which the software was purchased. For example, an entity could purchase software for a product it intends to sell to customers but might evaluate that the software's alternative future use is in research and development or the entity's own internal use. This test applies to software that will be sold in the same form in which it is purchased, and to software that will be modified or integrated into another product.

If the purchased software has an alternative future use beyond being sold or marketed to customers, the entity should capitalize the cost of the software when it is acquired.

The software should subsequently be accounted for according to how the entity will use it. For example, if the software is used by the entity for its internal processes in addition to being sold to its customers, the software should be capitalized and amortized based on the internal-use software guidance in ASC 350-40. If the purchased software will be integrated into an existing product or process that is still in the research and development phase, the entity should consider whether the software will potentially offer an alternative future use when determining the accounting for the purchase. If the software is also used internally by the entity in a process outside research and development, for example, the entity would determine if the software should be capitalized using the internal-use software guidance.

ASC 985-20-25-7

Some entities purchase software as an alternative to developing it internally. Purchased computer software may be modified or integrated with another product or process.

ASC 985-20-25-10

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.

Purchased software that does not have an alternative future use is capitalized or expensed based on whether technological feasibility has been established for the entity's software product to be sold or marketed. An entity accounts for this software similar to software developed internally that will be sold or marketed. If the software is purchased after technological feasibility of the project has been established, then the cost of the software is capitalized. If technological feasibility has not been established for the project when the software is purchased, the cost of the purchased software should be expensed as a research and development.

An entity may entity purchase software for which it has no alternative future use in order to integrate that software into an existing product, or one that is in development or already exists should be capitalized only if the end product's research and development activities have been completed at the time of purchase.

ASC 985-20-25-8

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.

ASC 985-20-25-9

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.

ASC 985-20-55-13

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.

If an entity purchases software to sell or market before its technological feasibility has been established, the cost of the software may be capitalized only to the extent that it has an alternative future use. If the software does not have an alternative future use and technological feasibility has not been established, the software cost should be charged to research and development. The example in ASC 985-20-55-14 below illustrates the accounting for software purchased before technological feasibility is established.

ASC 985-20-55-14

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.

4.4 Funded software-development arrangements

A funded software-development arrangement is an agreement in which a third party pays an entity for some or all of the cost of developing software. Accounting for costs under this type of an arrangement depends on whether technological feasibility has been established for the software being developed. If technological feasibility has not been established, the cost of the software is accounted for in accordance with the research and development guidance discussed in Section 3. If technological feasibility has been established before the arrangement takes effect, the entity should capitalize or expense costs in accordance with the guidance on software to be sold, leased, or marketed in this Section.

If the party funding the agreement is a collaborator or partner rather than a customer, the entity developing the software should offset any income received under the funded agreement against any capitalized development costs. If capitalized costs are reduced to zero, income should be deferred and offset against future capitalized development costs. Income that is deferred at the end of the capitalization period should be recognized in income as long as there are no capitalized development costs to offset.

If the counterparty funding the agreement is a customer rather than a collaborator or partner, the entity should recognize revenue derived from the arrangement under the guidance in ASC 606.

ASC 985-20-25-12

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

4.5 Costs of producing inventory

An entity may incur costs other than development costs for producing software that will be sold, leased, or marketed. These costs may include the cost of creating copies of the software for distribution, as well as documenting, and physically packaging the software and any related training materials. The costs of creating a physical product to be sold to the customer is treated separately from development costs. These costs are therefore capitalized as a part of inventory on a unit-specific basis. Just like any inventory item, when the entity sells the product the cost of the inventory is recognized as cost of goods sold.

ASC 985-330-25-1

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.

ASC 985-330-40-1

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 charged to cost of sales when revenue from the sale of those units is recognized.

4.6 Amortization of capitalized amounts

The capitalized costs of developing software that will be sold, leased, or marketed should be amortized separately for each software product. An entity should begin amortizing the capitalized costs of the software when the product first becomes available for general release to customers.

An entity should measure the amortization of a software product using a net realizable value test, which is based on the proportion of current gross revenue to the total of current and estimated future gross

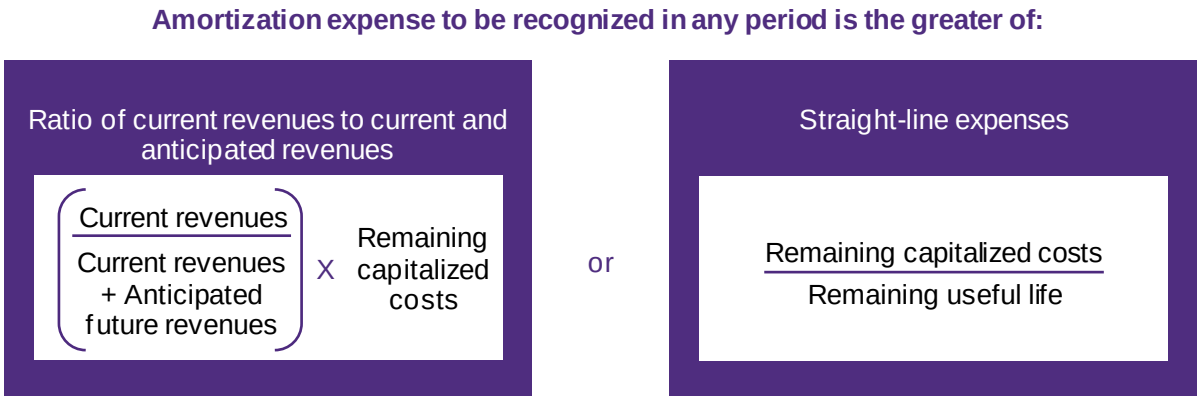
revenue for the product. However, the entity is also required to measure the amount of amortization that would be recognized on a straight-line basis for the software over the product's remaining useful life, and that measurement should be recognized as the minimum amount of amortization if it is greater than the amortization calculated using the net realizable value test. The calculation of straight-line minimum amortization is required to establish a floor on the amortization of costs of software to be sold because there is an inherent uncertainty involved in estimating future revenues.

When estimating the future revenue for a product, the entity should use the most recent information available. As a result, the estimate of the remaining future revenue or the economic life of a product may change as new information becomes available during the amortization period. Those changes in estimate are treated on a prospective basis, and therefore are recognized in calculations of amortization going forward only.

An entity is required to determine the amount to be recognized on a straight-line basis for the product over its remaining economic life by dividing the unamortized cost of the product over its remaining economic life, including the current year.

The following figure illustrates the calculation of amortization in any given period for capitalized costs of software that will be sold, marketed or leased.

Figure 4.2: Amortization of capitalized software costs intended to be sold, marketed, or leased



ASC 985-20-35-1

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.

ASC 985-20-35-2

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.

ASC 985-20-55-15

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.

ASC 985-20-55-16

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.

Amortization of capitalized software costs

Company A develops a software product that it will sell to customers. Total capitalized costs associated with the software are $1,000,000. Software is available for general release to customers on 1/1/X1 and has an estimated useful life of five years. At that date, Company A estimates that total revenue from the software will be $23,000,000.

In 20X1, Company A's revenue associated with the software is $3,000,000. Company A calculates the amortization of the capitalized software costs by determining the greater of the straight-line amortization or the proportion of current-year revenue to total current-year and remaining expected revenues. As the calculated straight-line amortization of $200,000 is greater than the revenue calculation, Company A recognizes amortization totaling $200,000 in Year 1.

Straight-line amortization (A)	Current year revenue / Total revenue (B)	Amortization: Greater of A or B	Net book value of costs
$1,000,000 / 5 years = $200,000	($3,000,000 / 23,000,000) x $1,000,000 = $130,435	$200,000	$800,000

In 20X2, Company A's revenue associated with the software totals $8,000,000. Company A's estimate that total remaining revenue over the remaining four-year life will be $20,000,000 is unchanged. Company A calculates the amortization of the capitalized software costs by determining the greater of the straight-line amortization or the proportion of current-year revenue to total current-year and remaining expected revenue. As the calculated amortization based on proportional revenue of $320,000 is greater than the straight-line calculation of $200,000, Company A recognizes amortization totaling $320,000 in Year 2.

Straight-line amortization (A)	Current year revenue / Total revenue (B)	Amortization: Greater of A or B	Net book value of costs
$800,000 / 4 years = $200,000	($8,000,000 / 20,000,000) x $800,000 = $320,000	$320,000	$480,000

In 20X3, Company A's revenue associated with software sales totals $9,000,000. In addition, based on greater demand than initially expected for the product, the company adjusts its estimate of remaining revenue from $12,000,000 to $16,000,000. Company A treats this adjustment as a change in accounting estimate and applies it to the amortization calculation for the remaining software costs prospectively, in accordance with the guidance in ASC 250. Company A calculates amortization of the capitalized software costs by determining the greater of the straight-line amortization or the proportion of current-year revenue to its new estimate of total current year and remaining expected revenues. As the calculated amortization based on proportional revenue of $270,000 is greater than the straight-line calculation of $160,000 Company A recognizes amortization totaling $270,000 in Year 3.

Straight-line amortization (A)	Current year revenue / Total revenue (B)	Amortization: Greater of A or B	Net book value of costs
$480,000 / 3 years = $160,000	($9,000,000 / 16,000,000) x $480,000 = $270,000	$270,000	$210,000

4.6.1 Amortization of product enhancements

All capitalized costs of a product enhancement should be amortized over the estimated useful life of the enhancement, regardless of whether those costs are directly related to the enhancement or are costs carried over, or allocated to the enhancement, from the original product (see Section 4.2.2). The estimated useful life of the enhancement is determined based only on the enhancement itself, not on the remaining life of the original product nor is it the remaining life of the original product for costs of the original product included in the enhancement and the estimated life of the enhancement for all other costs.

ASC 985-20-55-19

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.

4.7 Impairment of capitalized amounts

Each reporting period, the remaining unamortized costs of the software should be evaluated for impairment by comparing the total unamortized cost to the net realizable value of the product.

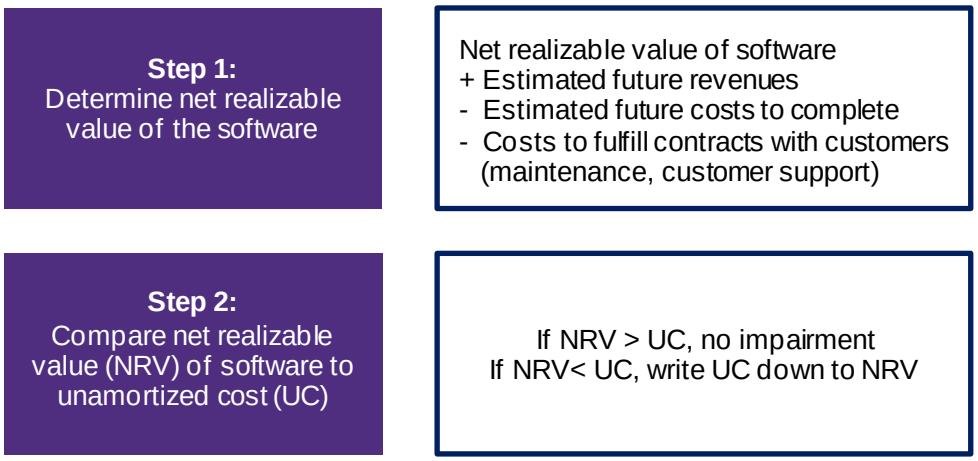
The net realizable value of the software product is made up of all of the following amounts

- Estimated future gross revenue of the product
- Estimated future costs of completing and disposing of the product
- Costs of performing maintenance and customer support required to fulfill the contracts with customers

If the unamortized cost of the software exceeds its net realizable value, the excess cost is written off.

The following figure illustrates the evaluation of impairment for capitalized costs of software to be sold or marketed.

Figure 4.3: Evaluating impairment for capitalized costs of software to be sold or marketed



If an entity determines that a software product is impaired, the net realizable value becomes the new cost of the asset. As with impairments of all long-lived assets, an entity may not write the value of the software costs back up in subsequent periods if there are changes in net realizable value or for any other reason.

ASC 985-20-35-4

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.

Impairment test for capitalized costs of software to be sold

Entity has two software products that it sells to its customers, Software A and Software B. At year end, Entity tests the unamortized capitalized costs of each of the products for impairment by comparing the net realizable value of the product to its unamortized cost. Entity's estimates and unamortized cost balances for Software A and Software B are shown in the table below.

	Software A	Software B
Estimated future gross revenue	$500	$100
Estimated future cost to complete	$100	$ 0
Estimated future costs of maintenance and customer service	$100	$ 25
Unamortized cost	$200	$100

The net realizable value of Software A is $300 ($500 - $100 - $100), which is in excess of its unamortized costs of $200. Therefore, no impairment is recognized for Software A.

The net realizable value of Software B is $75 ($100 - $0 - $25), which is less than its remaining unamortized cost of $100. Entity recognizes the excess of the net realizable value over the unamortized cost, as an impairment of $25. After impairment the new unamortized cost of Software B is $75, and the costs cannot be written back up in later periods, even if the net realizable value of Software B increases due to changes in estimated costs or future gross revenue.

4.8 Presentation and disclosure

Capitalized costs of software are an amortizable intangible asset. If these costs have a life span of one year or longer, the entity should present them as other assets on the balance sheet.

Amortization of the costs of capitalized software should be presented in the income statement as cost of sales or in a similar expense category, since it relates to the costs of a product that is sold or marketed.

An entity must also apply the presentation and disclosure requirements for the capitalized costs of internal-use software in ASC 350-40 to the capitalized costs of software that is sold or marketed. Those disclosure requirements refer to other topics in the Codification, as discussed in Section 2.6.

ASC 985-20-45-1

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.

ASC 985-20-45-2

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.

ASC 985-20-45-3

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.

An entity must disclose unamortized capitalized software costs for each balance sheet presented, as well as the total amounts for amortization expense and amounts written down to net realizable value in each income statement period. In addition, research and development costs charged to expense for software that has not yet reached technological feasibility should be disclosed.

ASC 985-20-50-1

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.

ASC 985-20-50-1

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.

In addition to the disclosures required by ASC 985-20, an entity must disclose information about the amortization of capitalized software costs, which is required under the guidance for risks and uncertainties in ASC 275. The amortization of capitalized software costs is based on an estimate of future revenue and the useful life of the software, both of which are inherently uncertain. The

implementation guidance in ASC 985-20-55 presents an example of a disclosure that would be required under ASC 275, which stresses the importance of disclosing the fact that it is reasonably possible the carrying amount of the software costs might be reduced in the near term due to these uncertain estimates.

Example 1: Disclosure of Risks and Uncertainties Related to Capitalized Software Costs

ASC 985-20-55-23

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.

ASC 985-20-55-24

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.

ASC 985-20-55-25

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.

ASC 985-20-55-26

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.

ASC 985-20-55-27

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.

ASC 985-20-55-28

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.

ASC 985-20-55-29

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.

5. Costs to implement a cloud computing arrangement

Cloud computing arrangements are becoming more common and are replacing arrangements where an entity licenses software and runs it on its own premise. Cloud computing refers to an arrangement where a third party provides hosting services. Accounting for costs incurred to implement a cloud computing arrangement depends on whether the arrangement includes a software license. When an entity (customer) can take possession of the hosted software, the arrangement includes a software license that is accounted for under ASC 350-40.

In circumstances where the entity cannot take possession of the hosted software, the entity accounts for the hosting arrangement as a service contract, by recognizing the expense of the contract in each period without capitalizing the hosted software. However, an entity often also incurs costs when implementing a cloud computing arrangement. For those costs, the entity should apply the guidance in ASC 350-40 to account for costs that are incurred to implement a cloud computing arrangement that is a service contract. Those costs might relate to customizing, configuring, or integrating the hosted software with existing software, or the cost of converting data or training employees to use the new software. Capitalized implementation costs are expensed over the term of the hosting arrangement and the entity is required to evaluate those costs for impairment as if the costs were long-lived assets, as discussed in this section.

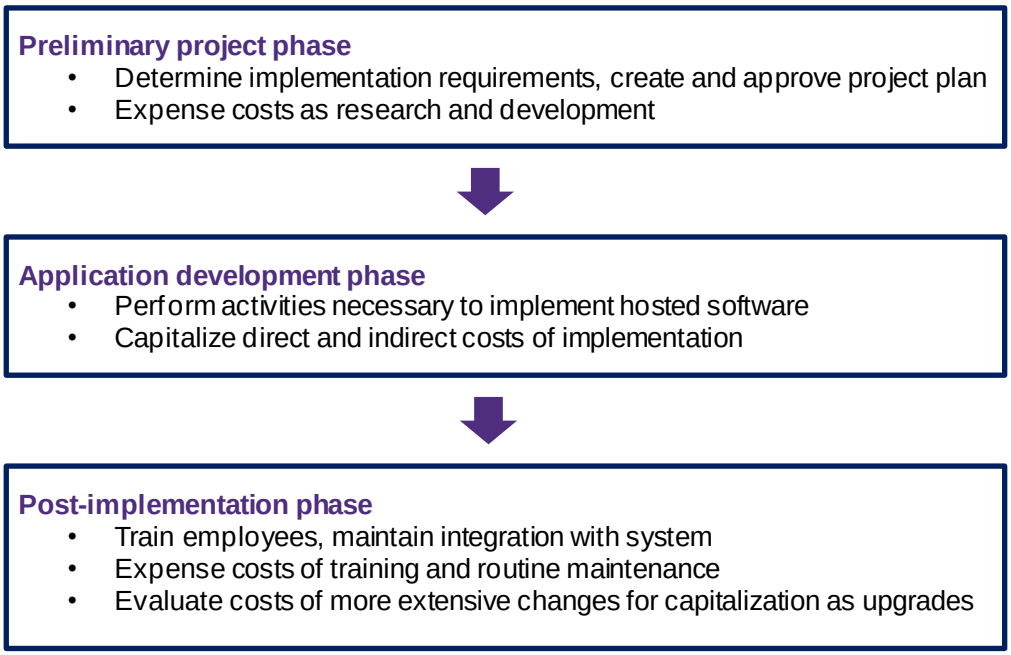
5.1 Costs to implement a cloud computing arrangement that is a service contract

When accounting for costs incurred as a customer to implement a cloud computing arrangement that is a service contract, an entity should apply the internal-use software guidance in ASC 350-40. That guidance considers both the nature of the costs and the phase of development in which the implementation costs are incurred to determine whether the costs should be capitalized or expensed.

During the preliminary project phase, all implementation costs are expensed as research and development. Once the preliminary project phase is complete and the entity has committed to moving forward with the implementation and it is probable that it will function as intended, the implementation enters the application development phase. During this phase, all direct and certain indirect costs of the implementation are capitalized, while other costs are expensed, based on the nature of the costs. When the implementation is complete and the hosted software is ready for use, the project reaches the post-implementation phase, where costs such as training or routine maintenance are expensed as incurred.

The following figure illustrates the phases and the accounting for associated costs, which are discussed in detail in Section 2.

Figure 5.1: Accounting for implementation costs in a hosted software arrangement



ASC 350-40-25-18

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.

Evaluating cloud computing implementation costs

Customer A enters into a cloud computing arrangement to access Company B's software, which it will use internally in its sales process as a customer relationship management tool. The contract contains an initial two years of access to the hosted software at $25,000 per month, with an option to renew for two additional annual periods.

Customer A does not have the contractual right to take possession of the software at any time during the hosting period without incurring a significant penalty. In addition, Customer A does not have the hardware capabilities to feasibly run the software and would not have the ability to contract with a third party unrelated to the vendor to host the software. As a result, Customer A deems that it has not obtained a software license from Company B, but instead has entered into a service agreement to use Company B's software. Customer A therefore applies the guidance for a cloud computing arrangement that is a service agreement in ASC 350-40 to the implementation costs incurred in relation to the contract.

Customer A incurs $100,000 in the initial phase of the project to obtain information about the technology and costs of implementing the system, and to create and select an implementation plan. These costs are considered preliminary project stage expenses and are expensed as incurred.

Company B performs implementation services to integrate the software in its cloud computing arrangement into Customer A's existing systems, and to train Customer A's employees in the use of the new software. Customer A pays $500,000 upfront to Company B for these services, of which $400,000 is directly related to implementing the hosted software and is therefore capitalized in accordance with ASC 350-40-25-2. The remaining $100,000 is related to the manual cost of converting data from Customer A's existing system into the hosted software and is therefore expensed as incurred.

Customer A also incurs $100,000 of payroll and benefits for its IT personnel working with Company B to perform implementation activities. Customer A determines that these costs relate directly to implementing the hosted software and therefore capitalizes them in accordance with ASC 350-40.

Once the project is ready for its intended use, Customer A incurs an additional $100,000 to train employees on using the new hosted software.

In total, Customer A capitalizes $500,000 of the implementation costs and expenses $200,000 for preliminary stage and training costs.

5.1.1 Multiple element arrangements in a hosted arrangement

A contract for a hosted arrangement may contain several services in addition to the right to use the hosted software, such as training, maintenance and support. To the extent that any of those services is recognized separately, an entity should allocate the price of the arrangement among the elements of the arrangement based on their relative stand-alone prices. For further discussion, see Section 2.2.2.

5.2 Amortization of implementation costs of cloud computing arrangement

Under ASC 350-40, an entity starts to amortize the capitalized implementation costs when the software in the hosting arrangement is ready for its intended use, which is generally after all substantial testing has been completed. An entity should consider the unit of account in determining when to start amortizing capitalized implementation costs related to a hosting arrangement that is a service contract. ASC 350-40 requires an entity to consider implementation of each module or component of the hosting arrangement separately, unless its functionality depends on the implementation of other modules. An entity should begin amortizing implementation costs related to an individual module when all of the following criteria are met:

- The module is ready for its intended use.
- Testing on the module has been completed.
- Using the module does not depend on the completion of any other modules.

If the functionality of an individual module or component depends entirely on the completion of another module, amortization should begin only when both modules are completed and ready for use.

ASC 350-40-35-17

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.

Capitalized implementation costs stemming from a cloud computing arrangement should be amortized over the term of the associated hosting arrangement, as defined. Amortization should be recognized on a straight-line basis, unless another systematic and rational basis better depicts the use of the software. When determining the amortization term, an entity should include the noncancelable term of the hosting arrangement, plus periods covered by any option to

- Extend the arrangement, if the entity is reasonably certain to exercise that option
- Terminate the arrangement, if the entity is reasonably certain not to exercise that option
- Extend (or not terminate) the arrangement, if the vendor controls whether or not the option will be exercised

An entity should determine the amortization period for hosted software the same way it determines the amortization period when it purchases and uses a software license. An entity would consider the length of time over which it plans to use the hosted software rather than how many transactions are expected to be processed using the hosted software in determining the amortization period.

Grant Thornton insights: Determining the amortization period

The process of estimating the amortization period for capitalized implementation costs stemming from a cloud computing arrangement requires entities to exercise judgment when evaluating the particular facts and circumstances in each case. In determine the amortization period an entity might consider the expected life of the underlying software. Because of the rapid pace of change in the software and technology industry, the underlying software in a hosting arrangement could have an expected life that extends through only the initial term, or for less than all of the renewal options that are included in the arrangement. Sometimes, the entity may be more likely to enter into an arrangement for a new hosted software product rather than renewing an existing arrangement for an older product.

The estimated term of the arrangement should be periodically reassessed, and any change in the term should be accounted for prospectively as a change in accounting estimate under ASC 250.

ASC 350-40 includes the following factors to consider when assessing the estimated term of a hosting arrangement.

ASC 350-40-35-16

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.

5.3 Impairment

Capitalized implementation costs are reviewed for impairment when events or changes in circumstances indicate that the carrying amount may not be recoverable under the guidance in ASC 360-10-35. The costs of each module or component of a hosting arrangement that is a service contract should be evaluated separately to determine whether the module or component is no longer used, as an individual module or component may no longer be used even if the underlying core hosted software is still viable. If a module or component is discontinued, the related capitalized costs should be accounted for as "abandoned" under the guidance in ASC 360-10-35. Capitalized implementation costs should be evaluated for impairment as if the capitalized costs were long-lived assets. The following examples of events and changes in circumstances might indicate that the carrying amount of the related implementation costs cannot be recovered and that the asset is therefore impaired:

- The entity no longer expects that the hosting arrangement will provide future benefit
- The extent or manner in which the entity uses the software underlying the hosting arrangement has changed significantly or is expected to change significantly
- The entity has significantly changed the hosting arrangement, or expects to do so in the future

The termination of a cloud computing arrangement, whether in full or only for a particular component, cuts off the customer's access to any benefit that could be derived from using that component of hosted software. As a result, the entity should write off the related asset for capitalized implementation costs for each module or component that is terminated and no longer used.

ASC 350-40-35-11

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.

ASC 350-40-35-12

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.

Abandoned costs of implementing a cloud computing arrangement

Entity A has capitalized costs from implementing a cloud computing arrangement for hosted software containing modules that its employees use to report their time and their expenses. After a while, Entity A purchases new software for tracking expenses that allows employees to better break down and itemize expenses.

Due to this purchase, Entity A assesses that it will no longer use the expenses module in the cloud computing arrangement. Therefore, Entity A accounts for the implementation costs related to the expenses module as abandoned, and writes off the related implementation costs in accordance with the guidance in ASC 360-35. As Entity A expects to continue using the employee time recording module, it continues to recognize and amortize implementation costs associated with this module.

5.4 Presentation and disclosure for costs of a cloud computing arrangement

Although the implementation costs of a cloud computing arrangement are recognized as a separate asset, they are presented with balances and activity of the related cloud computing arrangement, since these costs exist only to enhance that arrangement for the customer. Therefore, entities should present capitalized implementation costs in the same line item of the balance sheet where any prepaid costs of the cloud computing arrangement are presented.

The amortization of capitalized costs for a cloud computing arrangement should be presented in the same line item of the income statement as expenses related to the arrangement. An entity should not present expenses related to the amortization of implementation costs in the same line item as depreciation for property, plant, and equipment or any other amortization, if these items are presented separately from other expenses within the financial statements.

Cash flows related to the capitalized implementation costs should be presented in the same line item of the statement of cash flows as payments for the cloud computing arrangement to which they relate.

An entity (customer) is required to disclose the nature of a hosting arrangement that is a service contract, and to provide the disclosures required in ASC 360 as though the capitalized implementation costs were a separate major class of depreciable assets, including the following information:

- Amortization expense for the period
- Balance of major classes of depreciable assets
- Accumulated amortization at the balance-sheet date
- A general description of the method/s used in computing amortization for major classes of depreciable or amortizable assets

6. Internal-use software subsequently marketed

At times, an entity may subsequently decide to sell software that was originally developed for internal use. Under the guidance in ASC 350-40, an entity that subsequently decides to market software that was designed for internal use is required to write down the carrying amount of the capitalized software as it receives licensing fees for the technology.

An entity should reduce the carrying amount of capitalized internal-use software by the proceeds, net of any direct incremental costs of marketing like commissions, reproductions costs, warranty expenses, and installation costs of licensing that software. Before recognizing profit from licensing, the carrying amount of that capitalized internal-use software should be written down to zero.

After the carrying amount is reduced to zero, additional proceeds are recognized as either revenue or as other income. If licensing software is an ordinary activity of the entity and the contract is with a customer, the entity applies ASC 606 and recognizes revenue. If licensing software is not an ordinary activity, the entity applies the guidance in ASC 610-20, *Other Income: Gains and Losses from the Derecognition of Nonfinancial Assets*, and recognizes a gain.

ASC 350-40-35-7

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.

ASC 350-40-35-8

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.

The following example demonstrates the application of this guidance.

Reducing the carrying amount of software prior to recognizing revenue

Entity E, a retail entity, develops software for its own point-of-sale system and capitalizes $1 million in development costs. The software is completed and placed into service on 1/1/20X5 and is amortized on a straight-line basis, or $200,000 per year, over five years. On 1/1/20X7, two years after completion of the project, the carrying amount of the development costs is $600,000. At that point, the entity decides to market the point-of-sale software to other companies.

During 20X7, the first year of marketing the software license externally, Entity E generates $200,000 in net proceeds, which reduces the carrying amount of the capitalized software. Additionally, the entity recognizes amortization of $200,000 bringing the carrying amount of the capitalized software to $200,000 as of the end of 20X7.

In Q1 20X8, the entity generates $250,000 in net proceeds from licensing the point-of-sale software. Because the carrying amount of the capitalized software is only $200,000, the entity reduces the carrying amount of the software to zero and recognizes $50,000 as a gain.

6.1 Decision to market made before software is complete

An entity that decides to market internal-use software externally while the software is being developed should stop applying the guidance in ASC 350-40 and begin accounting for costs to develop the software using the guidance on software that is sold, leased, or marketed in ASC 985-20.

ASC 350-40-35-9

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.

Any costs previously capitalized under ASC 350-40 should also be evaluated to determine if they exceed net realizable value under the guidance in ASC 985-20. In other words, an entity should compare the unamortized costs of the computer software to the product's net realizable value and then write off the amount by which the unamortized costs exceed net realizable value.

ASC 985-20-35-4

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 example illustrates the comparison of the net realizable value of a software product to be sold, to the carrying amount of the internal-use software that was capitalized when the entity decided to market the software.

Carrying value exceeds net realizable value

Entity B develops software to accumulate and analyze data collected through customer support calls. The entity began developing the software with the intention to use it only internally, but, as the development costs continue to mount, it decides to market the software to other companies to recoup some of these costs.

Entity B estimates that licensing the software will result in future sales of $500,000, maintenance costs of $50,000 and customer support costs of $100,000. As of the date it decided to market the software externally, the entity had capitalized $750,000 in application development costs and estimates that it will incur an additional $250,000 in costs before the software is ready for its intended use. The net realizable value is therefore calculated as $100,000 ($500,000 in future sales – $50,000 in maintenance costs – $100,000 in customer support costs – $250,000 remaining costs of development).

The entity compares the unamortized capitalized costs of $750,000 to the net realizable value of $100,000 and writes the asset down to $100,000 and expenses the remaining $650,000. The resulting asset and any additional costs to complete the project are subsequently accounted for under the guidance in ASC 985-20, as discussed in Section 4.

The first time an entity decides to subsequently market internal-use software, it sets a precedent for doing so with future projects. Under the guidance in ASC 350-40-35-10, a pattern of marketing internal-use software creates a rebuttable presumption that the entity intends to market any future software it develops, requiring future software projects to be accounted for under ASC 985-20. The entity should determine whether one or a series of past decisions creates such a pattern. Once a pattern is established, it is presumed that the entity will externally market any future software that it develops, unless the entity can overcome that presumption. In practice, overcoming a rebuttable presumption can be a high hurdle, requiring strong evidence and documentation. See Section 1.1.1 for further discussion of the rebuttable presumption.

ASC 350-40-35-10

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.

Contacts

Lynne Triplett
Partner-in-Charge
Accounting Principles Group
T +1 312 602 8060
E Lynne.Triplett@us.gt.com

Sandy Heuer
Partner
Accounting Principles Group
T +1 612 677 5122
Sandy.Heuer@us.gt.com