

AN INDUSTRY SUPPLEMENT TO
BDO BLUEPRINT

Revenue Recognition in the Software Industry

November 2026

BDO

TABLE OF CONTENTS

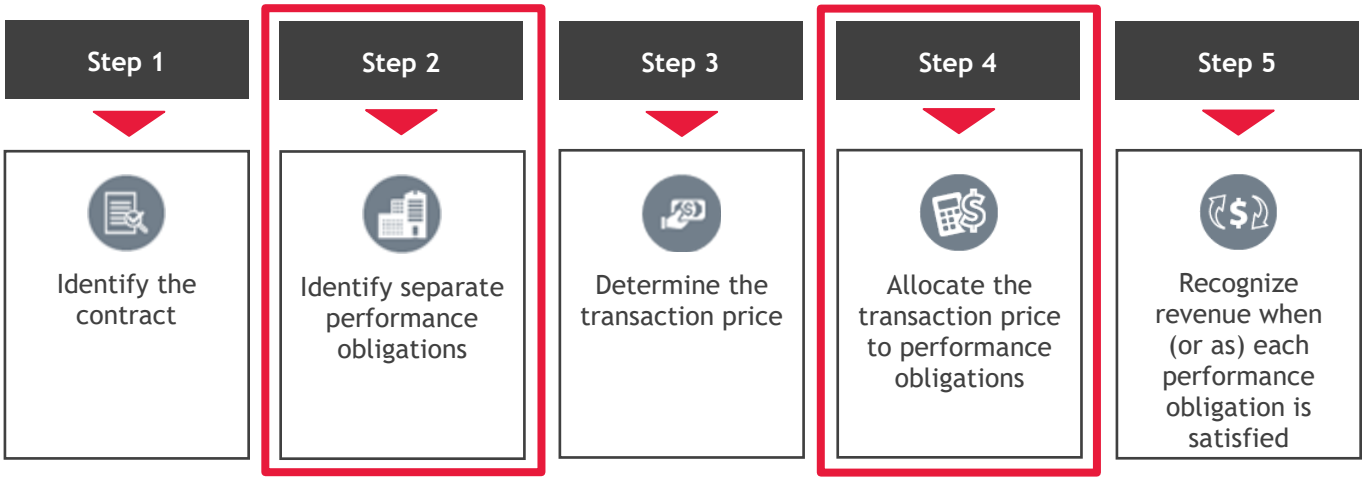
Revenue Recognition in the Software Industry 3
Introduction 3
Identifying Separate Performance Obligations 4
Overview: Identifying Separate Performance Obligations 4
Identifying Promised Goods or Services 7
Are the Promised Goods or Services Distinct? 13
Series of Distinct Goods or Services 27
Material Right 35
Principal Versus Agent Considerations 42
Interaction of Step 2 With Identifying a Contract and Determining the Contract Term in Step 1 47
Determining Standalone Selling Price and Allocating Transaction Price 51
Overview: Determining Standalone Selling Price and Allocating Transaction Price 51
Allocation Objective 52
General Allocation Model 52
Allocation of Variable Consideration 60
Appendix A: BDO Blueprints 70
Appendix B: Summary of Significant Changes 71
Contacts 72

Revenue Recognition in the Software Industry

INTRODUCTION

A software entity applies ASC 606, *Revenue from Contracts with Customers*, to recognize revenue from a contract with a customer. The revenue recognition guidance is industry agnostic, and a software entity makes several judgments and estimates to determine how much revenue to recognize, and when. Software entities face unique challenges in applying certain aspects of the guidance because of the judgments inherent in applying the guidance, the unique nature of software products, and the continuous evolution of software products and related pricing practices.

The revenue recognition guidance is applied in the five steps illustrated below (the five-step revenue recognition model). In applying ASC 606, it is crucial for a software entity to appropriately identify the performance obligations in Step 2 to correctly recognize revenue from a contract with a customer because each performance obligation is a separate unit of account for determining when and how much revenue to recognize. In Steps 3 through 5 of the five-step revenue recognition model, the transaction price in a contract is allocated to each performance obligation in the contract, and the allocated transaction price is recognized in revenue when or as the performance obligation it relates to is satisfied.



About This Publication

This publication discusses certain revenue recognition considerations specific to the software industry with a focus on identifying the performance obligations in a contract with a customer (or Step 2) and determining standalone selling price to allocate the transaction price (Step 4). Throughout this publication we make references to guidance included in Steps 1, 3 and 5 to give better context or explain the effects of the conclusions reached in Steps 2 and 4. We encourage you to read the interpretations and examples in this publication with BDO's Blueprint, **Revenue Recognition Under ASC 606**, which includes a detailed discussion of the five-step revenue recognition model and certain other key concepts included in ASC 606.

Revenue recognition for contracts with customers varies based on the specific facts and circumstances of each contract and, therefore, may differ from the examples and insights in this publication or in BDO's Blueprint, **Revenue Recognition Under ASC 606**.

Identifying Separate Performance Obligations

OVERVIEW: IDENTIFYING SEPARATE PERFORMANCE OBLIGATIONS

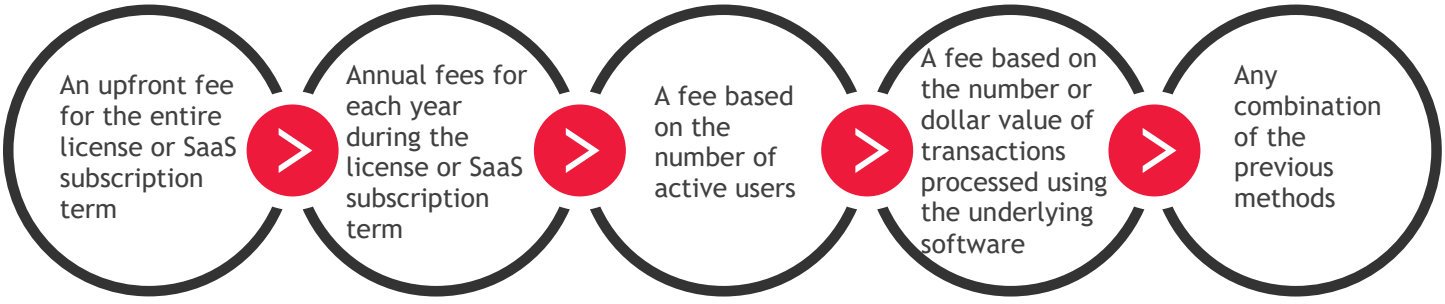
A software entity may generate revenue by selling software licenses or making a software's functionality available to customers as software-as-a-service (SaaS). Additionally, a software entity often sells hardware or services such as post-contract support (PCS) for software licenses and professional services such as implementation, training, or customization.

A software license establishes a customer's rights over the intellectual property (IP) of the software entity. A software license can be for a specified term (referred to as an on-premise term license or simply a term license) or in perpetuity. In recent years, the software industry has generally shifted away from an on-premise license model towards SaaS offerings and new product offerings such as a "hybrid" product that includes the elements of both an on-premise software license and SaaS and allows a customer the flexibility to choose between the on-premise software license and SaaS. Additionally, as the software industry has evolved, pricing models have also evolved, with more emphasis on usage-based pricing.

To illustrate the multiple ways in which a software entity can offer the same software functionality to its customer, consider a software entity that has developed a proprietary software to calculate sales tax. The software entity may monetize its product in any of the following ways:

- By granting an on-premise software license that allows a customer to download and install the tax software on the customer's systems to derive the benefits of tax computation functionality of the software
- By providing a cloud-based tax computation service or SaaS that allows the customer to derive the benefits of the tax computation functionality by accessing the software hosted on a cloud-based platform through the internet
- By providing a hybrid offering that includes both SaaS that the customer can access through the internet and an offline version of the software application that the customer can access on its device without the need for an internet connection

In addition, a software entity might price a license or SaaS in various ways, including:



Given the assortment of product offerings and pricing models in the software industry, identifying performance obligations for revenue recognition is often complex. Nuanced differences in the facts and circumstances of each arrangement could lead to different conclusions for arrangements that may appear similar on the surface. A software entity must consider the requirements in Step 2 of the five-step revenue recognition model and the license-specific guidance (hereinafter referred to as the "licensing guidance"), if applicable, to appropriately identify its separate performance obligations in a contract with a customer.

BDO INSIGHT 1: GUIDANCE THAT APPLIES TO SOFTWARE LICENSES AND SAAS IS DIFFERENT

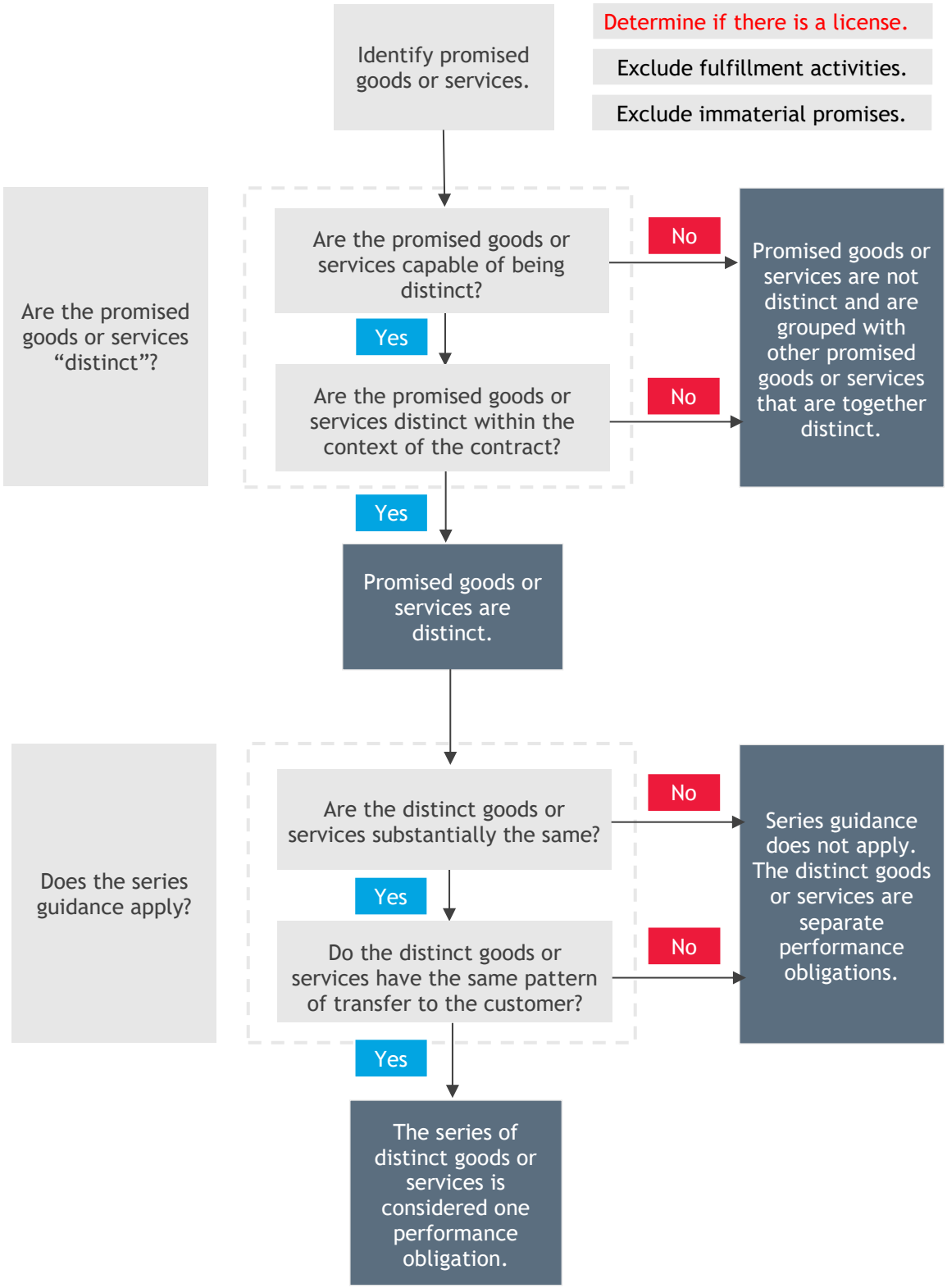
Given the unique nature of licenses, ASC 606 includes specific guidance that applies only to licenses of IP. Revenue from licenses of IP is recognized in accordance with the five-step revenue recognition model and the licensing guidance.

The licensing guidance does not apply to SaaS arrangements that do not transfer a license to the customer. Instead, revenue from a SaaS arrangement is recognized in the same manner as revenue from any other services under the five-step revenue recognition model. In particular, the exception from estimating certain sales- or usage-based fees described in ASC 606-10-55-65 applies only to licenses and not to SaaS arrangements.

To identify its separate performance obligations in a contract with a customer, a software entity evaluates all its promises to the customer at contract inception to determine:

- Which promised goods and services are distinct and hence separate performance obligations
- Whether it has granted a software license

The following diagram provides an overview of this analysis:


Identify promised goods or services.
Determine if there is a license.
Exclude fulfillment activities.
Exclude immaterial promises.
Are the promised goods or services "distinct"?
Are the promised goods or services capable of being distinct?
No
Promised goods or services are not distinct and are grouped with other promised goods or services that are together distinct.
Yes
Are the promised goods or services distinct within the context of the contract?
No
Yes
Promised goods or services are distinct.
Does the series guidance apply?
Are the distinct goods or services substantially the same?
No
Series guidance does not apply. The distinct goods or services are separate performance obligations.
Yes
Do the distinct goods or services have the same pattern of transfer to the customer?
No
Yes
The series of distinct goods or services is considered one performance obligation.

BDO INSIGHT 2: REASSESSING PERFORMANCE OBLIGATIONS

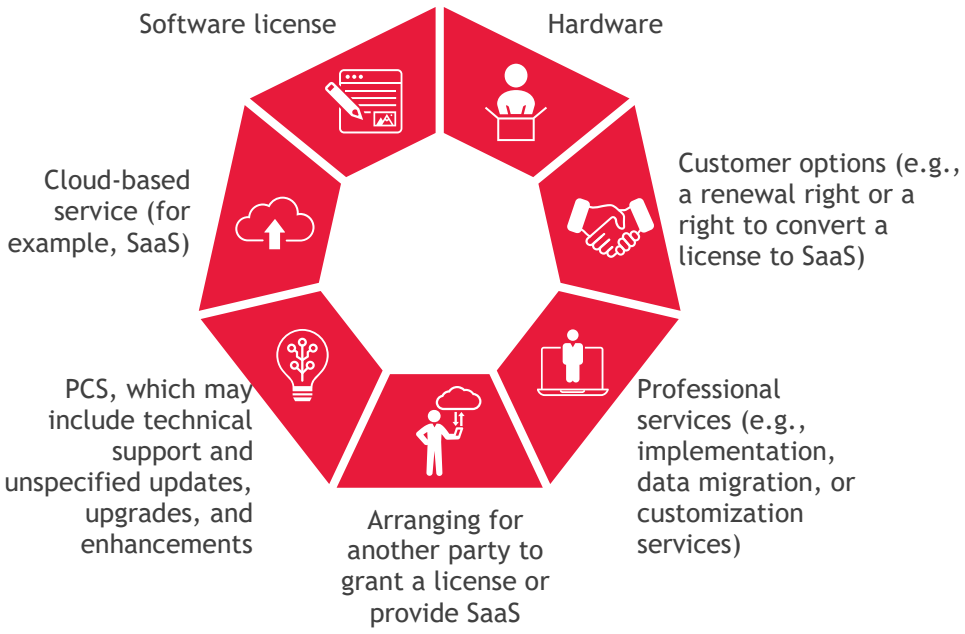
The performance obligations identified in a contract are not reassessed unless the contract is modified, and the modification is not accounted for separately from the existing contract.

Additionally, a software entity must also consider:

- Whether a customer option to purchase additional goods or services represents a material right
- Whether the software entity is a principal or an agent in a software arrangement that involves a third party
- The effects of determining the existence and duration of a contract in Step 1 on identifying the performance obligations in the contract in Step 2

IDENTIFYING PROMISED GOODS OR SERVICES

To identify the performance obligations, a software entity first identifies, at contract inception, all the promised goods or services in a contract with a customer. Promised goods or services may be explicitly identified in the contract or implied by the software entity's customary business practices. Examples of common promised goods or services include:



Determining Whether a Software Entity Has Granted a License

A software entity must determine whether it has granted a software license or is providing SaaS to its customer. A software license establishes a customer's rights over the IP of a software entity. A software entity recognizes revenue from licenses of IP in accordance with the licensing guidance, which is incremental to the five-step revenue recognition model. The licensing guidance does not apply to SaaS, and a software entity recognizes revenue from SaaS arrangements in the same manner as revenue from any other services under the five-step revenue recognition model.

To determine whether a contract with a customer includes a software license for accounting purposes, a software entity needs to determine whether both of the following criteria are met for a hosting arrangement[1]:

- The customer has the contractual right to take possession of the software at any time without significant penalty.
- It is feasible for the customer to run the software on its own hardware or to contract with an unrelated party to host the software.

[1] ASC 985-20 defines a hosting arrangement as *"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 contract includes a software license for accounting purposes if both criteria are met. Conversely, if either one of the criteria is not met then the contract does not include a software license for accounting purposes (regardless of the contractual terms) and the software entity is providing SaaS to the customer.

BDO INSIGHT 3: DETERMINING WHETHER A SOFTWARE ENTITY HAS GRANTED A LICENSE REQUIRES JUDGMENT

Determining whether a software entity has granted a software license to its customer requires judgment. To reach an appropriate accounting conclusion, it is critical to thoroughly understand the solution being provided and what access rights the customer has. In practice, an arrangement that meets the definition of a license is often referred to as an “on premise” or "on prem" solution which reflects the fact that the software is or could be delivered to the customer and the customer could install and run it on its own hardware.

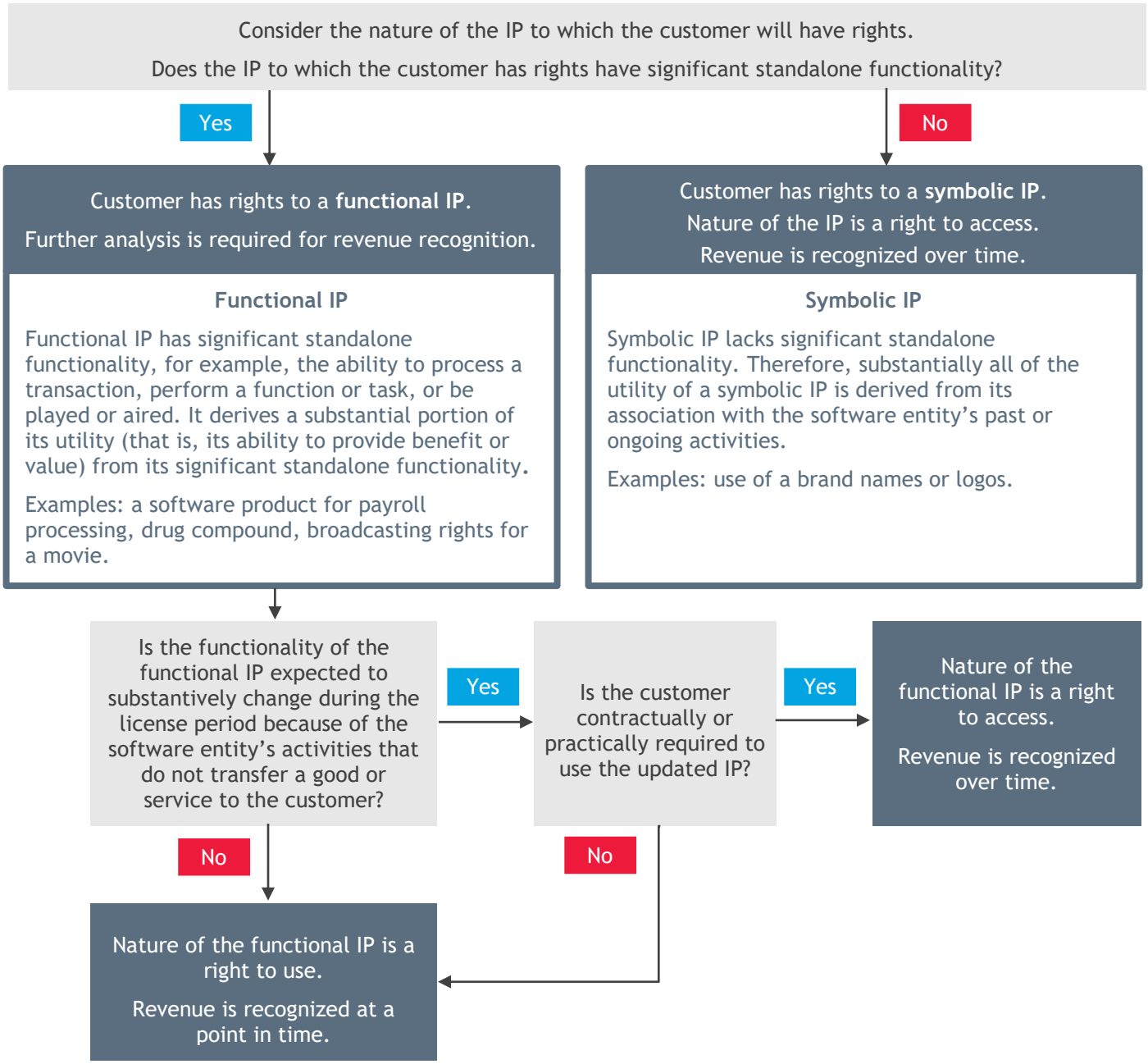
A software entity that grants a software license must apply the licensing guidance to determine the nature of its promise as either a right to access or a right to use the IP. This determination affects the conclusion about whether the software license transfers to a customer at a point in time or over time and hence affects whether the software entity recognizes revenue for the license at a point in time or over time.

Licensing Guidance: Nature of Software License — Right to Access Versus Right to Use

The licensing guidance requires a software entity to consider whether the nature of the software entity’s promise in granting a software license to a customer is to provide the customer with either:

- Right to access: A right to access the software entity's IP throughout the license period or its remaining economic life, if shorter
- Right to use: A right to use the software entity's IP as it exists at the point in time at which the license is granted.

To further assist software entities in evaluating the nature of an IP license, the licensing guidance defines an IP license as either a functional or symbolic IP. The following diagram depicts the decision process in evaluating whether a license of IP is a license of functional IP or symbolic IP, and in determining whether the nature of a software entity's promise in granting that license is to provide the customer a right to access or a right to use the IP.



Generally, a license to functional IP grants a right to use the software entity's IP as it exists at the point in time at which the license is granted. However, in certain instances, future changes to the underlying IP may result in the licenses being treated differently. Specifically, a license grants a right to access the software entity's IP that is recognized over time if it meets both of the following criteria in ASC 606-10-55-62:

- The functionality of the IP to which the customer has rights is expected to substantively change during the license period because of activities of the software entity that do not transfer a promised good or service to the customer. Additional promised goods or services (for example, IP upgrade rights or rights to use or access additional IP) are not considered in assessing this criterion.
- The customer is contractually or practically required to use the updated IP resulting from the activities in the previous criterion.

See Example 1 for an illustration of how to determine whether the nature of a software entity's promise in granting a software license is to provide the customer a right to use or a right to access the software entity's IP.

EXAMPLE 1 (ADAPTED FROM ASC 606-10-55-362 THROUGH 55-363B): RIGHT TO USE — SOFTWARE LICENSE

FACTS

- A software developer enters a contract with a customer to transfer a software license, perform an installation service, and provide unspecified software updates and technical support for a specified period.
- The software developer sells the license, installation service, and technical support separately.
- The installation service includes changing the web screen for each type of user (for example, marketing, inventory management, and IT). The installation service is routinely performed by competitors and does not significantly modify the software. The software remains functional without the updates and technical support.
- The software developer identifies four performance obligations in the contract:
  - The software license
  - Installation services
  - Software updates
  - Technical support
- The software developer assesses the nature of its promise to transfer the software license and concludes that the license is functional IP because the software has significant standalone functionality from which the customer can derive substantial benefit regardless of the software developer's ongoing business activities.

CONCLUSION

The software developer provides the customer with a right to use its software in the form existing on the license grant date and recognizes the software license performance obligation as satisfied at a point in time.

ANALYSIS

- The software developer considers whether the functional license represents a right to access IP and concludes:
  - While the functionality of the underlying software is expected to change during the license period because of the software developer's continued development efforts, the functionality of the software to which the customer has rights (that is, the customer's instance of the software) will change only because of the software developer's promise to provide when-and-if available software updates. Because the software developer's promise to provide software updates represents an additional promised service in the contract, the software developer's activities to fulfill that promised service are not considered in evaluating the two criteria in ASC 606-10-55-62 regarding whether a functional IP license represents a right to access.
  - The customer has the right to install, or not install, software updates when they are provided. Therefore, the criterion in in ASC 606-10-55-62 on whether the customer is contractually or practically required to use the updated IP would not be met even if the software developer's activities to develop and provide software updates met the other criterion in ASC 606-10-55-62 on whether the functionality of the IP to which the customer has rights is expected to substantively change during the license period because of activities of the software developer that do not transfer a promised good or service to the customer.
- Thus, the software developer concludes that it has provided the customer with a right to use its software as it exists when the license is granted. Accordingly, it accounts for the software license performance obligation as a performance obligation satisfied at a point in time.
- The software developer recognizes revenue for the software license at the point in time at which the control of the license transfers to the customer, which cannot occur before the customer can use and benefit from the license.

BDO INSIGHT 4: ANALYZING FUNCTIONAL IP REQUIRES JUDGMENT

In practice, we generally observe that most software licenses provide the customer with significant standalone functionality. As such, they represent functional IP that provide the customer with the right to use a software entity's IP as it exists at the point in time at which the license is granted. However, in certain limited scenarios a functional IP license may provide a customer with the right to access a software entity's IP because the ongoing updates are critical to maintaining the software's functionality. See Example 3 and the SEC staff guidance below for example fact patterns.

Determining whether the functionality of the IP is expected to substantively change during the license term or the ongoing updates are necessary to maintain the original functionality of the software requires significant professional judgment based on the facts and circumstances, including the customer's perspective of the utility to be derived from the IP. See BDO Insight 9 for additional information.

Guarantees given by a software entity that it has a valid patent to IP and that it will defend that patent from unauthorized use do not affect whether a license provides a right to access or a right to use the software entity's IP. Similarly, a promise to defend a patent right is not a promised good or service because it gives assurance to the customer that the license transferred meets the specifications of the license promised in the contract.

Fulfillment Activities

Activities performed to fulfill a contract that do not transfer goods or services to the customer are not considered promised goods or services in a contract, even though those activities are required to successfully transfer the goods or services for which the customer has contracted. Fulfillment activities may exist in a contract in which a software entity undertakes separate activities that do not directly transfer goods or services to a customer. For example, a software entity may need to perform various administrative tasks or set up activities to establish a customer's access in the software system before providing SaaS to the customer. The performance of the administrative tasks (or set up activities) does not transfer a service to the customer as the tasks are performed and, therefore, those activities are not promised goods or services in the contract. See the following discussion on nonrefundable upfront fees to determine whether a nonrefundable upfront fee relates to fulfillment activities or promised goods or services.

BDO INSIGHT 5: DISTINGUISHING SET-UP ACTIVITIES FROM PROMISED GOODS OR SERVICES

Administrative tasks are sometimes referred to as "set-up activities." Distinguishing set-up activities from promised goods or services in the contract can be challenging. We believe the following factors may be relevant when making the distinction:

- Does the customer continue to benefit from the activities after they are performed? Customers continue to benefit from promised goods and services after control of the promised goods or services has transferred to them. Conversely, customers do not benefit from set-up activities after a software entity completes such activities. For example, a software entity may perform tasks to provide customers with access to the promised goods or services. Such tasks are set-up activities because the tasks do not benefit the customer after the promised goods or services are transferred.
- Can other entities perform the services? If the activities cannot be performed by the customer or other entities, they may be set-up activities. Conversely, if the activities can be performed by the customer or other entities, they may be promised services.
- In performing the set-up activities, is the software entity enhancing its own assets or the customer's? If performing the services enhances or improves the software entity's assets, the activities are set-up activities because the software entity retains control over any benefits derived therefrom. Conversely, if the customer's assets are being enhanced, the activities may be promised services.

- Does the customer simultaneously receive and benefit from the activities? When customers simultaneously receive and consume the services being performed, an asset is created, if only momentarily. That is, customers obtain control of a software entity's outputs as it performs, and the activities being performed are promised services.

All factors must be holistically evaluated and no one factor is conclusive. Determining whether activities in the contract constitute set-up activities requires the application of professional judgment based on the facts and circumstances.

BDO INSIGHT 6: ANALYZING INITIAL ADMINISTRATIVE OR SET UP ACTIVITIES

A software entity may perform certain initial administrative or set up activities when selling a software license or SaaS to a customer. To determine whether those activities are separate promises (and potential performance obligations) or fulfillment activities, the software entity must determine whether those activities provide an incremental benefit to the customer (that is, a benefit beyond granting the software license or access to SaaS).

For example, initial account set up activities may be necessary for the customer to access the hosted software to derive the benefits of a SaaS subscription. In that case, the initial account set up activities do not directly transfer additional services to the customer and hence are not separate promises.

Conversely, a software entity may perform certain services upfront that provide incremental benefits to the customer (that is, benefits beyond granting the software license or access to SaaS). For example, a software entity may provide data migration services to migrate historical data from a customer's existing systems to its SaaS platform. Additional examples of initial activities in a software arrangement that might be separate promises rather than fulfillment activities include certain customer-specific customization or training to use the software effectively.

We generally observe that initial activities that can be provided by a third party are promised services, not fulfillment activities. However, the converse is not true. An initial activity that cannot be provided by a third party may nonetheless represent either a promised service or a fulfillment activity. For example, consider a software entity that contracts with a customer for a SaaS subscription along with proprietary training services. The training allows the customer to use the software more efficiently and effectively in the customer's business environment. A third party cannot provide the training service. Regardless, the training is a separate promise because it provides the customer benefits that are incremental to accessing the SaaS.

Exception for Immaterial Promises in a Contract

A software entity does not need to assess whether promised goods or services are performance obligations if they are immaterial in the context of the contract with the customer. In assessing immateriality, a software entity considers both the quantitative and qualitative nature of the promised goods or services in the contract. Note that the exception for immaterial promises does not apply to a customer's option to acquire additional goods or services that provides the customer with a material right.

BDO INSIGHT 7: IDENTIFYING IMMATERIAL PROMISES IN SOFTWARE ARRANGEMENTS

Assessing whether a promise is immaterial in the context of a contract requires consideration of both the quantitative value of the promised good or service and qualitative factors from the customer's perspective. Qualitative factors to consider may include:

- Why the software entity included the promise in the contract.
- Whether the promise represents a key feature of the arrangement or differentiates the software entity's offering.
- Whether the promise is prominently described in the contract or otherwise emphasized during the sales process.
- Whether the customer negotiated for the promise.
- Whether the promise is important to the customer.
- Whether the promise influences the customer's decision to enter the arrangement.
- Whether the customer would view the failure to deliver the promise as a significant shortfall in the arrangement.
- Whether the customer likely would place significant value on the promise even if its value is small relative to the overall contract.
- Whether the customer likely would have received substantially the same benefit from the arrangement without the promise.
- Whether the promise is ancillary to the arrangement or integral to the customer obtaining the expected benefit of the contract.

No single factor is determinative.

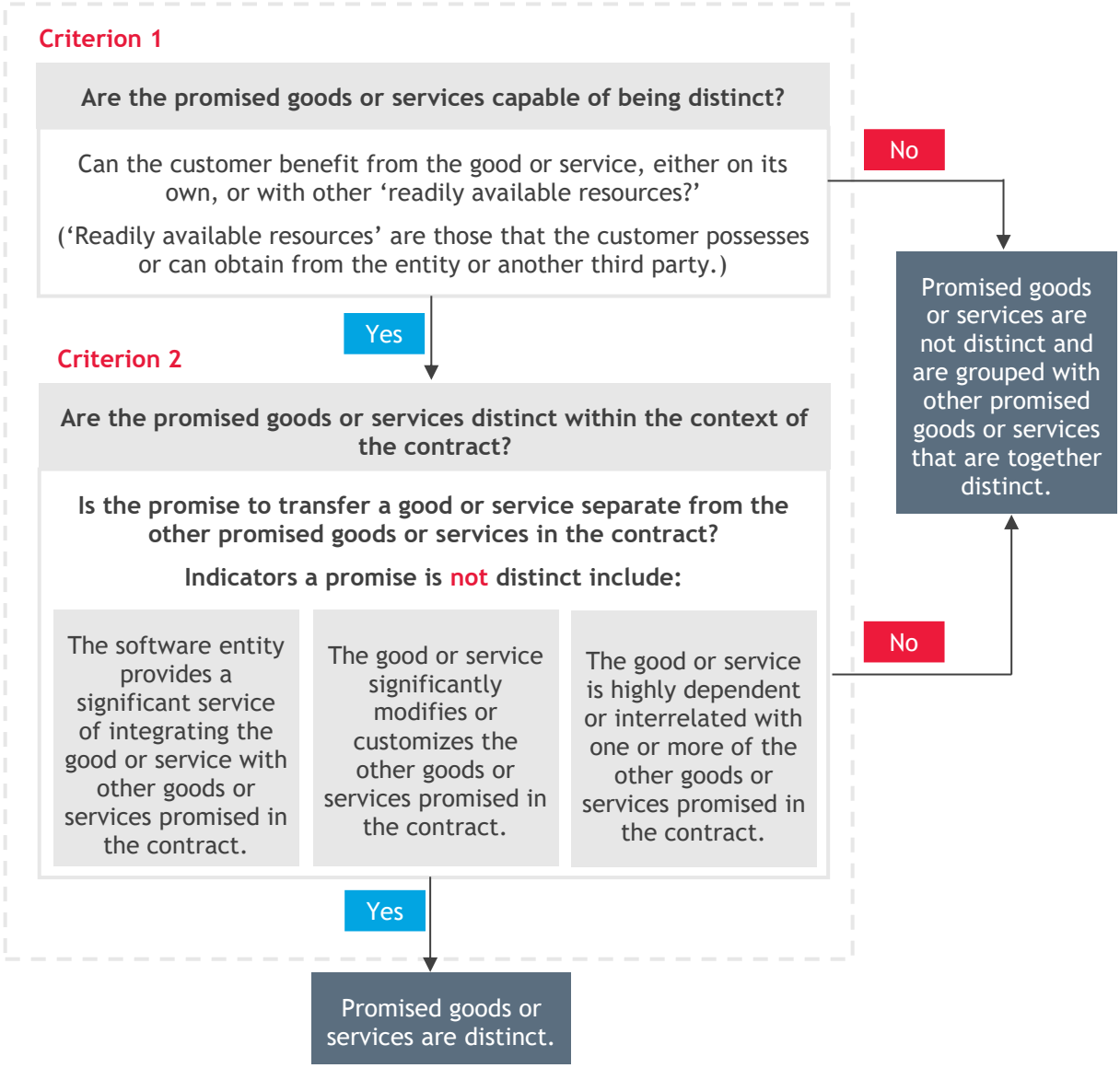
Additionally, a software entity must consider whether promised goods or services that are individually immaterial are material in the aggregate. Multiple promises that appear insignificant when evaluated separately may represent a material portion of the overall arrangement when considered together.

Because the promises are evaluated from the customer's viewpoint for potential immateriality, we generally do not observe immaterial promises in a contract for a software license or SaaS. However, reaching a conclusion about whether a promised good or service is immaterial in the context of a contract requires the application of professional judgment based on the facts and circumstances.

ARE THE PROMISED GOODS OR SERVICES DISTINCT?

After identifying the promised goods or services in the contract, a software entity determines which of those promised goods or services are distinct and, hence, represent separate performance obligations. If a promised good or service is not distinct, the software entity must combine that good or service with other promised goods or services until a bundle of goods or services that is distinct can be identified.

The following diagram illustrates the guidance on determining whether a promised good or service is distinct, and hence a separate performance obligation:



As illustrated in the diagram, for a good or service to be distinct, it must meet both of the following criteria:

- The good or service is capable of being distinct: The customer can benefit from the good or service either on its own or with other resources that are readily available to the customer.
- The promise to transfer the good or service is distinct within the context of the contract: The software entity's promise to transfer the good or service to the customer is separately identifiable from other promises in the contract.

Criterion 1: Good or Service Is Capable of Being Distinct

A good or service is capable of being distinct if a customer can benefit from the good or service either on its own or with other resources that are readily available to the customer. A customer can benefit from a good or service if the good or service can be used, consumed, or sold (other than for scrap value), or held in any other way to generate economic benefits. In some instances, a customer may only be able to obtain benefits from goods or services in conjunction with other readily available resources.

A readily available resource is either:

	A good or service that is sold separately (either by the software entity or a third party)
	A resource that the customer has already obtained from the software entity, including a good or service that the software entity has already transferred to the customer under the contract or from other transactions or events

Various factors can corroborate whether the customer can benefit from a good or service, either on its own or in conjunction with other readily available resources. For example, if a software entity regularly sells a good or service separately, that indicates that a customer can benefit from that good or service either on its own or in conjunction with other readily available resources. If a good or service is not capable of being used on its own or with other resources, conceptually there would not be a market for a software entity to provide that good or service on a standalone basis.

Additionally, the assessment of whether the "customer can benefit from the goods or services on its own" is based on the characteristics of the goods or services themselves rather than the way in which the customer contractually may use or obtain the goods or services. Therefore, a software entity disregards any contractual limitations that might prevent the customer from obtaining readily available resources from a source other than the software entity.

Criterion 2: Promise to Transfer Good or Service Is Distinct Within the Context of the Contract

A software entity's promise to transfer a good or service is distinct within the context of the contract if that promise is separately identifiable from other promises in the contract. The objective of assessing whether a software entity's promise to transfer a good or service is separately identifiable from other promised goods or services in the contract is to determine whether the nature of the promise, within the context of the contract, is to transfer each of those goods or services individually or instead, to transfer a combined item(s) to which the promised goods or services are inputs.

Determining whether a software entity's promise to transfer a good or service is separately identifiable in a contract requires the application of professional judgment based on the facts and circumstances for each contract with a customer.

ASC 606 specifies three factors to assist a software entity in making that judgment.

These following factors indicate that a software entity's promise to transfer two or more goods or services to the customer are not separately identifiable:



The software entity provides a significant service of integrating goods or services with other goods or services promised in the contract into a bundle of goods or services that represents the combined output(s) for which the customer has contracted



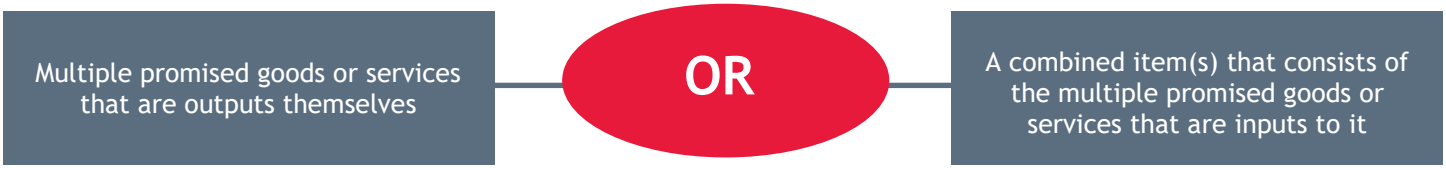
One or more of the goods or services significantly modifies or customizes or are significantly modified or customized by one or more of the other goods or services promised in the contract



The goods or services are highly interdependent or highly interrelated

The “separately identifiable” principle is influenced by the notion of separable risks, which is whether the risk that a software entity assumes to fulfill its obligation to transfer one of those promised goods or services to the customer is inseparable from the risk relating to the transfer of the other promised goods or services in the contract.

The intent of the guidance is that a software entity must evaluate whether a contract is to deliver:



For multiple promised goods or services to be considered inputs to a combined item(s), a software entity’s promise to transfer the promised goods or services must result in a combined item(s) that is(are) greater than (or substantively different from) the sum of those promised goods and services.

The “separately identifiable” principle is intended to consider the level of integration, interrelation, or interdependence among promises to transfer goods or services. In other words, the separately identifiable principle aims to evaluate when a software entity’s performance in transferring a bundle of goods or services in a contract is in substance fulfilling a single promise to a customer. Therefore, the software entity evaluates whether two or more promised goods or services each significantly affect the other and thus are highly interdependent or highly interrelated. The software entity cannot merely evaluate whether one item by its nature depends on the other. For example, the fact that a customer would never purchase PCS absent the purchase of the related software license in that contract or an earlier contract does not indicate that the license and PCS each significantly affect the other.

The three factors are intended to supplement the “separately identifiable” principle. Thus, they are not an exhaustive list, and not all of them need to be met (or not met) to conclude that the software entity’s promises to transfer goods or services are not (are) separately identifiable. Furthermore, the factors are more or less relevant to the evaluation of the “separately identifiable” principle in a given contract and therefore, software entities must consider the principle, not only the previously mentioned three factors.



Significant Integration Service

Two or more promises to transfer goods or services to a customer are not separately identifiable if a software entity provides a significant service of integrating those goods or services with other goods or services promised in the contract into a bundle that represents the combined output or outputs for which the customer has contracted.

In other words, the software entity is using the goods or services as inputs to produce or deliver the combined output(s) specified by the customer. A combined output may include more than one phase, element, or unit. When a software entity provides an integration service, the risk of transferring the individual goods or services is inseparable, because a substantial part of the software entity’s promise is to incorporate the individual goods and services into the combined output.



Significant Modification or Customization

Two or more promises to transfer goods or services to a customer are not separately identifiable if one or more of the goods or services significantly modifies or customizes, or is significantly modified or customized by, another good or service promised in the contract. If a good or service significantly modifies or customizes another good or service in the contract, each good or service is being assembled (that is, as an input) to produce a combined output for which the customer has contracted.

This factor may be relevant in a software development contract in which a software developer customizes a software product for a specific customer. ASC 606 includes an example fact pattern to illustrate whether a significant customization service provided by a software developer in a contract with a customer is distinct from a software license (see Example 2).

EXAMPLE 2 (ADAPTED FROM ASC 606-10-55-141 THROUGH 55-150): SOFTWARE DEVELOPMENT CONTRACT — SIGNIFICANT CUSTOMIZATION IS PROVIDED

FACTS

- A software developer contracts with a customer to:
  - Transfer a software license
  - Perform a customized installation service
  - Provide unspecified software updates and technical support for a specified period
- The software developer sells the license, installation service, and technical support separately.
- As part of the installation service, the software developer is required to substantially customize the software to add significant new functionality to enable the software to interface with other customized software applications used by the customer. Other competitor entities can provide the customized installation service.
- The software remains functional without the updates and the technical support.

CONCLUSION

The software developer identifies three performance obligations in the contract:

- Software customization (comprised of the software license and the customized installation service)
- Software updates
- Technical support

ANALYSIS

Capable of Being Distinct

The software developer assesses the promised goods and services and concludes that the customer can benefit from each of the goods and services either on its own or with the other goods and services that are readily available (that is, the goods and services are capable of being distinct) because:

- The software is delivered before the other goods and services and remains functional without the updates and the technical support.
- The customer can benefit from the updates with the software license transferred at the outset of the contract.

Distinct in the Context of the Contract

The software developer then considers the following:

- The contractual terms result in a promise to provide a significant service of integrating the licensed software into the existing software system by performing a customized installation service as specified in the contract. In other words, the software developer is using the license and the customized installation service as inputs to produce the combined output (that is, a functional and integrated software system) in accordance with the customer's specifications.
- The installation service significantly modifies and customizes the software.

Therefore, the software developer determines that the promise to transfer the software license is not separately identifiable from the promise to provide customized installation service and those promises are not distinct in the context of the contract. Therefore, the software license and the customized installation service are combined into a single performance obligation.

Additionally, the software developer concludes that the software updates and technical support are distinct from the other promises in the contract because:

- The software updates do not significantly affect the customer's ability to use and benefit from the software license because the software updates are not necessary for the software to maintain a high level of utility to the customer during the license period.
- Neither the software updates nor the technical support significantly modifies or customizes one another or the software license bundled with the customized installation service.
- The software developer is not providing a significant service of integrating the updates or technical support into a combined output with the software license.
- Neither the software updates nor the technical support significantly affects each other or the software license bundled with the customized installation service, and therefore they are not highly interdependent or highly interrelated with another promise. The software developer would be able to fulfill its promise to transfer the initial customized software license independent from its promise to subsequently provide the software updates or technical support.



High Interdependency or Interrelation

Two or more promises to transfer goods or services to a customer are not separately identifiable if the goods or services are highly interdependent or highly interrelated. That is, each of the goods or services is significantly affected by one or more of the other goods or services in the contract. For example, two or more goods or services may be significantly affected by each other because the software entity would not be able to fulfill its promise by transferring each of the goods or services independently.

When considering whether the promises in a contract are highly interdependent or interrelated, it is important for a software entity to consider whether one promise in a contract has a transformative effect on another promise rather than merely an additive effect. The fact that a good or service depends on a second good or service (that is, they have a functional relationship) does not necessarily mean the two promises are not distinct in the context of the contract. Instead, the dependency between the promises must be bi-directional, that is, they must each effect the other.

BDO INSIGHT 8: IDENTIFYING DISTINCT GOODS OR SERVICES REQUIRES JUDGMENT

The identification of distinct goods or services in a contract requires the application of professional judgment based on the facts and circumstances. A software entity may need to perform a detailed analysis of contractual terms and conditions, unique facts and circumstances, and customer expectations and apply judgment to determine whether a promise in a contract is a distinct good or service (and thus constitutes a performance obligation) or must be combined with other promises in the contract to create a single performance obligation. A software entity may also consider other information, such as marketing materials, to assist in its determination of its performance obligations. However, all facts and circumstances must be considered because marketing descriptions are not determinative, as discussed by the SEC (excerpted below).

SEC STAFF GUIDANCE

Remarks before the 2018 AICPA Conference on Current SEC and PCAOB Developments

Sheri L. York, Professional Accounting Fellow, Office of the Chief Accountant

December 10, 2018

Identification of Performance Obligations

. . . In a recent consultation with OCA, a registrant provided its customer with a commercial security monitoring service by integrating a variety of cameras and sensors (which I will refer to as "equipment") with the entity's technology platform. The equipment was integrated via a control panel that was installed at the customer's location and enabled communication between the equipment and the registrant's technology platform. The registrant's technology platform also incorporated an element of artificial intelligence that used data from the cameras and sensors to learn the patterns of the customer's behavior. It then used that information to create a "smart" security monitoring service. For example, motion detectors may identify an attempt to open a window while other sensors may simultaneously indicate, based on a lack of body temperature readings, that personnel are not currently located within the building. In this case, the control panel would route this information obtained from the equipment to the registrant's technology platform, which may alert the customer and/or the authorities about a potential issue.

The registrant concluded that each piece of equipment (including the control panel), the installation, and the monitoring services were capable of being distinct, but believed that these promises comprised a single performance obligation as they were not distinct in the context of the contract. The registrant believed it was providing a significant service of integrating the goods and services in the contract into a bundle that represented the combined output for which the customer had contracted. More specifically, the delivery of a "smart" security monitoring service would not be possible if the equipment were not integrated with the technology platform.

The [SEC] staff did not object to the registrant's conclusion and considered it reasonable to conclude that the nature of the promise is to transfer a combined item – the commercial security solution – to which each piece of equipment (including the control panel), the technology platform, and installation are inputs. In this fact pattern, the entity demonstrated reasonable judgment that they were providing a significant integration service that transformed the equipment and services into a combined output that provided the customer with an overall service offering that was greater than the customer could receive from each individual part.

SEC STAFF GUIDANCE

Remarks before the 2019 AICPA Conference on Current SEC and PCAOB Developments

Susan M. Mercier, Professional Accounting Fellow, Office of the Chief Accountant

December 9, 2019

> *A "Solution"*
>
> *. . . While the majority of registrants that consult with us on the identification of performance obligations provide a robust analysis addressing the factors in Topic 606, we sometimes hear registrants support their conclusion in consultation submissions or responses to comment letters from the Division of Corporation Finance by simply stating that they are providing a "solution," that the customer wants a combined "solution," or that there is only one promise in the contract to provide a "solution."*
>
> *While I understand that the term "solution" is commonly used nomenclature, I would observe that the staff is not persuaded that promises should be combined into a single performance obligation simply because an entity labels those promises as a "solution" that the "customer wants." Rather, a registrant must support its assertion that its promises to transfer goods or services are not separately identifiable based on the guidance in ASC 606-10-25-21, which describes the objective of the analysis and provides factors to consider.*
>
> *In addition to the guidance in ASC 606, I think it is also helpful to remember the discussion in the Basis for Conclusions of ASU 2016-10. I think that the notion of considering if an entity's combined output is greater than or substantively different from the sum of the parts is helpful in many cases.*

Utility of Promised Good or Service: Customer's Perspective

When evaluating whether two or more promises in a contract are separately identifiable, a software entity must consider the utility of the promised goods or services (that is, the ability of each good or service to provide benefit or value) to the customer. While a software entity may be able to fulfill its promise to transfer each good or service in a contract independently of the other, each good or service may significantly affect the other's utility to the customer.

Utility is relevant in evaluating whether two or more promises in a contract are separately identifiable because even if two or more goods or services are capable of being distinct because the customer can derive some economic benefit from each one, the customer's ability to derive its intended benefit from the contract may depend on the software entity transferring both of the goods or services.

For example, the FASB considered a scenario in which a software entity contracts with a customer to provide an anti-virus software license with updates, as illustrated in Example 3. In that scenario, the software entity's ability to transfer the initial license is not affected by its promise to transfer the updates or vice versa, but the provision (or not) of the updates will significantly affect the utility of the licensed IP to the customer such that the license and the updates are not separately identifiable. They are in effect inputs to the combined solution for which the customer contracted (that is, protection from viruses).

EXAMPLE 3 (ADAPTED FROM ASC 606-10-55-140D THROUGH 55-140F): LICENSE TO ANTI-VIRUS SOFTWARE WITH WHEN-AND-IF AVAILABLE UPDATES

FACTS

- A software entity grants a customer a three-year term license to an anti-virus software and promises to provide the customer with when-and-if available updates to that software during the license period.
- The software entity often provides updates that are critical to the continued utility of the anti-virus software.
- Without those updates, the customer's ability to benefit from the anti-virus software would decline significantly during the three-year arrangement.

CONCLUSION

The software entity accounts for the anti-virus software license and the when-and-if available updates as a single performance obligation because the promises are not separately identifiable in the context of the contract.

ANALYSIS

Capable of Being Distinct

The software entity determines that the anti-virus software and the updates are each capable of being distinct because:

- The customer can derive economic benefit from the software on its own throughout the license period (that is, without the updates the software would still provide its original functionality to the customer).
- The customer can benefit from the updates with the software license transferred at the outset of the contract.

Distinct in the Context of the Contract

- The software entity determines that its promises to transfer the anti-virus software and the unspecified updates, when-and-if available, are not separately identifiable because the license and the updates are, in effect, inputs to a combined item (anti-virus protection) in the contract.
  - The unspecified updates will significantly modify the software's functionality (that is, they permit the software to protect the customer from numerous additional viruses that the software did not protect against previously)
  - The updates are integral to maintaining the utility of the software license to the customer.
- Consequently, the anti-virus license and periodic unspecified updates fulfill a single promise to the customer in the contract: a promise to provide protection from computer viruses for three years.

In addition to the anti-virus scenario, a member of the SEC staff elaborated on a consultation in which the SEC staff did not object to the conclusion that a software license and updates represent a single, combined performance obligation:

SEC STAFF GUIDANCE

Remarks before the 2019 AICPA Conference on Current SEC and PCAOB Developments

Susan M. Mercier, Professional Accounting Fellow, Office of the Chief Accountant

December 9, 2019

License to Software with Updates — Single Performance Obligation

In the fact pattern consulted on, a software entity licenses its software that allows its customers, application (app) developers, to build and deploy, and therefore monetize, their own apps on various third-party platforms. The third-party platforms include phones and home entertainment systems, which often undergo software updates as well. The software entity's software and updates make sure that the app built by customers using the software is compatible with all platforms that it supports, both when the app is initially deployed on a platform and over time as that platform is updated. To maintain continued compatibility of the software with third-party platforms, the software entity partners with the third-party platforms to understand their timelines for internal updates so that the software entity can timely initiate corresponding updates to its software. Without those software updates, the customer's ability to benefit from the software would be significantly limited over the contract term.

The SEC staff did not object to the software entity's conclusion that the software license and updates represent a single performance obligation because, in the SEC staff's view, the promises to provide the software license and the updates are, in effect, inputs that fulfill a single promise to the customer, which is to continually be able to deploy and monetize content using third-party platforms of the customer's choice in a rapidly changing environment, and that the updates are integral to maintaining the utility of the software license. The SEC staff stated that in this fact pattern, the combined output (whether marketed as a "solution" or not) is greater than, or substantively different than, the individual promises (that is, the software license and the updates).

BDO INSIGHT 9: DETERMINING WHETHER A SOFTWARE LICENSE AND UPDATES ARE A SINGLE PERFORMANCE OBLIGATION

In the fact pattern discussed in the SEC staff speech and the example of an anti-virus software license (Example 3), the software entity concluded that the software license and related updates are a single performance obligation. We believe it may be possible to reach the same conclusion in additional fact patterns, although we believe that will be rare.

Key factors we believe may be helpful in determining whether a software license and updates are a single performance obligation include:

- How frequently updates are available
- Whether the updates are driven by factors within the software entity's control
- How significantly the functionality of the software is affected by the upgrades
- How often, and how many, customers elect not to adopt an upgrade and continue to derive the intended benefit from the software license.

Another area of complexity results when a software entity delivers a hybrid solution that includes both cloud-based functionality and an on-premise license, as illustrated in the following example.

EXAMPLE 4: IDENTIFYING PERFORMANCE OBLIGATIONS IN A CONTRACT FOR A HYBRID CLOUD-BASED SOLUTION

FACTS

- A software entity provides a cloud-based business productivity and collaboration solution to customers that includes both cloud-based SaaS and on-premise software licenses for desktops and mobile devices.
- The on-premise software and cloud-based SaaS each provide certain functionalities. However, certain significant functionalities are only available through the interaction of cloud-based features with the on-premise software.

CONCLUSION

The software entity concludes that the contract includes one performance obligation.

ANALYSIS

- In identifying the performance obligations in a contract with a customer for the cloud-based solution, the software entity considers that the customer derives the expected benefits from the solution because of the interaction between the on-premise software and the cloud-based features of the solution. The customer cannot derive the intended benefits through the on-premise software alone because the cloud-based service includes significant features and functionalities that the customer can access only when connected to the cloud.
- Therefore, the software entity determines that the cloud-based features are integral to the promised solution and work together with the on-premise software to create significant features and functionalities promised to the customer.
- The software entity concludes that the contract includes one performance obligation.

Attributes of a License Versus an Additional License

Software licenses frequently include restrictions of time, geographical region, or use. For example, a customer may be permitted to use a software license for two years within the U.S. Additionally, there might be restrictions regarding the usage of the software license. For example, a customer may be permitted to use a software application for non-commercial purposes only.

Contractual provisions that explicitly or implicitly require a software entity to transfer control of additional goods or services to a customer (for example, by requiring the software entity to transfer control of additional rights to use or rights to access IP that the customer does not already control) must be distinguished from contractual provisions that explicitly or implicitly define the attributes of a single promised license (for example, restrictions of time, geographical region, or use). Attributes of a promised license define the scope of a customer's right to use or right to access the software entity's IP and therefore do not define whether the software entity satisfies its performance obligation at a point in time or over time and do not create an obligation for the software entity to transfer any additional rights to use or access its IP.

Typically, restrictions of time, geographical region, or use are considered attributes of a license and not separate promises or licenses. However, certain restrictions may effectively be a promise to grant additional licenses at a future date. See Examples 5 and 6 for illustrative fact patterns.

EXAMPLE 5: ATTRIBUTES OF A LICENSE

FACTS

A software entity's contract with a customer includes the following key terms:

- The software entity grants to the customer a four-year term license to a payroll processing software for 25 authorized users at contract inception
- The customer can use the license within the U.S.

CONCLUSION

The contract includes a single license transferred at contract inception.

ANALYSIS

The software entity determines that the restrictions of time (four-year term) and geographical region (the U.S. only) define the attribute of a single license transferred at contract inception. There are no additional promises or licenses to be delivered by the software entity in the future.

EXAMPLE 6: ATTRIBUTES OF A LICENSE VERSUS ADDITIONAL LICENSE

FACTS

Consider the same facts in Example 5 except that the number of authorized users increases to 35 at the beginning of Year 2 of the license term.

CONCLUSION

The contract includes two distinct promises.

ANALYSIS

The software entity determines that the customer does not control the additional rights for 10 more users at contract inception. Rather, the software entity must grant additional rights to the customer to use its software at the beginning of Year 2.

Therefore, at contract inception, the contract includes two promises:

- At contract inception, a 4-year license to 25 users.
- At the beginning of Year 2, a 3-year license to 10 users.

Exclusivity is generally an attribute of a license rather than a separate promise. While exclusivity may increase the value of the license in the marketplace, it does not change the nature of the underlying IP nor the software entity's promise. Therefore, a software entity does not separately account for exclusivity in a license arrangement, and the exclusivity does not affect whether that license is transferred at a point in time or over time.

Combining a Good or Service with Other Promised Goods or Services and Applying the Licensing Guidance to the Bundle

If a good or service is not distinct, a software entity must combine that good or service with other promised goods or services until it identifies a bundle of goods or services that is distinct. If the promise to grant a license is not distinct from other promised goods or services in the contract, a software entity must account for the promise to grant a license and those other promised goods or services together as a single performance obligation.

When a single performance obligation includes a license (or licenses) of IP and one or more other goods or services, a software entity must consider whether the license that is part of the single performance obligation provides the customer with a right to use or a right to access IP to determine the nature of the combined good or service for which the customer has contracted.

Following are examples of licenses that are not distinct from other goods or services promised in the contract:

- A license that forms a component of a tangible good and is integral to the functionality of the good
- A license that the customer can benefit from only in conjunction with a related service, for example, an online service provided by the software entity that enables, by granting a license, the customer to access content.

Firmware

Many consumer products include embedded software that is required for the product to function as intended. This type of software is often called firmware. A license that forms a component of a tangible good and is integral to the functionality of the good is not distinct from the good. In that case, the tangible good and license are accounted for as a single performance obligation. As a result, firmware is typically combined with the related product and accounted for as a single performance obligation.

In a growing number of instances, a software entity may also provide functionality related to a consumer product through a mobile application available at no cost. That software entity needs to determine whether the application either:

- Is integral to the functionality of the product and is thus combined with the product and firmware as a single bundled performance obligation
- Represents incremental functionality that is a distinct performance obligation

See Examples 7 and 8 for illustrative fact patterns.

EXAMPLE 7: HARDWARE WITH EMBEDDED FIRMWARE AND DISTINCT MOBILE APPLICATION

FACTS

- A software entity sells a blood glucose monitor with a limited license to the embedded firmware on the monitor.
- A customer cannot determine blood glucose levels as promised without the license to the firmware.
- The software entity makes available a mobile application that the customer can download for free onto a mobile device to use with the monitor.
- The blood glucose monitor includes a user interface, which, when used properly, visually reports blood glucose levels of the user.
- The mobile application provides additional functionality, including wirelessly connecting to the monitor to receive measurements, storing the measurements, and predicting when a user may experience high or low measures based on that history.

CONCLUSION

The embedded license is not a separate performance obligation, but the application is distinct from the blood glucose monitor (i.e., the application is a separate performance obligation from the monitor).

ANALYSIS

- The embedded firmware is a component of the blood glucose monitor and is integral to its functionality. As such, the license to the firmware is not capable of being distinct.
- The software entity then considers whether the mobile application is capable of being distinct from the blood glucose monitor. The software entity concludes that the blood glucose monitor is capable of being distinct because a user can obtain the promised benefit of determining blood glucose levels without downloading the mobile application. In addition, because the mobile application can be used in conjunction with the blood glucose monitor that a customer already owns, it is also capable of being distinct from the monitor.

- The software entity then determines whether the mobile application is distinct in the context of the contract. The software entity notes that while the mobile application is dependent upon the blood glucose monitor, the monitor is not dependent upon the mobile application. That is, the interdependency between the mobile application and the monitor is not bi-directional. In addition, neither the mobile application nor the monitor modifies or customizes the other, and the software entity does not perform a significant service of integrating the monitor and mobile application into a combined output.
- Therefore, the software entity concludes that the mobile application is distinct from the blood glucose monitor.

EXAMPLE 8: HARDWARE WITH EMBEDDED FIRMWARE AND BUNDLED MOBILE APPLICATION

FACTS

Assume the same fact pattern as in Example 7 except:

- The blood glucose monitor does not include a user interface. Instead, it automatically delivers the measurements wirelessly to the mobile application.
- A user must use the mobile application to receive the blood glucose measurement taken by the monitor.
- The mobile application also provides tracking and charting of historical measurements, and predictive analysis based on that history.

CONCLUSION

The software entity identifies a single performance obligation consisting of the blood glucose monitor, the embedded firmware, and the mobile application.

ANALYSIS

- Consistent with Example 7, the software entity concludes that the firmware is not distinct from the blood glucose monitor.
- However, in this example, the software entity concludes that the mobile application is also not capable of being distinct. Specifically, the blood glucose monitor cannot function as intended by a user without the mobile application because the mobile application is the only mechanism by which the user can obtain his or her blood glucose measurements. Therefore, the software entity concludes that it has a single performance obligation consisting of the blood glucose monitor, the firmware, and the mobile application.
- Because the software entity bundles the mobile application with the monitor and firmware into a single performance obligation, it must recognize the revenue over the period in which it is transferring control of the combined promise to the customer.

Concurrently Satisfied Performance Obligations

ASC 606 does not specify the accounting for concurrently delivered distinct goods or services that have the same pattern of transfer. Based on the FASB's observation in BC116 of ASU 2014-09, a software entity is not precluded from accounting for the goods or services in a contract that are delivered concurrently as if they were a single performance obligation if the outcome is the same as accounting for the goods and services as individual performance obligations.

For example, a software entity may account for SaaS and customer support or the individual components of PCS as a single performance obligation if it provides those services concurrently over the same term and uses the same measure of progress for satisfaction of a performance obligation.

BDO INSIGHT 10: MODIFICATION OF A CONTRACT THAT INCLUDES CONCURRENTLY SATISFIED PERFORMANCE OBLIGATIONS

Even if a contract includes multiple performance obligations that are satisfied concurrently and could practically be accounted for as a single performance obligation, a software entity must correctly identify separate performance obligations if that contract is modified. Accounting for a contract modification is based on, among other things, whether a promised good or service is distinct (and therefore a performance obligation) and whether a performance obligation in a contract has been satisfied before the modification date. Those considerations affect the amount and timing of revenue recognized at or after the modification date.

Identifying performance obligations and accounting for contract modifications requires the application of professional judgment based on the facts and circumstances.

SERIES OF DISTINCT GOODS OR SERVICES

A software entity must consider the applicability of the series guidance in circumstances in which it provides the same good or service repeatedly over a period of time, for example, in a repetitive service contract, such as for PCS or SaaS.

The series guidance states that a series of distinct goods or services that are substantially the same and that have the same pattern of transfer to the customer are considered one performance obligation in the contract for applying the five-step revenue recognition model. When the series guidance applies to a promise to transfer a series of distinct goods or services, a software entity identifies a single performance obligation and allocates the transaction price to that performance obligation in Step 4. The software entity then recognizes revenue in Step 5 by applying a single measure of progress to that performance obligation.

A series of distinct goods or services has the same pattern of transfer to the customer if it meets **both** of the following criteria:

	Each distinct good or service in the series that the software entity promises to transfer to the customer would meet the criteria in ASC 606-10-25-27 to be a performance obligation satisfied over time.
	The same method would be used to measure the software entity's progress toward complete satisfaction of the performance obligation to transfer each distinct good or service in the series to the customer.

In determining whether the series guidance applies, a software entity must first determine the nature of the goods or services promised to the customer before assessing whether the promised services are distinct and substantially the same. In some cases, this analysis may include determining whether the promise is the actual delivery of a specified quantity of service or the act of standing ready to perform.

In assessing whether multiple goods or services are treated as a series, it is not necessary that:

- The goods or services be transferred consecutively
- The accounting result from applying the series guidance be substantially the same as would result from accounting for each promised good or service as a separate performance obligation

BDO INSIGHT 11: SERIES GUIDANCE IS MANDATORY IF IT APPLIES

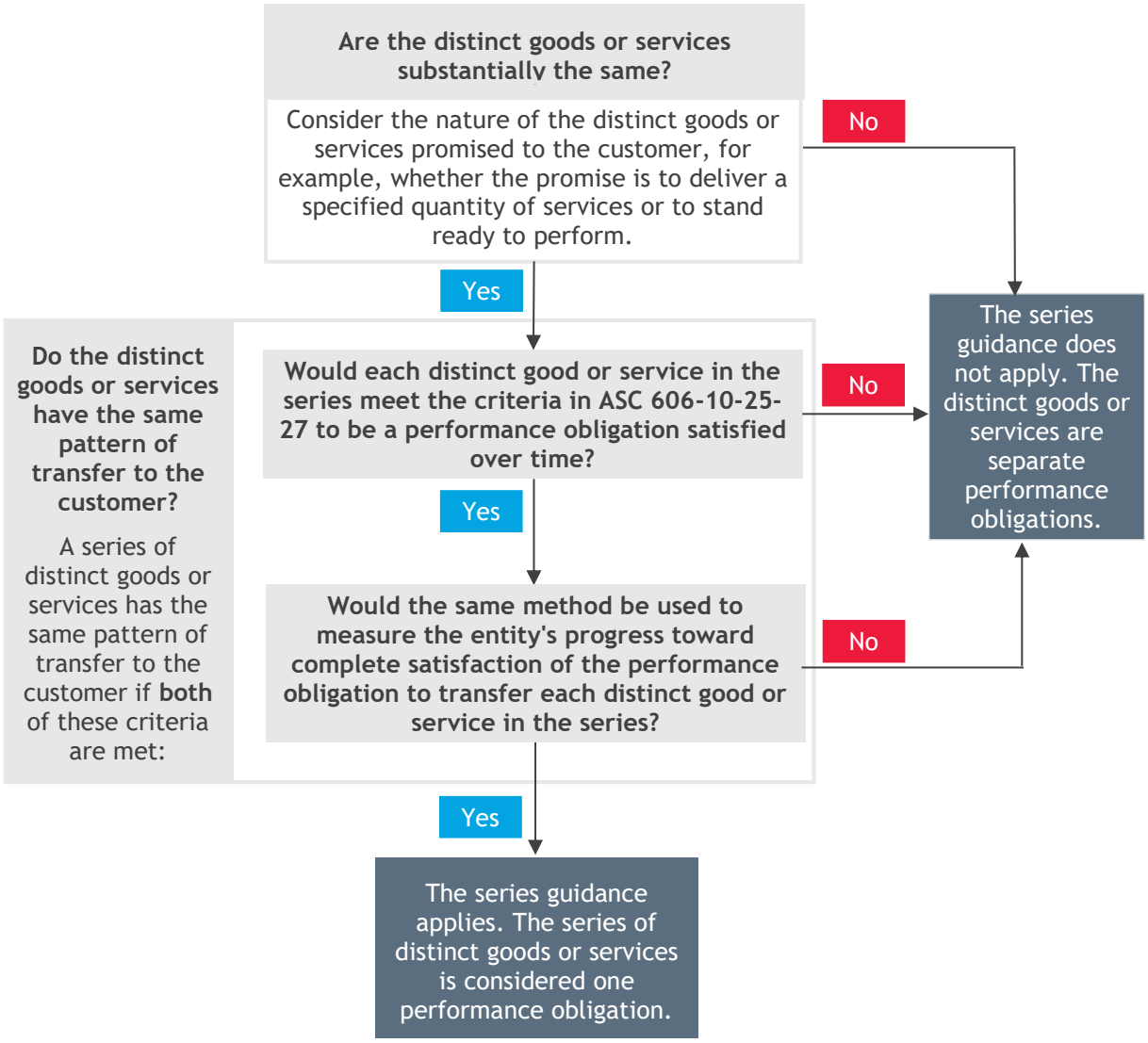
Although the series guidance simplifies the application of ASC 606 in many situations, it is not a practical expedient that software entities have a choice of applying. Rather, it is mandatory if the criteria for its application are met. Therefore, software entities need to consider whether the series guidance applies, in particular to repetitive service contracts. Common examples where series guidance may apply include SaaS and PCS.

BDO INSIGHT 12: EFFECT OF THE SERIES GUIDANCE ON OTHER ASPECTS OF ASC 606

The accounting treatment may vary in three primary areas when a promise is a single performance obligation comprised of a series of distinct goods or services rather than a single performance obligation comprised of goods or services that are not distinct from each other:

- **Changes in transaction price**: ASC 606 requirements apply differently in some cases to a single performance obligation comprised of non-distinct goods or services compared to a single performance obligation resulting from applying the series guidance.
- **Allocation of variable consideration**: The amount of the variable consideration that is recognized for each reporting period could be different, as the standard requires a software entity to allocate variable consideration solely to one or more distinct goods or services even if those goods or services form a single performance obligation.
- **Contract modifications**: If the remaining undelivered goods or services are distinct (even if part of a single performance obligation under the series guidance), a software entity accounts for the modified contract on a prospective basis, whereas if the remaining goods or services are not distinct from those already provided, a software entity makes a cumulative effect adjustment for the effects of the modification.

The following diagram gives an overview of the guidance for determining whether the series guidance applies:



See BDO's Blueprint, **Revenue Recognition Under ASC 606**, for discussion on whether a performance obligation is satisfied over time and measurement of an entity's progress towards satisfying its performance obligation.

TRG DISCUSSIONS: SERIES GUIDANCE — DISTINCT TIME INCREMENTS OR DISTINCT UNITS OF GOODS OR SERVICES

In evaluating the applicability of the series guidance, a software entity must first determine the nature of its promise in providing the goods or services to the customer. The nature of a software entity's promise may be either to deliver a specified quantity of services or to stand ready to perform. This evaluation requires judgment.

The FASB's and IASB's Joint Transition Resource Group (TRG) for Revenue Recognition considered the following when discussing whether a series of distinct goods or services is comprised of multiple distinct time increments or multiple distinct units of goods or services:

Is each time increment distinct?

- If the nature of the software entity's promise is the act of standing ready or providing a single service for a period of time (that is, because there is an unspecified quantity delivered), the evaluation would likely focus on whether each time increment, rather than the underlying activities, are distinct and substantially the same.
  - For example, ASC 606-10-55-157B through 55-157E illustrate a hotel management contract where the nature of the hotel manager's promise is to provide a daily management service, and not a specified quantity of services, and that could include management of the hotel employees, accounting services, training, and procurement. While the underlying activities could significantly vary within a day and from day to day, that would not be relevant to the evaluation of the nature of the promise. In that example, the hotel management performance obligation is a series of distinct days of service.

Is each unit of output distinct?

- If the nature of the promise is the delivery of a specified quantity of a service, then the evaluation considers whether each service is distinct and substantially the same.
  - ASC 606-10-55-160 illustrates an annual contract to provide monthly payroll processing services that are considered a series. In that contract, the nature of the promise is to deliver 12 distinct instances of the service, rather than a promise to stand ready to perform an undefined number of tasks.

BDO INSIGHT 13: APPLICATION OF SERIES GUIDANCE TO SAAS

In practice we observe that the series guidance generally applies to a SaaS that is a stand-ready obligation because:

- Each increment of SaaS (for example, each hour or day) is distinct and satisfied over time
- A software entity uses the same method to measure its progress toward satisfaction of each distinct increment of SaaS

Determining whether a SaaS subscription is a stand-ready obligation requires the application of professional judgment based on the facts and circumstances, as further discussed by the TRG below.

TRG DISCUSSIONS: SERIES GUIDANCE — STAND-READY OBLIGATIONS — UNSPECIFIED SOFTWARE UPGRADES

ASC 606-10-25-18 provides examples of promised goods or services, one of which is a service of standing ready to provide goods or services or of making goods or services available for a customer to use as and when the customer decides. If the goods or services provided meet the criteria in ASC 606-10-25-27 to be performance obligations that are satisfied over time and have the same pattern of transfer, then a stand-ready obligation generally meets the definition of a series and is accounted for as a single performance obligation. However, determining whether the promise to the customer is a stand-ready obligation requires judgment.

The TRG addressed this question by considering certain types of promises (Types A-D) and discussing whether the nature of the good or service underlying each promise is the act of "standing ready" or the actual delivery of the underlying goods or services that the software entity stands ready to provide to the customer.

Type A: Obligations in which the delivery of the good, service, or IP underlying the obligation is within the control of a software entity, but the software entity must still further develop its good, service, or IP. One example is a software entity's promise to transfer unspecified software upgrades at the software entity's discretion.

Type B: Obligations in which the delivery of the underlying good or service is outside the control of the software entity and the customer. For example, a software entity's promise to process transactions when originated by the customer's customers is outside both parties' control.

Type C: Obligations in which the delivery of the underlying good or service is within the control of the customer. One example is a software entity's promise to provide periodic maintenance, when-and-if needed, on a customer's equipment after a specified amount of usage by the customer.

Type D: Making a good or service available to the customer continuously, such as unlimited access to a SaaS platform.

In arrangements such as in Type B, C, or D in which a software entity provides uncertain goods or services, the software entity provides a service to the customer in "standing ready" to perform. For example, while the software entity that sells PCS with a software license knows it will provide maintenance services, it does not know the number of service calls the customer will make or the specific bugs it will be required to fix.

Promises such as those in Type A (for example, to provide when-and-if available updates or upgrades in software) may require additional analysis to determine the nature of the promise because the software entity itself, rather than the customer or external events, might unilaterally control when updates or upgrades become available for transfer to the customer. However, the nature of the software entity's promise in Type A arrangements may be similar to those in Types B through D arrangements because a software entity often will not be able to predetermine the timing, number, or specific functionality or features of major IP improvements that will be completed in the future and made available for transfer to a customer. For example, a software entity might have no major update or upgrade available for release during a contract period, especially if the contract period is relatively short (for example, one year). In that example, the software entity's performance on a Type A arrangement such as a when-and-if available upgrade right, similar to those obligations in Types B through D arrangements, might be dependent upon events or circumstances that are largely outside the software entity's control.

Additionally, similar to the obligations in Types B through D arrangements, a Type A promise to unspecified when-and-if available upgrades is also often about the customer obtaining assurance that it will have access to future improvements to the product it has obtained thus providing the customer with a guarantee against obsolescence or defects in that product. In the software license example, this guarantee provides a benefit to the customer by protecting the customer's investment in the software (which may include, for example, an expensive implementation that would not be recovered economically if the customer had to implement a new software in the near future) or the customer's related business interests (for example, a customer that embeds the software in its

own products might want assurance that it will have access to upgrades of the software so that its products remain competitive in the marketplace). Absent the right to unspecified upgrades, the software entity might charge the customer exorbitant fees in a separate negotiation for the next version of the software or might enter an exclusive arrangement with another customer that restricts the customer's ability to obtain the upgrades.

In determining the nature of its promise to a customer in a Type A arrangement, a software entity must evaluate whether it has promised one or more specified upgrades to a customer, even if the contract refers only to unspecified upgrades that will be transferred to the customer only when and if they become available. A promise to deliver a specified upgrade is accounted for in the same manner as any other specifically promised good or service. For example, a promise to deliver a specified upgrade to a software license is evaluated in the same manner as any other software license.

BDO INSIGHT 14: SAAS WITH USAGE-BASED PRICING

In practice, SaaS is generally considered a series of distinct time increments because the nature of a software entity's promise in providing SaaS is to stand ready to perform as the customer requests. This usually results in:

- Ratable recognition of any fixed consideration in revenue
- Recognition of any variable consideration in the period to which it relates because of the application of the variable consideration allocation exception in ASC 606-10-32-39 through 32-40.

However, software entities must not default to that revenue recognition pattern for SaaS with usage-based pricing. Rather, a software entity must assess the nature of their promise to determine whether it has promised to deliver a specified volume of service (that is, a series of distinct service or usage) or stand ready to perform as the customer requests (that is, a series of distinct time increments).

To assess the nature of its promise, a software entity may consider the following factors:

- Whether the customer benefits from the right to access the SaaS platform in the amount and at the time as needed. In other words, whether the software entity stands ready to perform throughout the contract duration.
- Whether the software entity can predetermine the timing and volume of transactions, and its performance is dependent upon events or circumstances that are outside its control (for example, because the customer controls when and how much to use the service).
- Whether the software entity stands ready to process unlimited transactions or whether the software entity's obligation to the customer diminishes as the customer uses the services.

See Examples 9 and 9A for illustrative fact patterns.

BDO INSIGHT 15: SAAS WITH ARTIFICIAL INTELLIGENCE

Many software entities now incorporate artificial intelligence (AI) into their SaaS offerings and have shifted their pricing strategies from seat-based pricing models to outcome- or usage-based pricing models. Questions have arisen about whether a software entity can recognize outcome- or usage-based revenue as the usage occurs or must estimate and recognize the variable revenue over the contract term. While AI-enabled SaaS offerings may be complex to analyze because of their uniqueness or novelty, the key revenue recognition considerations remain the same as for any other SaaS offering:

- Is the nature of the software entity's promise to deliver a specified volume of service or to stand ready to perform as the customer requests? (This is the most important aspect of the analysis and requires the application of professional judgment based on the facts and circumstances. See BDO Insight 14).
- Does the promise meet the definition of a series, for example, a series of distinct services or transactions or a series of distinct time increments?
- If the performance obligation meets the criteria to be recognized over time, is a time- or usage-based method most appropriate to measure progress?

The timing and pattern of revenue recognition, including for variable usage-based consideration, is affected by the nature of the software entity's performance obligation and the structure of fees. Examples 9 and 9A illustrate whether a software entity recognizes variable usage-based fees in revenue as transactions occur or estimates the fees and recognizes them over the contract term. See BDO Insight 16 and Chapter 4 of BDO's Blueprint, **Revenue Recognition Under ASC 606**, for guidance related to variable consideration.

BDO INSIGHT 16: LIMITED EXCEPTIONS FROM ESTIMATING VARIABLE CONSIDERATION

While ASC 606 generally requires software entities to estimate all variable consideration, there are a few exceptions:

- A software entity does not estimate variable consideration in the form of sales- or usage-based royalties received in exchange for licenses of IP (in accordance with ASC 606-10-55-64 through 55-65B). That exception is explicitly limited to licenses of IP and cannot be applied to sales- or usage-based royalties received in exchange for providing other goods or services, including SaaS.
- A software entity does not estimate variable consideration if it applies the as-invoiced practical expedient to recognize revenue in accordance with ASC 606-10-55-18.
- A software entity does not estimate variable consideration if the variable consideration allocation exception in ASC 606-10-32-39 through 32-40 applies as follows:
  - A performance obligation will be satisfied in future, and the variable consideration is allocated entirely to that performance obligation.
  - A series of distinct services is determined to be a series of distinct time increments, and the variable consideration relates to the distinct services provided in the period when it is earned.

Reaching a conclusion on whether any of the exceptions from estimating variable consideration applies requires the application of professional judgment based on the facts and circumstances. See Chapter 4 of BDO's Blueprint, **Revenue Recognition Under ASC 606**, for details.

EXAMPLE 9: SAAS WITH USAGE-BASED PRICING

FACTS

- A software entity enters a contract to give a customer access to its SaaS platform for two years.
- The customer will have the ability to process 100,000 transactions each year through the SaaS platform.
- The customer must make annual payments of $120,000.
- If needed, the customer can process transactions beyond the annual volume of 100,000 transactions but must pay a higher rate for the incremental transactions.
- The software entity does not expect the customer to exceed 100,000 transactions per year.

CONCLUSION

The software entity's promise is to stand ready to provide continuous access to the SaaS platform over the two-year term, rather than to deliver a fixed quantity of transaction-processing services.

ANALYSIS

The software entity evaluates the nature of its promise to determine whether it has an obligation to provide:

- A specified amount of services (that is, 100,000 annual transactions processed through SaaS) where each transaction is distinct and substantially the same; or
- A stand-ready obligation, comprised of distinct increments of time, to provide SaaS to process all transactions if and when required by the customer.

The software entity determines that it has a stand-ready obligation to provide SaaS for the entire contractual period, regardless of the amount of services specified in the contract or expected to be sold. In making that determination, the software entity considers the following:

- The customer benefits from the right to access the SaaS platform and process transactions in the amount and at the time as needed. The software entity stands ready to perform throughout the contract duration.
- The software entity is unable to predetermine the timing and volume of transactions, and its performance is dependent upon events or circumstances that are outside its control.
- The software entity stands ready to process unlimited transactions. While transactions that exceed the annual volume are billed at a premium price, the customer does not have to separately contract for the right to transact those volumes. The software entity's obligation to the customer does not diminish as transactions are processed. Selling a specified volume at a fixed price and overages on a variable basis is a pricing strategy rather than determinative of the software entity's obligations to the customer.

EXAMPLE 9A: AI FUNCTIONALITY IS A SPECIFIED QUANTITY

FACTS

- A software entity enters a contract to give a customer access to its AI-enabled SaaS platform for two years.
- The customer uses the SaaS platform for process automation. The customer can use AI features to perform other tasks, for example, to extract data from and perform analytical procedures over invoices processed through the platform. The SaaS platform and AI functionality are separate performance obligations.

Regarding the AI performance obligation:

- The customer can perform 10 million AI tasks each year through the SaaS platform. Each AI task processed in a year reduces the remaining number of AI tasks that the customer can process that year.
- If needed, the customer can process AI tasks beyond the annual volume of 10 million but must execute an addendum to buy more AI tasks before being able to process those tasks.
- If the customer processes less than 10 million AI tasks in a year, the unused amount rolls over to the next year.
- The customer must make an upfront payment of $900,000 at the beginning of each year, that is, $0.09 per AI task.
- The software entity offers additional bundles of AI tasks that customers may buy at any point during an active SaaS subscription period.

CONCLUSION

The software entity's promise is to provide a specified quantity of AI tasks rather than continuous access to the AI functionality over the two-year term.

ANALYSIS

The software entity evaluates the nature of its promise to determine whether it has an obligation to provide:

- A specified amount of services (that is, 10 million AI tasks processed through SaaS in the first year) where each service of processing an AI task is distinct and substantially the same; or
- A stand-ready obligation, comprised of distinct increments of time, to provide AI tasks if and when required by the customer.

The software entity determines that it has promised to provide a specified amount of AI tasks where each task is distinct and substantially the same based on the following:

- Each AI task performed reduces the customer's remaining rights to process AI tasks in a year. Therefore, the software entity's obligation to the customer diminishes as the customer processes those tasks.
- The customer's right to the AI functionality ends once it reaches the annual volume of 10 million. To process AI tasks beyond the annual volume, the customer must make a separate purchasing decision through execution of an addendum, a separate contract, for the right to process the additional tasks.
- Unused volume rolls over to a future period.
- The software entity sells the AI functionality separately to its customers at any point during an active SaaS subscription term.

MATERIAL RIGHT

A software entity may grant its customer an option that allows the customer to purchase additional goods or services as part of or in conjunction with a contract. For example, a software entity may sell a one-year software license to customer and give the customer the option for annual renewals at a fixed price.

A customer option for future goods or services represents a material right to the customer and hence a performance obligation in the contract if the option gives a material right to the customer that it would not receive without entering that contract. Therefore, a discount that is incremental to the range of discounts typically given for those goods or services to that class of customer in that geographical area or market represents a material right.

Conversely, if the option allows the customer to acquire an additional good or service at a price that would reflect the standalone selling price for that good or service, that option does not provide the customer with a material right, even if the option can be exercised only by entering a previous contract.

A material right is a performance obligation identified in Step 2 of the five-step revenue recognition model and therefore, the subsequent steps related to allocation of transaction price and recognition of revenue must be applied to a material right.

If an option provides a material right to the customer, the customer in effect pays the software entity in advance for future goods or services. Therefore, the software entity must recognize the related revenue when either of the following applies:

- Those future goods or services are transferred.
- The option expires unexercised.

If an option does not provide a material right to the customer, the software entity has made a marketing offer that is accounted for in accordance with ASC 606 only when the customer exercises the option to purchase the additional goods or services.

IMMATERIALITY EXCEPTION IN IDENTIFYING THE PERFORMANCE OBLIGATIONS IS NOT AVAILABLE FOR A MATERIAL RIGHT

A software entity cannot apply the immateriality exception in identifying the performance obligations in Step 2 to a customer option that is determined to represent a material right.

TRG DISCUSSIONS: ASSESSMENT OF WHETHER VARIABLE QUANTITY CONSTITUTES A PURCHASE OPTION OR VARIABLE CONSIDERATION

The TRG considered how a software entity determines whether a contractual right to acquire additional goods or services represents an option to purchase additional goods and services or variable consideration based on a variable quantity. This question might arise, for example, if a software entity grants 500 licenses to use software for a fixed fee of $500,000, with the price for additional user licenses of $800.

TRG members agreed that all facts and circumstances must be considered when analyzing contracts with similar provisions and that this analysis requires judgment. However, they concluded that the first step in distinguishing between optional goods or services and variable consideration for promised goods or services is to identify:

- The nature of the software entity's promise to the customer
- The enforceable rights and obligations of the parties

With an option for additional goods or services, the customer has a present right to choose to purchase additional distinct goods or services (or change the goods and services delivered). Before the customer's exercise of that right, the software entity is not presently obligated to provide those goods or services, and the customer is not obligated to pay for those goods or services.

In contrast, treatment as variable consideration requires the software entity and the customer to have previously executed a contract that requires the software entity to transfer the promised good or service and the customer to pay for that promised good or service. The future events that trigger payment of additional consideration to the software entity occur after (or as) control of the goods or services have (or are) transferred. When a contract includes variable consideration based on a customer's actions, those actions do not oblige the software entity to provide additional distinct goods or services (or change the goods or services transferred), but rather, resolve the uncertainty from the amount of variable consideration that the customer is obligated to pay.

BDO INSIGHT 17: SOFTWARE LICENSE OR SAAS WITH USAGE-BASED PRICING

A software entity may price its software license or SaaS arrangement based on a customer's usage. Determining whether usage-based pricing for a software license or a SaaS arrangement constitutes customer purchase options or variable consideration requires professional judgment, based on the facts and circumstances.

In practice we generally observe that usage-based pricing represents variable consideration rather than a customer option for additional purchases in an arrangement for a software license or a SaaS arrangement that is a stand-ready obligation to provide a customer continuous access to a hosted platform. See above for discussion of stand-ready obligations and Examples 9 and 9A for further discussion.

Customer Option for Cloud Conversion

A software entity's contract with a customer to grant a software license may provide the customer the right to convert the software license to a SaaS subscription for the same software. That is, at contract inception the contract explicitly or implicitly includes a customer option for cloud conversion. Accounting for the conversion option depends on the specific facts and circumstances.

Cloud Conversion Results in Rights to SaaS and Continued Right to Software License

If the customer continues to have a right to the software license after exercising its conversion option, then the contract provides the customer an option to purchase an additional service (SaaS access), and the software entity must determine whether the customer option is a material right. If the customer is not required to pay for access to the SaaS platform or the payment is not commensurate with the standalone selling price of the access, then the conversion option is a material right. If, however, the payment for access to the SaaS platform is commensurate with its standalone selling price, then the conversion option is a marketing offer and not a separate performance obligation.

Cloud Conversion Results in Rights to SaaS and Forfeiture of Right to Software License

Diversity exists in practice in accounting for a cloud conversion option when the customer forfeits its software license upon conversion to a SaaS subscription. While some apply the above discussed material right approach to determine whether the customer option is a material right that results in allocating a portion of the consideration to the conversion right, others view the forfeiture of the software license as a return of that license and apply the right to return guidance to account for the conversion option. Under the right of return approach, a software entity estimates an amount of payment for the software license that must be attributed to the SaaS subscription because the software entity expects the customer to return the license. The software entity defers the estimated amount ("refund") and recognizes it in revenue as SaaS revenue after converting the license to a SaaS subscription. The Emerging Issues Task Force (EITF)[2] considered this issue and discussed alternative views but did not reach a conclusion.

Note that if the payment for a SaaS subscription is commensurate with the standalone selling price, then the conversion option is effectively a marketing offer and not a separate performance obligation because:

Under the material right approach →	The option is not a material right because the customer does not receive a discount on the SaaS subscription.
Under the right of return approach →	There is no theoretical refund of the customer's payment for the software license upon conversion to a SaaS subscription because the customer pays the standalone selling price for that subscription.

BDO INSIGHT 18: ACCOUNTING FOR CLOUD CONVERSION RIGHTS

Accounting for cloud conversion rights requires the application of professional judgment based on the facts and circumstances.

A software entity that grants a cloud conversion option that results in forfeiture of the customers' software licenses must make an accounting policy election to account for the conversion option as a material right or a right to return the software license. We believe this election must be consistently applied to similar contracts and appropriately disclosed.

[2] EITF 19-B, Issue 2

A customer option for cloud conversion may be explicit or implied by the software entity's customary business practices at contract inception. For example, consider a software entity that is actively encouraging customers to convert software licenses to its SaaS offering at discounted prices. A customer that contracts for a software license may expect at contract inception that it has an option to convert its software license to a SaaS subscription even though the contract may not explicitly provide that right.

If a cloud conversion option is not explicitly or implicitly provided at contract inception, and a software entity and its customer subsequently agree to convert a license to a SaaS subscription, the original license contract is modified and the software entity must apply the contract modification guidance.

Remix Rights

A software entity may grant multiple licenses of software to a customer (for example a license to Product A with 10 users and a license to Product B with 20 users) and allow the customer to change the mix in which it uses each license (referred to as a "remix right"). A remix right does not provide the customer with a right to purchase additional goods or services in the future. Rather, it allows the customer to change the composition of the rights that it has already received (and the rights that a software entity has already granted) under the existing contract. Therefore, remix rights are typically not customer options or material rights.

Early Renewal Rights

Software entities often offer noncancellable contracts that provide the customer with the option to renew the contract before it expires. For example, a software entity may sell a software license with a one-year term and provide the customer with a right to renew the contract for another year at any time within the last 90 days of the original license term. Revenue from license renewal cannot be recognized before the renewal period has begun and the customer can use and benefit from the renewed license.

Additionally, when a customer exercises an early renewal option, a software entity must consider whether the scope or price stipulated in the original contract has been amended, resulting in a contract modification. For example, when a customer exercises an early renewal option for a software license in the original contract, it may also negotiate with the software entity to reduce the term of another software license included in the original contract or purchase additional goods or services. In that case, a software entity applies the contract modification guidance to determine the appropriate accounting. See the following related discussion on blend-and-extend modifications.

Blend-and-Extend Modifications

A software entity may amend a contract with a customer to extend the period of the contract in exchange for a new blended price for the remaining term (commonly referred to as a blend-and-extend modification). For example, a two-year software license priced at $10,000 per year may be modified at the end of one year to extend the term for two additional years at $7,500 per remaining year.

To account for a blend-and extend modification, a software entity applies the contract modification guidance that requires an analysis of:

- Whether a performance obligation in the original contract has been satisfied before the modification date
- Whether the modification adds distinct goods or services, that is, separate performance obligations
- Whether the prices of the distinct goods or services are at their standalone selling prices

If the remaining undelivered goods or services are distinct, a software entity accounts for the modified contract prospectively, whereas if the remaining goods or services are not distinct from those already provided, a software entity recognizes a cumulative effect adjustment for the effects of the modification. See BDO's Blueprint, **Revenue Recognition Under ASC 606**, for a detailed discussion of the contract modification guidance.

Customers' Unexercised Rights: Breakage

A customer's nonrefundable prepayment to a software entity gives the customer a right to receive a good or service in the future and obliges the software entity to stand ready to transfer a good or service. Upon receipt of a prepayment from a customer, a software entity must recognize a contract liability in the amount of the prepayment for its performance obligation to transfer, or to stand ready to transfer, goods or services in the future.

A software entity must derecognize that contract liability and recognize revenue when it transfers those goods or services and therefore satisfies its performance obligation. However, customers might not always exercise all their contractual rights. Those unexercised rights are often referred to as "breakage." Common examples include declining to exercise a renewal option that is accounted for as a material right and failing to meet usage subject to a guaranteed minimum price.

A software entity accounts for any resulting breakage:

If a software entity expects to be entitled to a breakage amount in a contract liability	→	Then it recognizes the expected breakage amount as revenue in proportion to the pattern of rights exercised by the customer
If a software entity does **not** expect to be entitled to a breakage amount	→	Then it recognizes the expected breakage amount as revenue when the likelihood of the customer exercising its remaining rights becomes remote

CONSIDER VARIABLE CONSIDERATION CONSTRAINT TO DETERMINE BREAKAGE

To determine whether it expects to be entitled to a breakage amount, a software entity must consider the guidance in Step 3 on constraining estimates of variable consideration. Specifically, a software entity must recognize breakage revenue only to the extent that it is probable that there will not be a significant reversal of cumulative revenue recognized when the uncertainty related to the breakage is subsequently resolved.

A software entity must not recognize breakage revenue for any consideration received that must be remitted to another party, for example, to a government entity in accordance with applicable unclaimed property laws. Instead, the software entity reports a liability due to the relevant third party, rather than a contract liability or deferred revenue.

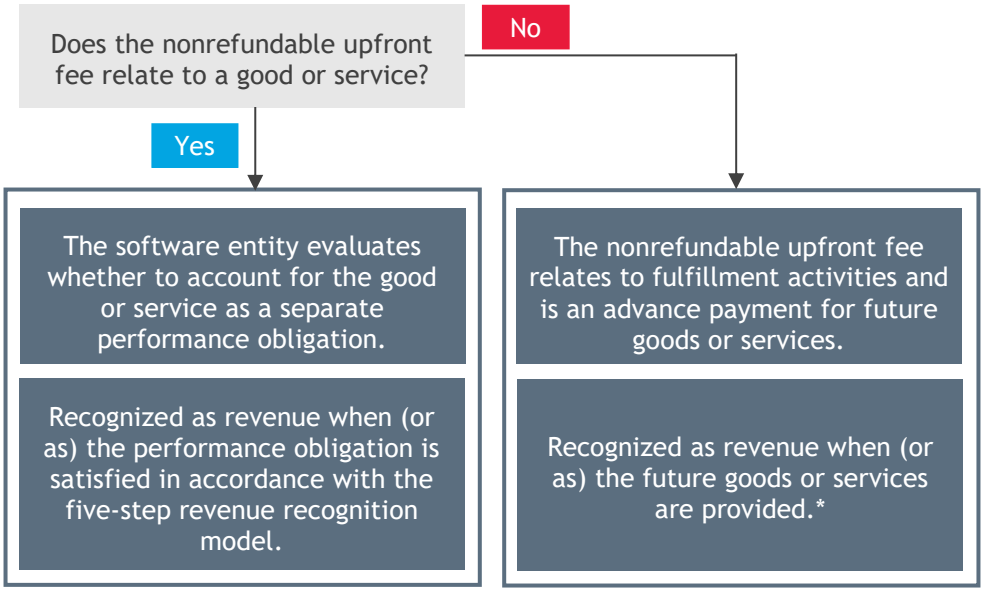
BDO INSIGHT 19: ACCOUNTING FOR BREAKAGE LIABILITIES MAY REQUIRE LEGAL ANALYSIS

Determining whether breakage liabilities are subject to unclaimed property laws is a legal question and may require judgment. Because the laws often vary by state, a software entity could have multiple breakage liabilities that are accounted for differently. As such, software entities may need to discuss their fact patterns with unclaimed property specialists. Accounting for breakage liabilities requires the application of professional judgment based on the facts and circumstances.

Nonrefundable Upfront Fees

A software entity may charge a customer a nonrefundable upfront fee at or near contract inception such as a set-up fee in SaaS contracts.

The following diagram illustrates the key accounting considerations for nonrefundable upfront fees:



*Revenue recognition period generally extends beyond the initial contract term if the software entity grants the customer a renewal option that is determined to be a material right.

When a contract includes a nonrefundable upfront fee, a software entity assesses whether that fee relates to the transfer of a promised good or service and accounts for the fee as follows:

- If the nonrefundable upfront fee relates to a promised good or service, the software entity evaluates whether to account for the good or service as a separate performance obligation.
- If the nonrefundable upfront fee does not result in the transfer of a promised good or service to the customer but rather relates to fulfillment activities, the fee is an advance payment for future goods or services and therefore is recognized as revenue when those future goods or services are provided.
  - The revenue recognition period generally extends beyond the initial contract term if the software entity grants the customer renewal options for the contract that are determined to be material rights.

In many cases, even though a nonrefundable upfront fee may relate to an activity that the software entity must undertake at or near contract inception to fulfill the contract, that activity may not result in the transfer of a promised good or service to the customer but rather may be a fulfillment activity. See above for a discussion on accounting for fulfillment activities.

TRG DISCUSSIONS: ANALYZING CUSTOMER LIFE TO DETERMINE WHETHER A NONREFUNDABLE UPFRONT FEE PROVIDES A MATERIAL RIGHT TO THE CUSTOMER

The TRG discussed whether a one-time nonrefundable upfront fee (for example, an initial set-up fee in a SaaS arrangement) gives the customer a material right. At that meeting, the TRG members stated that a software entity's average customer life may indicate whether an upfront fee provides the customer a material right. For example, a customer life that extends beyond a one-month contractual period may indicate the upfront fee incentivizes the customer to continue purchasing access to the SaaS platform because the customer will not be charged an additional initial set up fee in subsequent months, as opposed to incurring a similar upfront fee with a new software provider.

EXAMPLE 10 (ADAPTED FROM 606-10-55-358 THROUGH 55-360): NONREFUNDABLE UPFRONT FEE — MATERIAL RIGHT VERSUS ADVANCE PAYMENT FOR SERVICES TO BE PROVIDED IN FUTURE

FACTS

- A software entity enters a contract with a customer to provide access to a SaaS platform for one year.
- The contract includes standard terms and conditions.
- The contract requires the customer to pay a nominal nonrefundable upfront fee to allow the software entity to recover the costs it incurs in setting up the customer in the software entity's systems and processes.
- The customer can renew the contract each year at the standalone selling price of the services without any additional fee after the initial set up.

CONCLUSION

The renewal option does not provide a material right, and the nonrefundable upfront fee is an advance payment recognized in revenue over the SaaS service period.

ANALYSIS

The software entity observes that:

- Its initial set-up activities do not transfer a good or service to the customer and therefore do not give rise to a performance obligation.
- Although a customer can renew the contract without paying another upfront fee, the amount of the fee is not significant to the price of the services during the renewal period.

Therefore, the software entity concludes that the renewal option is priced at the standalone selling price of the services and thus does not provide a material right to the customer.

The software entity determines that the upfront fee is essentially an advance payment for future services that it will provide to the customer. Therefore, it includes the nonrefundable upfront fee in the transaction price in Step 3 and recognizes it as revenue as the SaaS service is provided.

EXAMPLE 11: NONREFUNDABLE UPFRONT FEE — MATERIAL RIGHT

FACTS

- A software entity contracts with a customer to provide access to a SaaS platform for one year.
- The contract includes standard terms and conditions.
- The contract requires the customer to pay a significant nonrefundable upfront fee in addition to the amounts payable for the initial one year of service.
- The customer can renew the contract for four additional years at the same price as the initial year of service but without paying another nonrefundable upfront fee.

CONCLUSION

The upfront fee provides a material right to the customer that represents a separate performance obligation.

ANALYSIS

The software entity observes that:

- The amount of the upfront fee is significant to the price of the services during the renewal period.
- The renewal options are priced below the standalone selling price of the services because the customer can renew the contract without paying an additional nonrefundable upfront fee.
- The significance of the upfront fee and amounts paid for the initial one year compared to the price payable for renewals may economically compel the customer to renew the services. In other words, the upfront fee could influence the customer's purchasing decision regarding renewing the services.

Based on the above observations, the software entity concludes that the upfront fee provides a material right(s) (separate performance obligations) to the customer.

The software entity allocates a portion of the upfront fee and amounts paid for the initial one year to the material right(s) and recognizes it in revenue as the option(s) is exercised or lapses unexercised.

PRINCIPAL VERSUS AGENT CONSIDERATIONS

Promised goods and services in a contract may include:

- Resale of rights to goods or services purchased by a software entity to its customer. For example, a software reseller might purchase rights to software licenses from a software licensing entity and resell those rights to end customers.
- Providing a service of arranging for another party to transfer goods or services to a customer. For example, a software reseller might arrange for a software licensing entity to provide PCS to the reseller's customers who have purchased licenses to the software licensing entity's IP from the reseller.

If a third party is involved in providing goods or services to the customer, then a software entity must determine whether it is the principal or agent in that arrangement for revenue recognition purposes.

A software entity is a principal and therefore recognizes revenue on a gross basis if it controls a good or service before transferring it to the customer. A software entity is an agent and therefore recognizes revenue on a net basis if it arranges for a good or service provided by another entity to be delivered to the end customer. The standard includes indicators and examples to assist with the analysis. See BDO's Blueprint, Revenue Recognition Under ASC 606, for more guidance on performing a principal versus agent assessment.

DETERMINING THE NATURE OF A SOFTWARE ENTITY'S PROMISE IN A THREE-PARTY ARRANGEMENT

When a third party is involved in providing goods or services to a customer, a software entity must determine whether the nature of its promise is to:

- Provide the specified goods or services itself; that is, the software entity is a principal
- Arrange for those goods or services to be provided by the third party; that is, the software entity is an agent

To determine the nature of its promise, a software entity must:

- Identify the specified goods or services provided to the customer (for example, a right to a good or service provided by a third party)
- Assess whether it controls each specified good or service before it transfers the good or service to the customer

BDO INSIGHT 20: INTERRELATIONSHIP BETWEEN IDENTIFYING THE CUSTOMER (STEP 1) AND PERFORMANCE OBLIGATIONS (STEP 2)

The first step in applying the principal versus agent analysis is appropriately identifying the good or service that a software entity has promised to transfer to the customer (the specified good or service). In some arrangements, identification of the good or service that a software entity has promised to transfer may inform the conclusion about which party is the software entity's customer.

For example, consider a software entity that has developed an app to connect car drivers to end customers to receive car rides. The end customers download the app at no cost to connect with car drivers and make payments to the technology platform entity. If the software entity determines that the nature of its promise is to connect the driver to the end customer (rather than to provide ride services to the end customer), then the software entity would conclude that its customer is the driver (not the end customer). Reaching a conclusion about identifying the obligation(s) and the customers requires the application of professional judgment based on the facts and circumstances.

BDO INSIGHT 21: USING THE PRINCIPAL VERSUS AGENT GUIDANCE TO IDENTIFY THE CUSTOMER

The guidance in ASC 606 on performing a principal versus agent assessment is written from the viewpoint of an intermediary software entity that obtains a good or service from a third party and subsequently sells that good or service to a customer. In that scenario, the software entity may conclude either of the following:

- It is the principal in which case the cost of the good or service procured from the third party will be reported as an operating expense (that is, cost of goods sold or cost of services).
- It is an agent in which case the cost of the good or service will be reported net in revenue.

However, in many cases the software entity performing the principal versus agent assessment may be the party that provides the good or service to a third party, who then transfers the good or service to an end customer. In that case, the software entity is considered a principal in the transaction because it controls the good or service before transferring it to the interim buyer. In that fact pattern, the assessment focuses on whether the interim buyer or the end customer is the software entity's customer. To identify its customer, the software entity must assess whether its immediate buyer is a principal or an agent in the sale to the end customer.

- If the software entity concludes that the interim buyer is the principal in the sale to the end customer, then its customer is the interim buyer, and the software entity's revenue is the amount charged to the interim buyer. The software entity's income statement does not reflect any subsequent increase or decrease in price the interim buyer charges to the end customer.
- If the software entity concludes that the interim buyer is acting as the software entity's agent and the end customer is the software entity's customer, then the amount charged to the end customer by the interim buyer is recognized as the software entity's revenue, and any difference between that amount and the amount charged to the interim buyer is reflected as an operating expense (that is, cost of goods sold, cost of services, or selling expense). However, when a software entity does not have visibility into the price charged by the interim buyer to the end customer, that unknown amount is excluded from the software entity's revenue (that is, the software entity does not estimate the amount the interim buyer charges to the end customer).

A software entity may have more than one customer in a three-party arrangement. For example, a software entity might contract with a reseller to sell licenses and related PCS to end customers. The software entity controls both the license and PCS before transferring them to the reseller or end customer. Therefore, the software entity is the principal in this transaction. However, it needs to determine whether the reseller or end customer is its customer. It is possible that the reseller is its customer for the license and the end customer is the customer for PCS (presuming the license and PCS are separate performance obligations).

BDO INSIGHT 22: DETERMINING CONTROL OF AN INPUT TO A COMBINED OUTPUT

When a software entity incorporates a good or service it obtains from a third party into a combined good or service it then delivers to a customer, it must control that good or service to integrate it into the combined output.

However, determining whether a software entity controls a good or service obtained from a third party and integrates that good or service into a combined output can be challenging. In a 2021 speech, the SEC staff discussed consultations in which an entity concluded it is a principal because it integrates a good or service received from a third party into its own good or service. The SEC staff indicated that the entity considers:

- The nature of the integration service
- The significance of the integration service
- Whether it controls the third party good or service before performing the integration service

The SEC staff further noted that it may be unclear how an entity could perform a significant integration service if it does not first control a promised good or service received from a third party.

Determining whether a software entity controls a good or service that it integrates into a combined output requires the application of professional judgment based on the facts and circumstances.

BDO INSIGHT 23: RESALE OF SOFTWARE LICENSES AND SAAS

A software entity that resells software licenses or SaaS subscriptions evaluates the control principle and indicators to determine whether it is a principal or an agent in each transaction.

A reseller might control **a software license** before transferring it to a customer if it has an enforceable right to sell the third-party provider's licenses and therefore to direct the third-party software provider to perform. Additional indicators of control of a software license might include:

- The reseller maintains a call center and helpdesk for customer support and is responsible for resolving customer matters.

- The reseller provides refunds when the third-party provider does not accept a customer return.
- The reseller can separately negotiate pricing with both the third-party provider and the customer.

By contrast, a reseller generally does not control **SaaS** before transferring it to the end customer, unless the reseller performs a significant integration, modification, or customization service.

Determining whether a reseller controls a software license or SaaS requires the application of professional judgment based on the facts and circumstances. See Examples 12 through 14 for illustrative examples.

EXAMPLE 12: SOFTWARE LICENSE RESELLER IS AN AGENT

FACTS

- A software entity sells information technology (IT) security software products to its customers. The software entity evaluates technology solutions available from third parties and determines which combination of products will best meet its customers' needs.
- The software entity buys specific software licenses selected by its customer directly from third-party providers and concurrently resells them to the customer.
- The third-party provider transfers a copy of the software directly to the customer. The customer executes an end-user license agreement (EULA) with the third-party provider.
- The software entity does not have a pre-purchased pool of licenses for resale. The software entity has not entered a master reseller agreement with a third-party provider or agreed to buy a minimum volume of licenses.
- The software entity serves as a point of contact to facilitate a customer's contact with the third-party provider but does not provide any support for or updates to the software itself.
- The software entity does not accept returns or issue refunds.
- The software entity charges the customer the third-party software cost plus a fixed percentage markup.
- Each software license is a specified good that is distinct. The software entity does not modify or customize the licensed software, nor does it provide an integration service.

CONCLUSION

The software entity is an agent for the resale of the software licenses.

ANALYSIS

The software entity concludes that it is an agent for the specified licenses because it:

- Does not control the software licenses before they are transferred to its customers and does not direct the third-party provider to perform. The software entity has not entered a master reseller agreement with a third-party provider that provides the software entity with an enforceable right to resell software licenses.
- Does not have primary responsibility for fulfillment. The third-party provider transfers a copy of the software directly to the customer. The customer executes a EULA with the third-party provider. Also, the software entity serves as a point of contact to facilitate a customer's contact with the third-party provider but does not provide any support for or updates on the software itself.
- Does not have inventory risk. The software entity does not have a pre-purchased pool of licenses for resale, nor has it agreed to buy a minimum volume of licenses. The software entity also does not accept returns or issue refunds.
- Has limited pricing discretion. The software entity charges the customer the third-party software cost plus a fixed percentage markup.

EXAMPLE 13: SOFTWARE LICENSE RESELLER IS A PRINCIPAL

FACTS

Assume the same facts in Example 12 except:

- The software entity has entered a master reseller agreement with the third-party provider, which gives the software entity an enforceable right to grant the third-party provider's software licenses to its customers. The agreement also gives the software entity the ability to negotiate pricing with end customers, for example, through volume discounts.
- The software does not require configuration or modification by the third-party provider before the software entity provides the software license.
- The software entity maintains a call center and helpdesk for customer support and is responsible for resolving customer matters.
- The software entity provides refunds when the third-party provider does not accept a customer return.
- The software entity invoices the customer for its services, including the price of the software licenses.

CONCLUSION

The software entity is a principal for the resale of the software licenses.

ANALYSIS

The master reseller agreement gives the software entity the right to direct the third-party provider to issue the software licenses to the customers because it gives the software entity an enforceable right to grant the third-party provider's software license to its customers. The third-party provider has transferred a right to the software entity to sell licenses to its IP, which exists before a customer places an order. Therefore, the software entity concludes that it has an enforceable right to grant licenses before transferring those licenses to the customer. Therefore, it controls the software licenses before they are transferred to its customers and is a principal for those licenses.

While the above analysis is conclusive, the software entity also considers the following that supports the fact that it is acting as the principal:

- The software entity maintains a call center and helpdesk for customer support and is responsible for resolving customer matters.
- The software entity has back-end inventory risk because it provides a refund if a customer return is not accepted by the third-party provider.
- The software entity has pricing discretion.

EXAMPLE 14: SAAS RESELLER IS A PRINCIPAL

FACTS

- A software entity develops and sells access to a SaaS platform to provide background verification services to law enforcement agencies.
- The software entity has a master reseller agreement with a third-party provider, which gives the software entity the right to resell access to the third-party provider's SaaS platform as part of its own SaaS offering. The software entity has designed and coded its software such that it works with the third-party provider's SaaS platform for integrated software functionality.
- The end customers, law enforcement agencies, enter contracts with the software entity. The software entity has full pricing discretion for the integrated solution.
- The software entity is responsible for fulfilling the contract with the end customer, including resolving customer matters.

CONCLUSION

The software entity is the principal in its contracts with its customers.

ANALYSIS

The software entity considers the following and concludes that it is the principal in its contracts with its customers:

- The software entity's SaaS solution combines the software functionalities of the third-party provider's SaaS platform as an input into its own software to create a single integrated service that provides a single output to the customer.
- The master reseller agreement is between the software entity and the third-party provider. The end customers are not parties to any contracts with the third-party provider.
- The master reseller agreement gives the software entity the right to resell access to the third-party provider's SaaS platform to its customers without requiring the third-party provider's approval.
- The customer expects to buy the single integrated functionality from the software entity.
- The software entity enters the contracts with the law enforcement agencies, is responsible for fulfilling the contracts and resolving customer matters and has pricing discretion.

INTERACTION OF STEP 2 WITH IDENTIFYING A CONTRACT AND DETERMINING THE CONTRACT TERM IN STEP 1

A software entity applies ASC 606 to the term of the contract in which the parties to the contract have enforceable rights and obligations. Therefore, to correctly apply the guidance, a software entity must first determine whether a contract with the customer exists and the duration of the contract (or contract term) for accounting purposes in Step 1. See detailed discussion of Step 1 in BDO's Blueprint, Revenue Recognition Under ASC 606.

Free Trials: Does a Contract Exist?

A software entity may offer certain services (for example, access to a SaaS platform) for free for a limited period during which a customer can try the services and decide whether to purchase it for a longer term. During the trial period, the customer can opt out of the free trial at any time or decide to accept the offer and purchase the services for a longer term.

Accounting for the services provided during the free trial period depends on whether and when the customer accepts the software entity's offer and purchases the services for a longer term. A contract with a customer does not exist until the customer accepts the software entity's offer to provide services after the free trial period in exchange for consideration.

If the customer contracts for the services for a longer term during the trial period	→	Then the services provided during the remainder of the free trial period and the term beginning after the end of the free trial period are both considered in determining the performance obligations in that contract. The services provided in the free trial period before the customer accepts the offer are accounted for as sales incentives.
If the customer does not contract for the services for a longer term or executes a contract after the free trial ends	→	Then the services provided during the free trial period are accounted for as sales incentives

Effect of Termination Clauses on Contract Term

Termination clauses affect the contract term in the following manner:

- If each party to the contract has the unilateral enforceable right to terminate a wholly unperformed contract without compensating the other party, then a contract does not exist under ASC 606.
- If either party to the contract has the unilateral enforceable right to terminate the contract only by paying a substantive termination penalty to the other party, then a contract exists under ASC 606. A contract continues to exist during the specified contractual period regardless of each party's unilateral and enforceable right to terminate the contract at any time because enforceable rights and obligations exist throughout the contractual period, evidenced by the fact that compensation (a substantive termination penalty) would be required to terminate the contract.

BDO INSIGHT 24: ANALYZING TERMINATION PENALTIES

ASC 606 does not define the terms "penalty" or "termination penalty," nor does it contain bright lines for determining whether a termination penalty is substantive. In analyzing whether either party must compensate the other to exercise a termination right, a software entity evaluates any amount payable upon termination that is unrelated to the payments due for the goods or services transferred up to the termination date. Therefore, the analysis is not restricted only to payments explicitly characterized as termination penalties. Rather, a software entity considers the substance of the payments made to compensate the other party to terminate a contract before the end of the stated term.

The analysis of termination penalties is not restricted only to cash payments. For example, requiring the customer to return an exclusive software license upon early termination of the contract may represent a substantive termination penalty. Conversely, requiring the customer to return a nonexclusive software license upon early termination of the license contract will likely not represent a substantive termination penalty.

In assessing whether a penalty is substantive, a software entity evaluates whether the penalty economically compels the customer **not** to terminate the contract. This analysis is both quantitative and qualitative in nature. A software entity may consider these factors:

- What is the reason for the termination penalty?
- How important are the software entity's goods or services to the customer's operations or business objectives?
- Does the software entity have a history of enforcing similar termination penalties?
- Have customers historically chosen to terminate similar contracts despite the penalty?
- Can the customer continue to benefit from goods or services already received after the contract is terminated?
- Are there readily available alternative vendors or substitute products or services? (Note: Costs that the customer would incur to switch vendors are only relevant if imposed by the software entity making the assessment.)
- Is the penalty quantitatively significant relative to the remaining contractual payments?

This list is not exhaustive, and no single factor is determinative. All relevant factors are be evaluated holistically. Determining what constitutes a termination penalty and whether the termination penalty is substantive requires the application of professional judgment based on the facts and circumstances.

A software entity must evaluate substantive termination provisions (termination rights, termination penalties, or other payments) in a contract with a customer and its customary business practices to determine whether the duration for which enforceable rights and obligations exist is shorter or longer than the contractually stated term. The termination clauses may affect the contract term for accounting purposes, which might impact the assessment of the performance obligations and revenue recognition.

Examples 15 and 16 illustrate this concept.

EXAMPLE 15: CANCELLABLE CONTRACT FOR SOFTWARE LICENSE WITH SUBSTANTIVE TERMINATION PENALTY

FACTS

A software entity enters a contract to grant a software license to a customer.

Following are the key terms of the contract:

- The contractually specified term is four years.
- The customer must pay $4 million, payable in four equal installments at the beginning of each year of the term.
- The customer has the unilateral right to terminate the contract for convenience before the beginning of each year of the term by paying a termination penalty equal to 25% of the remaining fees, as follows:

	YEAR 1	YEAR 2	YEAR 3	YEAR 4
License fee	$ 1,000,000	$ 1,000,000	$ 1,000,000	$ 1,000,000
Termination penalty	-	$ 750,000	$ 500,000	$ 250,000

- Assume that the nature of the software entity's promise is to provide the customer with a right to use the software entity's IP, that is, the license is a functional IP for which revenue is recognized at a point in time.

CONCLUSION

Because the termination penalty is substantive, enforceable rights and obligations exist for the full four-year term.

ANALYSIS

- The software entity must first consider the effects of the termination clause and determine the contract term in Step 1 before moving to Step 2 to identify separate performance obligations.
- The software entity considers whether the penalty that the customer must pay to terminate the contract each year is substantive. The software entity compares the termination penalty in each period to the remaining fees to be paid and concludes that the customer must pay a substantive termination penalty (25% of the remaining fees due under the contract) to terminate the contract. Therefore, the enforceable rights and conditions exist for the entire duration of the four-year term. We note that there may be other acceptable methods to determine whether the penalty in this scenario is significant.
- The software entity then determines that it has granted a software license with a four-year term (not four software licenses with one-year terms) to the customer.
- Assuming all revenue recognition criteria are met, the software entity would recognize $4 million in revenue at the beginning of Year 1.

EXAMPLE 16: CANCELLABLE FOUR-YEAR CONTRACT FOR SOFTWARE LICENSE WITH NO TERMINATION PENALTY

FACTS

Consider the same facts in Example 15 except that the customer is not required to pay any termination penalty if it exercises its termination right.

CONCLUSION

The software entity determines that the contract term is one year.

ANALYSIS

- Because the customer has the unilateral right to terminate the contract at the beginning of each year without paying any termination penalty, enforceable rights and obligations exist only for one year.
- The customer has three options to renew the contract annually by not exercising the termination right.
- Assume that the options do not provide the customer with material rights.
- The software entity then determines that it has granted four software licenses with one-year terms (not a software license with a four-year term) to the customer.
- Assuming all revenue recognition criteria are met, the software entity would recognize $1 million in revenue at the beginning of each of years one through four.

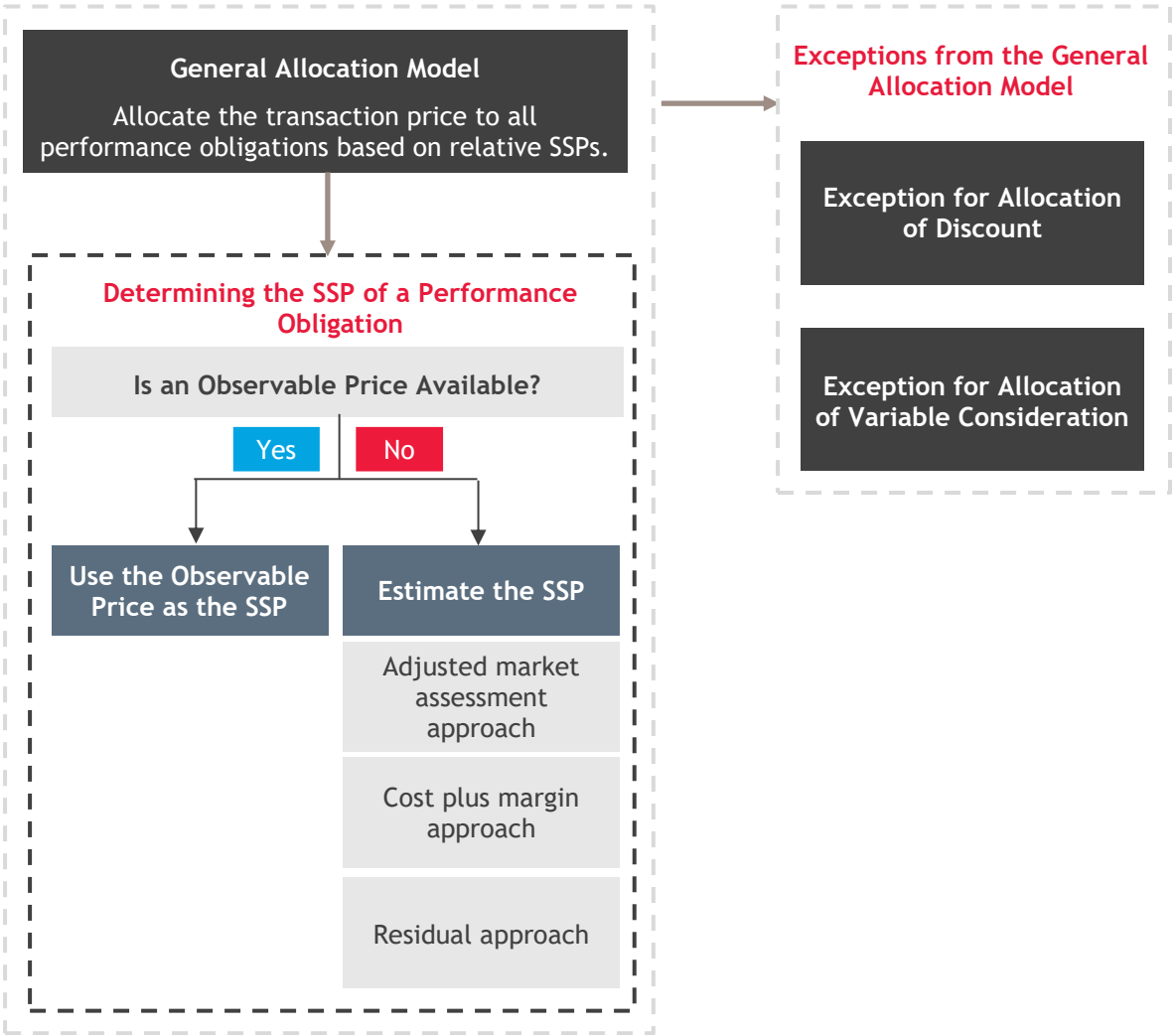
Determining Standalone Selling Price and Allocating Transaction Price

OVERVIEW: DETERMINING STANDALONE SELLING PRICE AND ALLOCATING TRANSACTION PRICE

A software entity allocates the transaction price to the performance obligations identified in the contract in Step 2. ASC 606 includes an allocation objective that generally requires the allocation of transaction price to each performance obligation based on the **relative standalone selling price** (SSP) of the goods or services underlying each performance obligation. There are **exceptions** to the general allocation objective for specific discounts and variable consideration.

In Step 4, determining the appropriate SSP for each performance obligation at contract inception is the first step to determining the amount to allocate to each performance obligation in a contract. The amount allocated to each performance obligation is ultimately recognized in revenue when or as that obligation is satisfied.

The following diagram summarizes the key concepts in Step 4:



ALLOCATION OBJECTIVE

The objective of Step 4 is to allocate an amount of transaction price to each performance obligation (or distinct good or service) that reflects the consideration to which a software entity expects to be entitled in exchange for transferring the distinct goods or services to the customer. That concept is generally referred to as the "allocation objective."

GENERAL ALLOCATION MODEL

To meet the allocation objective, a software entity generally allocates the transaction price to all performance obligations identified in the contract in proportion to the SSPs of those performance obligations at contract inception, that is, on a relative SSP basis. That allocation approach is sometimes referred to as the "general allocation model." There are exceptions to the general allocation model for discounts and variable considerations. The first step in applying the general allocation model is determining SSP for each of the performance obligations.

BDO INSIGHT 25: APPROPRIATE LEVEL FOR DETERMINING STANDALONE SELLING PRICE

The SSP is determined at the performance obligation level. When a single performance obligation comprises multiple goods or services, the SSP of that performance obligation is determined by considering the price at which a software entity would sell the bundle to a customer. That price could be different than the sum of the prices at which a software entity would separately sell the individual goods or services in the bundle.

For example, consider a software entity that sells a software license and customization services as a bundle that comprise a single performance obligation. The software entity enters a contract to sell that bundle, along with a distinct post-implementation service, to a customer. To allocate the transaction price to the two performance obligations (the bundle and the post-implementation service), the software entity must determine the SSPs of the bundle and post-implementation service. In determining the SSP of the bundle, the software entity must consider the price at which it would sell the **bundle** to a customer, rather than the price at which it would separately sell the promised goods or services within the bundle (the software license and customization services) to a customer.

The best evidence of SSP is the **observable price** of a good or service when the software entity sells that good or service separately in similar circumstances to similar customers. If the SSP is not directly observable, a software entity estimates that price at an amount that would result in the allocation of the transaction price meeting the allocation objective.

BDO INSIGHT 26: MULTIPLE STANDALONE SELLING PRICES FOR A GOOD OR SERVICE

ASC 606 states the best evidence of SSP is the price when the good or service is sold separately in similar circumstances and to similar customers. As such, the SSP of a particular good or service could vary based on the circumstances and type or class of customer.

When determining the SSP for a good or service, a software entity considers stratifying customers into meaningful groups and identifying a different SSP for each group. The groups could be based on customer type, volume of sales to customers, geography, distribution channel, or other relevant characteristics. For example, a software entity that regularly sells its software license to its retail customers within a specific price range may offer a significantly lower price to its single largest customer to promote a strategic long term business relationship.

CONTRACTUALLY STATED PRICE IS NOT PRESUMED TO BE THE STANDALONE SELLING PRICE

While a contractually stated or list price for a good or service might be the SSP for that good or service, it must not be presumed to be SSP. For example, a software entity might typically grant discounts from its list prices for software licenses, or it might not sell a distinct software license separately from other goods or services, such as PCS. In those scenarios, the list price of a distinct software license does not provide a directly observable price of that good or service.

BDO INSIGHT 27: ESTIMATING STANDALONE SELLING PRICE USING A RANGE

A software entity may sell goods or services within a range of prices rather than at a single price. While ASC 606 is silent on whether the SSP must be estimated as a precise amount or can be as a range of prices, we believe the SSP of a performance obligation can be derived from a range of amounts if **all** the following conditions are met:

- The range is sufficiently narrow.
- The software entity would sell the performance obligation separately to a customer at any point within the range.
- The resulting allocation outcome is consistent with the allocation objective.

Determining an appropriate range of SSPs requires the application of professional judgment based on the facts and circumstances. A software entity must maximize the use of observable inputs and consider the contract circumstances (for example, class of customer) in determining the range.

The use of a range is not appropriate when the SSPs are so widely dispersed that an appropriate SSP for a performance obligation in a transaction is not determinable.

When it establishes SSP as a range, a software entity allocates the transaction price as follows:

- Using the contractually stated price as the SSP if it falls within the established range.
- Using a point within the range based on the software entity's accounting policy election if the contractually stated price falls outside the established range. We believe a software entity may use the midpoint in a range or the outer limit closest to the contractually stated price as the SSP. The software entity must consistently apply its accounting policy election. Also, the resulting allocation from the software entity's accounting policy election must be consistent with the allocation objective in ASC 606.

EXAMPLE 17: ALLOCATING THE TRANSACTION PRICE — CONTRACT PRICES REFLECT SSP

FACTS

- A software entity enters a contract with a customer to sell a one-year SaaS subscription and upfront implementation services for $300,000. The SaaS subscription and the implementation services are separate performance obligations.
- Stated contract prices for the SaaS subscription and the implementation services are $250,000 and $50,000, respectively.
- The software entity's range of observable SSPs for its SaaS subscription and implementation services are $245,000 to $255,000 and $49,000 to $51,000, respectively.

CONCLUSION

The software entity uses the stated contract prices for the SaaS and the implementation services of $250,000 and $50,000, respectively, to allocate the transaction price between the two performance obligations.

ANALYSIS

- The software entity uses the stated contract prices as SSP because those prices for the SaaS subscription and the implementation services fall within the established ranges, which are sufficiently narrow.
- The stated contract price may be used, without adjustment, to allocate the transaction price to each performance obligation.

EXAMPLE 18: ALLOCATING THE TRANSACTION PRICE — CONTRACT PRICES DO NOT REFLECT SSP

FACTS

- Assume the same facts from Example 17, except the stated contract prices for the SaaS subscription and the implementation services are $235,000 and $65,000, respectively.
- The software entity has established an accounting policy to use the midpoint of its range of observable SSPs when stated contract prices fall outside the established ranges.

CONCLUSION

The software entity uses the midpoints of the ranges of $250,000 and $50,000, respectively, to allocate the transaction price between the two performance obligations.

ANALYSIS

- The software entity uses the midpoints of the ranges to allocate the transaction price to the SaaS subscription and the implementation services because the stated prices for those performance obligations fall outside their respective estimated SSP ranges.
- The midpoints of the ranges for the SaaS subscription and the implementation services are $250,000 and $50,000, respectively.

When observable standalone prices are not available, a software entity must estimate SSP. Suitable methods for estimating the SSP of a good or service include, but are not limited to, the following three approaches:



While ASC 606 does not specify a hierarchy of evidence to estimate the SSP of a good or service, a software entity must maximize the use of observable inputs when developing estimates of SSPs. Additionally, the estimation approaches (adjusted market assessment approach, expected cost plus a margin approach and residual approach) are examples of acceptable methods for estimating SSP. ASC 606 does not preclude or prescribe any particular method for estimating an SSP so long as the estimate is a faithful representation of the price at which the software entity would sell the distinct good or service if it were sold separately to the customer.

Further, a software entity may use a combination of methods to estimate the SSPs of the goods or services in a contract if two or more of those goods or services have highly variable or uncertain SSPs. For example, software entity may use a residual approach to estimate the SSP for multiple promised goods or services with highly variable or uncertain SSPs and another method to estimate the SSPs of the individual goods or services relative to that estimated aggregate SSP determined by the residual approach. However, as further discussed below, to use the residual approach to estimate the aggregate SSP for multiple promised goods or services, the observable SSPs of at least one other good or service promised in the contract must be available.

When a software entity uses a combination of methods to estimate the SSP of each promised good or service in the contract, it must evaluate whether allocating the transaction price at those estimated SSPs would be consistent the allocation objective, which requires a software entity to estimate each SSP at an amount that reflects the amount of consideration that the software entity expects to receive for transferring the promised goods or services to the customer.



INFORMATION CONSIDERED IN ESTIMATING STANDALONE SELLING PRICE

In estimating an SSP, a software entity must consider all information reasonably available based on the specific facts and circumstances, including:

- Reasonably available data points, such as a SSP of the good or service, the costs incurred to manufacture or provide the good or service, profit margins, published price listings, third-party or industry pricing, and the pricing of other goods or services in the same contract.
- Market conditions, including supply and demand for the good or service in the market, competition, restrictions, and trends.
- Entity-specific factors, such as business pricing strategy and practices.
- Information about the customer or class of customer, such as type of customer, geographical region, and distribution channel.

In considering the information available to estimate an SSP, a software entity must **both**:

- Maximize the use of observable inputs.
- Apply estimation methods consistently in similar circumstances.

Adjusted Market Assessment Approach

The adjusted market assessment approach:

- Requires an evaluation of the market in which a software entity sells goods or services to estimate the price that a customer in that market would be willing to pay for those goods or services.
- May include referring to prices from the software entity's competitors for similar goods or services and adjusting those prices as necessary to reflect the software entity's costs and margins.

Expected Cost Plus a Margin Approach

The expected cost plus a margin approach requires a software entity to forecast its expected costs of satisfying a performance obligation and then add an appropriate profit margin.

Residual Approach

The residual approach requires an estimation of the SSP by reference to the total transaction price less the sum of the observable SSPs of other goods or services promised in the contract. It may be used to estimate the standalone selling price of a good or service only if **both** of the following conditions are met:

The software entity has **observable** standalone selling prices for one or more other goods or services promised in the contract. That is, at least one good or service promised in the contract is sold on a standalone basis in other situations.

AND

The software entity sells the same good or service to different customers at or near the same time for a broad range of amounts (that is, the selling price is highly variable).

OR

The software entity has not yet established a price for that good or service, and the good or service has not previously been sold on a standalone basis (that is, the selling price is uncertain).

As previously discussed, an estimate of SSP must result in an amount that reflects the amount of consideration that the software entity expects to receive when selling the good or service on a standalone basis. Therefore, when the software entity uses the residual approach to determine the SSP of a distinct good or service, the outcome of that approach cannot realistically result in an SSP of zero if the good or service is in fact distinct. To be distinct, a good or service must have value on a standalone basis. Therefore, if the residual approach results in no, or very little, consideration being allocated to a performance obligation, the software entity must consider whether that estimate is appropriate in those circumstances. If not, the software entity must use another method to estimate SSP.

BDO INSIGHT 28: RESIDUAL APPROACH — SOFTWARE LICENSE AND MAINTENANCE SERVICE

In some situations, a software entity never sells a distinct good or service separately. Instead, it sells them as part of a bundle that may be sold for a range of amounts. This practice is common in the software industry in which software licenses are often bundled with maintenance for an initial period. Software maintenance (PCS) can typically be renewed after the initial period on a standalone basis. However, the license and the PCS will often represent separate performance obligations even though the software entity might never sell one without the other (see above discussion on identifying separate performance obligations).

In some circumstances, a software entity may have strong pricing policies for PCS by which it charges customers a fixed amount for maintenance renewals and the price does not vary from customer to customer. That could also be the case if PCS renewals are stated as a percentage of the list price of a license (that is, the list price before any customer-specific discounts or adjustments) if the list price is not subject to significant regular and artificial adjustments. That scenario leads to the question of whether it would be acceptable for a software entity to apply the residual approach to establish the SSP for a license that is never sold separately.

We believe using a residual approach to calculate the SSP of the software license is appropriate if a software entity can identify that the pricing variability that exists in the software license and PCS bundle is attributable to the software license, and that the SSP of the PCS is not highly variable. Although a software entity may not sell the software license on its own for a broad range of amounts, the software entity may sell a bundle that contains both software and PCS for a broad range of amounts. There may be observable evidence that a PCS renewal is always sold for either a fixed amount or a fixed percentage of the list price or of the contractual price of the software license. In this case, a software entity can identify that it is the license component of the bundle that is sold to different customers for a broad range of amounts, and not the PCS (because there is an observable SSP for the PCS). If there are no other goods or services in the bundle for which observable SSP does not exist, then the use of the residual approach may be appropriate to calculate the SSP of the license.

Determining SSPs of software licenses and PCS requires the application of professional judgment based on the facts and circumstances.

Other Approaches for Determining Standalone Selling Price and Allocating Transaction Price

As previously discussed, the three estimation methods listed in ASC 606 are only examples of acceptable approaches, and other estimation methods may be appropriate. One approach commonly used in the software industry is the **value relationship** approach.

Software entities often sell co-terminus bundles that include both a term software license and PCS. Because the license and PCS are bundled both upon initial purchase and renewal, a software entity might not have directly observable SSP for either the license or PCS and therefore must estimate SSP for each performance obligation.

BDO INSIGHT 29: ESTIMATING SSP BY USING A VALUE RELATIONSHIP

While ASC 606 requires a software entity to estimate the standalone selling price of a distinct performance obligation, we believe that in some instances, a software entity may instead establish a **value relationship** between a software license and PCS and use that relationship to allocate the transaction price. For example, a software entity may determine the percentages of the transaction price to allocate to each of the license and PCS based on the relationship between the two performance obligations.

As discussed above, a software entity must maximize the use of observable inputs when estimating SSP. For term licenses, relevant observable inputs could include:

- Standalone PCS renewals related to perpetual licenses of similar types of IP
- Publicly disclosed information by similar entities

A software entity may use such inputs as a starting point to determine the value relationship between the license and PCS. However, the software entity must also consider the facts and circumstances specific to its distinct promises and adjust the value relationship as appropriate. Factors to consider include:

- **Product features:** If the software entity's software includes features not found in a competitor's product or in similar perpetual licenses, then that could indicate a greater portion of the transaction price must be allocated to the license.
- **License and PCS term length:** A term of more than one year indicates that a greater portion of the transaction price must be allocated to PCS. Specifically, when using standalone PCS renewals to estimate the value of PCS related to a term license, the software entity must adjust the PCS percentage, which typically represents PCS for only one year, to reflect the full term of the license.
- **Frequency and significance of software updates:** If updates are minor or infrequent during the term, a smaller portion of the transaction price is allocated to the PCS. Conversely, if the software entity frequently releases more significant new features as part of the PCS, a greater portion of the transaction price is allocated to the PCS.
- **Level of technical support:** If the software entity provides extensive support during the term, then that could indicate a greater portion of the transaction price must be allocated to PCS.

Determining the value relationship between a software license and PCS in a co-terminus bundle requires professional judgment based on the facts and circumstances.

Renewal Options

A renewal option gives a customer the right to acquire additional goods or services of the same type as those supplied under an existing contract. Because of that difference, a software entity may elect a practical alternative to estimate the SSP of a renewal option that constitutes a material right.

Instead of estimating the SSP of the renewal option directly, a software entity can elect a practical alternative to allocate the transaction price to the optional goods or services by reference to the goods or services expected to be provided and the corresponding expected consideration if **both** of the following conditions are met:

- The additional goods or services are similar to the original goods or services in the contract (that is, the software entity continues to provide additional goods or services that are similar to what it was already providing). As a result, it may be more intuitive to view the goods or services underlying such options as part of the initial contract.
- The additional goods or services are provided in accordance with the terms of the original contract. As a result, their pricing cannot be changed beyond the parameters specified in the original contract.

In essence, that alternative approach allows a software entity to treat the material right as if the optional goods or services were additional promises in the contract and the consideration to be received when the option is exercised as additional consideration. (See Section 7.5.3.3 of BDO's Blueprint, **Revenue Recognition Under ASC 606**, for a discussion of rights to renew licenses of IP.)

BC393 of ASU 2014-09 states that the practical alternative requires an entity to include the optional goods or services that it expects to provide and the corresponding consideration that it expects to receive in the initial measurement of the transaction price. The FASB reasoned that it would be simpler for an entity to view a contract with renewal options as a contract for its expected term (that is, including the expected renewal periods) rather than as a contract with a series of renewal options.

EXAMPLE 19 (ADAPTED FROM ASC 606-10-55-343 THROUGH 55-352): OPTION THAT PROVIDES THE CUSTOMER WITH A MATERIAL RIGHT — RENEWAL OPTION

FACTS

- A software entity enters 100 separate contracts with customers to provide a software license and one year of PCS for $10,000 per contract, or $1 million in total.
- At the end of the year, each customer has the contractual right (or option) to renew the PCS contract for a second year by paying an additional $10,000.
- Customers that renew for a second year are granted the option to renew for a third year for $10,000.
- If customers do not sign up for the PCS initially (when obtaining their respective software licenses) or if they allow the PCS to lapse, the software entity charges significantly higher prices for PCS:
  - $30,000 in Year 2
  - $50,000 in Year 3
- At contract inception, the software entity estimates:
  - 90 customers are expected to renew at the end of Year 1 (90% of contracts sold).
  - 81 customers are expected to renew at the end of Year 2 (90% of the 90 customers renewing at the end of Year 1 will also renew at the end of Year 2).
- Estimated costs for a three-year contract are:

Year 1	$ 6,000
Year 2	7,500
Year 3	10,000

CONCLUSION

At contract inception, the software entity applies a portfolio approach and allocates $220,100 of consideration received to date to the option to renew at the end of Year 1 and, upon renewal at the end of the Year 1, allocates $242,800 to the option to renew at the end of Year 2.

ANALYSIS

- In Step 2 of the revenue recognition model, the software entity evaluates the customer's renewal option for PCS and determines that:
  - The renewal option provides a material right to the customer that it would not receive without entering the contract because the price for PCS is significantly higher if the customer elects to purchase the services only in Year 2 or 3.
  - Part of each customer's payment of $10,000 in Year 1 is, in effect, a nonrefundable prepayment for PCS provided in a subsequent year.
  - Thus, the software entity concludes that the promise to provide the renewal option is a performance obligation.
  - The software entity then observes that the renewal option is for a continuation of PCS provided in accordance with the terms of the existing contract.
- In Step 4 of the revenue recognition model, in accordance with the practical alternative for estimating the SSP of a renewal option, the software entity allocates the transaction price by determining the consideration it expects to receive in exchange for all the services it expects to provide under the renewal options (instead of determining the SSPs for the renewal options directly).
  - Therefore, the software entity determines that at contract inception, the expected consideration for each contract is $27,100 [$10,000 + (90% * $10,000) + (81% * $10,000)].
  - The software entity also determines that recognizing revenue based on costs incurred relative to the total expected costs depicts the transfer of services to the customer.
  - Accordingly, the pattern of revenue recognition expected at contract inception for each contract is:

	EXPECTED COSTS ADJUSTED FOR LIKELIHOOD OF CONTRACT RENEWAL	ALLOCATION OF CONSIDERATION EXPECTED
Year 1	$ 6,000 ($6,000 * 100%)	$ 7,799 [($6,000/$20,850) * $27,100]
Year 2	6,750 ($7,500 * 90%)	8,773 [($6,750/$20,850) * $27,100]
Year 3	8,100 ($10,000 * 81%)	10,528 [($8,100/$20,850) * $27,100]
Total	$20,850	$27,100

  - Therefore, at contract inception, the software entity allocates $220,100 of the consideration received to date (cash of $1 million less revenue of $779,900 ($7,799 * 100) to be recognized in Year 1) to the option to renew at the end of Year 1.
- In Step 5 of the revenue recognition model, assuming there is no change in its expectations and the 90 customers renew as expected, at the end of Year 1, the software entity has collected cash of $1.9 million [(100 * $10,000) + (90 * $10,000)], recognized revenue of $779,900 ($7,799 * 100), and recognized a contract liability of $1,120,100.
  - Consequently, upon renewal at the end of Year 1, the software entity allocates $242,800 to the option to renew at the end of Year 2 (cumulative cash of $1.9 million less cumulative revenue of $1.657 million ($779,900 + $8,773 * 100) recognized in Year 1 and to be recognized in Year 2).
  - If the actual number of contract renewals differs from what it expected, the software entity would update the transaction price and the revenue recognized accordingly.

ALLOCATION OF VARIABLE CONSIDERATION

The application of the general allocation model based on relative SSPs to a contract that includes multiple performance obligations and variable consideration results in the allocation of variable consideration to all performance obligations in the contract. However, that might not necessarily result in a faithful depiction of the amount of consideration to which the software entity expects to be entitled from the customer upon satisfying a particular performance obligation if the variable consideration does not relate to all the performance obligations in the contract. For example, a software entity might contract to sell two distinct services, a SaaS subscription and upfront customization services, which are delivered over different periods of time with a variable usage-based fee payable by the customer that is contingent on the usage of the SaaS platform. It might be inappropriate to allocate the usage-based fee (variable consideration included in the transaction price) to both services. Therefore, an exception to the general allocation model was provided for allocating variable consideration (the variable consideration allocation exception).

The variable consideration allocation exception states that a software entity allocates a variable amount of consideration (and subsequent changes to that amount) entirely to a single performance obligation (or a distinct good or service that forms part of a single performance obligation that meets the definition of a series) if **both** of the following criteria are met:

- The terms of a variable payment relate specifically to the software entity's efforts to satisfy the performance obligation or transfer the distinct good or service (or to a specific outcome from satisfying the performance obligation or transferring the distinct good or service).
- When considering all the performance obligations and payment terms in the contract, the allocation of the variable amount in its entirety to a performance obligation (or distinct good or service) is consistent with the allocation objective that the transaction price must be allocated to each performance obligation (or distinct good or service) to reflect the consideration to which the software entity expects to be entitled in exchange for transferring the promised good or service.

A software entity applies the general allocation model and the discount allocation exception to allocate the remaining amount of transaction price that does not meet the criteria for the variable consideration allocation exception.



TRG DISCUSSIONS: ALLOCATION OF VARIABLE CONSIDERATION TO A DISTINCT GOOD OR SERVICE IN A SERIES

In July 2015, the TRG discussed whether the allocation of variable consideration to a distinct good or service in a series must be based on SSPs. That approach could have limited the number of transactions that qualify for the variable consideration allocation exception because it might imply that each distinct service that is substantially the same would need to be allocated the same amount (absolute value) of variable consideration.

ASC 606-10-32-29 states that to meet the allocation objective, an entity must allocate the transaction price to each performance obligation on a relative SSP basis. However, that paragraph specifically excludes from its requirements ASC 606-10-32-39 through 32-41 on allocating variable consideration to a distinct service in a series. Further, ASC 606-10-32-30 states that the guidance in ASC 606-10-32-31 through 32-41 on the relative SSP allocation does not apply to the allocation of variable consideration. BC280 of ASU 2014-09 describes that while SSP is the default method for determining whether the allocation objective is met, the FASB decided that other methods could be used in some instances and therefore included the guidance on allocating variable consideration.

Thus, the TRG agreed that an allocation based on relative SSPs is not required to meet the allocation objective when allocating variable consideration to a distinct good or service in a series. However, as illustrated in Example 35 in ASC 606 (see adaptations in Examples 5-5 and 5-6), when variable consideration is allocated to different performance obligations, SSPs might sometimes (but do not have to) be used to determine the reasonableness of the allocation.

The FASB did not describe other methods that could be used to comply with the allocation objective other than stating in ASC 606-10-32-40(b) that an entity must consider all the payment terms and performance obligations. For instance, an entity might consider its effort to fulfill the obligation and whether the variable consideration is commensurate with the value provided to the customer. As such, an entity must apply judgment to determine whether the allocation results in a reasonable outcome.

In addressing those concepts, the TRG discussed three non-exhaustive examples (that is, there could be other reasons why a variable fee would or would not meet the allocation objective).

Example A

IT Seller and IT Buyer execute a 10-year IT outsourcing arrangement in which IT Seller provides continuous delivery of outsourced activities over the contract term. For example, IT Seller will provide server capacity, manage the customer's software portfolio, and run an IT help desk. The total monthly invoice is calculated based on different units consumed for the respective activities. For example, the billings might be based on millions of instructions per second of computing power, number of software applications used, or number of employees supported, and the price per unit differs for each type of activity.

Before the delivery of the service, IT Seller performs initial set-up activities to be able to provide the other services in the contract. IT Seller charges the IT Buyer a nonrefundable upfront fee related to the transition activities. IT Seller concludes that the set-up activities do not transfer services to the customer.

The per-unit price IT Seller charges declines over the life of the contract. The agreed pricing at the onset of the contract is considered to reflect market pricing. The pricing decreases to reflect the associated costs decreasing over the term of the contract as the level of effort to complete the tasks decreases. Initially, the tasks are performed by more expensive personnel for activities that require more effort. Later, the level of effort for the activities decreases, and the tasks are performed by less expensive personnel. The contract includes a price benchmarking clause whereby the IT Buyer engages a third-party benchmarking firm to compare the contract pricing to current market rates at specific points in the contract term. The price automatically adjusts if the benchmark is significantly below IT Seller's price.

IT Seller concludes that there is a single performance obligation that is satisfied over time because the customer simultaneously receives and consumes the benefits provided by its services as it performs.

The events that trigger the variable consideration are the same throughout the contract, but the price per unit decreases each year. Even with the declining prices, the allocation objective could be met if the pricing is based on market terms, or if the changes in price are substantive and linked to changes in the entity's cost to fulfill the obligation or value provided to the customer based on the third-party benchmarking data.

Example B

Transaction Processer (TP) enters a 10-year agreement with a customer. Over the 10-year period, TP will provide continuous access to its system and process all transactions on behalf of the customer. The customer is obligated to use TP's system to process all its transactions; however, the ultimate quantity of transactions is unknown. TP concludes that the customer simultaneously receives and consumes the benefits as it performs.

TP charges the customer per transaction. For each transaction, the customer is charged a contractual rate per transaction and a percentage of the total dollars processed. TP also charges the customer a fixed upfront fee at the beginning of the contract.

If the nature of the entity's promise is a single service to process as many transactions as the customer requires, the fees based on quantity processed and the fees based on a percentage of dollars processed could meet the allocation objective for each month of service. For example, the allocation objective could be met if the fees are priced consistently throughout the contract and the rates charged are consistent with the entity's standard pricing practices with similar customers.

Example C

Hotel manager (HM) enters a 20-year agreement to manage properties on behalf of the customer. HM receives monthly consideration based on 1% of monthly rental revenue, reimbursement of labor costs incurred to perform the service, and an annual incentive payment based on 8% of gross operating profit. HM concludes that the customer simultaneously receives and consumes the benefits provided by its services as it performs.

The base monthly fees could meet the allocation objective for each month because there is a consistent measure throughout the contract period that reflects the value to the customer each month (the percentage of monthly sales). Similarly, if the labor cost reimbursements are commensurate with the entity's efforts to fulfill the promise each day, the allocation objective for those variable fees could also be met. Finally, the allocation objective could be met for the incentive fee if it reflects the value delivered to the customer for the annual period (reflected by the profits earned) and is reasonable compared to the incentive fees that could be earned in other periods.

EXAMPLE 20 (ADAPTED FROM ASC 606-10-55-270 THROUGH 55-279): ALLOCATION OF VARIABLE CONSIDERATION ENTIRELY TO ONE PERFORMANCE OBLIGATION

FACTS

- A software entity enters a contract with a customer for two software licenses (Licenses A and B).
- Each license represents a separate performance obligation that is satisfied when it is transferred to the customer (that is, at a point in time).
- The SSPs of Licenses A and B are $1,200 and $1,500, respectively.
- The consideration payable by the customer is:
  - License A: a fixed amount of $1,200.
  - License B: a royalty payment of 5% of the selling price of the customer's future sales of products that use the IP to which License B relates.
- The software entity estimates that the amount of sales-based royalties it will be entitled to in respect of License B will be $1,500.

CONCLUSION

The software entity allocates the fixed payment of $1,200 to License A and allocates the variable royalty payment entirely to License B.

ANALYSIS

- The software entity considers the criteria in the variable consideration allocation exception to determine the allocation of the transaction price to each of the licenses.
- The software entity determines that:
  - The variable consideration (sales-based royalty) relates specifically to an outcome from the software entity's performance obligation to transfer License B (that is, the customer's subsequent sales of products that use that license).
  - Allocating the expected sales-based royalty amounts of $1,500 entirely to License B is consistent with the allocation objective because the estimated amount of royalties approximates the SSP of License B ($1,500) and the fixed amount of $1,200 approximates the SSP of License A.
- Based on an assessment of the facts and circumstances relating to both licenses, the software entity determines that allocating some of the fixed consideration ($1,200) to License B in addition to all the variable consideration would not meet the allocation objective.

- Therefore, the software entity allocates the transaction price in accordance with the variable consideration allocation exception as:
  - License A: a fixed payment of $1,200.
  - License B: the variable royalty payment.
- Although revenue of $1,200 will be recognized for License A when it is transferred to the customer, no revenue will be recognized when License B is transferred to the customer.
- Sales-based royalties allocated to License B are not recognized before the subsequent sales of the customer's products that use License B take place because of the exception in ASC 606-10-55-65 that precludes estimation of sales- or usage-based royalties for licenses of IP.

EXAMPLE 21 (ADAPTED FROM ASC 606-10-55-270 THROUGH 55-279): ALLOCATION OF VARIABLE CONSIDERATION BASED ON STANDALONE SELLING PRICES

FACTS

Assume the same facts as in Example 20 except that the prices included in the contract are:

- License A: a fixed amount of $450
- License B: a royalty payment of 7.5% of the selling price of the customer's future sales of products that use license B.
- The software entity estimates that the amount of sales-based royalties that it will be entitled to in respect of License B will be $2,250.

CONCLUSION

The software entity allocates both the fixed consideration and the sales-based royalties to Licenses A and B using their relative SSPs.

ANALYSIS

- The software entity considers the criteria in the variable consideration allocation exception to determine the allocation of the transaction price to each of the licenses.
- The software entity determines that although the variable payments (the sales-based royalties) relate solely to the transfer of License B, allocating the variable consideration only to License B would be inappropriate because allocating $450 to License A and $2,250 to License B would not reflect a reasonable allocation based on the licenses' SSPs. As a result, the software entity must apply the general allocation model based on relative SSPs.
- The software entity allocates the fixed amount of consideration of $450 to the two licenses based on their SSPs. The allocation is calculated as:

PRODUCT	ALLOCATED TRANSACTION PRICE	
License A	$ 200	($450 * ($1,200/$2,700))
License B	250	($450 * ($1,500/$2,700))
Total	$ 450	

- The software entity allocates the variable consideration as the sales by the customer of products that use License B occur. The sales-based royalties are allocated to Licenses A and B on a relative SSP basis.

- License A is transferred to the customer three months after contract inception and License B is transferred at contract inception. As a result:
  - Revenue of $250 is recognized for License B at contract inception when License B is transferred to the customer.
  - Revenue of $200 is recognized for License A three months after the contract inception when License A is transferred to the customer.
  - Recognition of the sales-based royalty allocated to each of the two licenses is deferred to future periods because ASC 606-10-55-65 precludes the recognition of sales-based royalty before the related sales occur.
- Although the royalty relates solely to the transfer of License B, the allocation of the fixed consideration ($450) to License A and the entire sales-based royalties to License B (estimated at $2,250) is disproportionate in comparison with the SSPs of the two licenses. In other words, there is pricing interdependency between the two licenses, indicating that some of the royalty generated by License B in fact relates to the sale of License A and that some of the fixed license fee ($450) stated in the legal contract as relating solely to License A also relates to the sale of License B.

EXAMPLE 22: ALLOCATION OF VARIABLE CONSIDERATION ENTIRELY TO ONE PERFORMANCE OBLIGATION

FACTS

Assume the same facts as in Example 20 except that the prices included in the contract are:

- License A: a fixed amount of $1,950.
- License B: a royalty payment of 2.5% of the selling price of the customer's future sales of products that use License B.
- The software entity estimates that the amount of sales-based royalties that it will be entitled to in respect of License B will be $750.

CONCLUSION

The software entity allocates the estimated sales-based royalties of $750 entirely to License B and allocates fixed consideration of $1,950 between Licenses A and B based on their relative SSPs after the royalty allocation.

ANALYSIS

- The software entity considers the criteria in the variable consideration allocation exception to determine the allocation of the transaction price to each of the two licenses. It determines that:
  - The variable consideration (sales-based royalty) relates specifically to an outcome from its performance obligation to transfer License B (that is, the customer's subsequent sales of products that use that license).
  - Allocating the expected sales-based royalty amounts of $750 entirely to License B is consistent with the allocation objective.
  - Although the estimated amount of royalties is less than the SSP of License B ($1,500), the fixed amount of $1,950 is also greater than the SSP of License A ($1,200). Therefore, the software entity allocates the fixed amount of $1,950 to each of Licenses A and B after allocating the expected sales-based royalty amounts of $750 entirely to License B.
- The software entity allocates the fixed amount of consideration of $1,950 to the two licenses based on their SSPs after allocating the variable amount to License B. The allocation is calculated as:

PRODUCT	ALLOCATED TRANSACTION PRICE	
License A	$ 1,200	($1,950 * ($1,200/$1,950))
License B	750	($1,950 * (($1,500 — $750)/$1,950))
Total	$ 1,950	

Software entities price software licenses and SaaS subscriptions based on usage or other variable metrics. Determining whether to allocate the usage-based fees solely to one or more periods of time in a series, especially when the pricing includes guaranteed minimums or other fixed fees can be complex and require judgment, as illustrated in Examples 23 through 26.

EXAMPLE 23: USAGE-BASED FEE ALLOCATED TO THE PERIOD IN WHICH THE USAGE OCCURS

FACTS

- A software entity enters a contract with a customer to provide access to a SaaS platform for one year.
- The customer pays both:
  - $10,000 fixed fee paid upfront.
  - Variable amount paid monthly at $10 per transaction processed using the SaaS platform.
- Assume that:
  - Access to the SaaS platform is a single performance obligation to provide access to the hosted solution satisfied over time.
  - The SaaS access comprises a series of distinct days of service, and a time-based measure of progress is appropriate to measure the satisfaction of the performance obligation.
  - The upfront fee does not result in a material right.
  - The as-invoiced practical expedient cannot be applied because of the upfront fee.

CONCLUSION

The software entity recognizes the fixed fee ratably over the one-year contract period and the variable usage-based fee in the period in which the usage occurs.

ANALYSIS

- The fixed fee is attributable to the entire performance obligation and is recognized ratably over the one-year contract period.
- The usage-based variable fee is allocated to the distinct day of service in which the usage occurs because both variable consideration allocation exception criteria are met:
  - The usage-based fee relates specifically to the customer's usage of the SaaS platform that day.
  - Allocating the usage-based fees to each day is consistent with the allocation objective because:
    - Each day has a similar pricing structure, and
    - The allocation of fixed and variable fees depicts what the software entity would expect to receive for each day of service.
- Based on the above allocation conclusion, the following amounts of revenue are recognized each quarter:

REPORTING PERIOD	USAGE (NUMBER OF TRANSACTIONS PROCESSED)	FIXED FEE (RECOGNIZED RATABLY)	VARIABLE USAGE-BASED FEE (USAGE * $10)	TOTAL REVENUE
Q1	100	$2,500	$1,000	$3,500
Q2	95	2,500	950	3,450
Q3	125	2,500	1,250	3,750
Q4	120	2,500	1,200	3,700
	440	$10,000	$4,400	$14,400

EXAMPLE 24: USAGE-BASED FEE WITH QUARTERLY TIERED PRICING IN WHICH THE ALLOCATION OBJECTIVE IS MET FOR EACH QUARTER

FACTS

- A software entity enters a contract with a customer to provide access to a SaaS platform for one year.
- The customer pays both:
  - $10,000 fixed upfront fee
  - A variable amount per transaction processed using the SaaS platform based on a tiered pricing that resets each quarter:

NUMBER OF TRANSACTIONS PROCESSED	PRICE PER TRANSACTION
0-250	$10
251-500	$9
>500	$7.5

- Assume that:
  - Access to the SaaS platform is a single performance obligation to provide access to the hosted solution satisfied over time.
  - The SaaS access comprises a series of distinct days of service, and a time-based measure of progress is appropriate to measure the satisfaction of the performance obligation.
  - The upfront fee does not result in a material right.
  - The as-invoiced practical expedient cannot be applied because of the upfront fee.

CONCLUSION

The software entity recognizes the fixed fee ratably over the one-year contract period and the variable usage-based fee in the quarter in which the usage occurs.

ANALYSIS

- The fixed fee is attributable to the entire performance obligation and is recognized ratably over the one-year contract period.
- The software entity evaluates whether it allocates variable amounts entirely to one or more of the distinct service periods as follows:
  - The criterion in ASC 606-10-32-40(a) to allocate the amounts earned to each distinct day of service is **not** met because the variable amounts are specifically related to the cumulative number of transactions processed in a quarter rather than each day. For example, the amount earned on the 251st transaction in a quarter depends on the previous 250 transactions purchased during that quarter and the amount earned on the 501st transaction during a quarter depends on the previous 500 transactions purchased during that quarter. Therefore, the variable consideration relates to the software entity's efforts to transfer distinct services every day in the quarter rather than on an individual day.
  - The criterion in ASC 606-10-32-40(a) is met for each quarter, and the variable amounts are allocated entirely to the quarter in which they are earned. The pricing resets each quarter such that the variable amounts earned from each transaction processed relate specifically to the software entity's efforts to provide access to the SaaS platform in the quarter in which the transaction is processed.
- The software entity concludes that allocating the amounts to each quarter would meet the allocation objective (the criterion in ASC 606-10-32-40(b)) because each quarter has a consistent price structure throughout the term of the contract.
- In recognizing revenue, the software entity considers that:
  - If the contractual quarters are coterminous with the software entity's quarterly reporting periods, the software entity could recognize the amounts as the transactions are billed and achieve the same outcome for the reporting period.
  - If the contractual quarters are not coterminous with the software entity's quarterly reporting periods, the software entity would need to estimate the transaction volumes for the contractual quarterly periods to determine the revenue to recognize during the quarterly reporting period and update that estimate each reporting period. The requirement to update the estimate could result in a cumulative catch-up adjustment during the quarter.

EXAMPLE 25: USAGE-BASED FEE WITH MONTHLY GUARANTEED MINIMUM IN WHICH THE ALLOCATION OBJECTIVE IS MET FOR EACH MONTH

FACTS

- A software entity enters a contract with a customer to provide access to a SaaS platform for two years.
- The customer pays:
  - $10,000 fixed upfront fee.
  - A variable amount per transaction processed during each month using the SaaS platform priced at $10 per transaction processed.
  - A guaranteed minimum of $20,000 for 2,000 transactions each month, that is, $480,000 for two years.
- Assume that:
  - Access to the SaaS platform is a single performance obligation to provide access to the hosted solution satisfied over time.

- The SaaS access comprises a series of distinct days of service and a time-based measure of progress is appropriate to measure the satisfaction of the performance obligation.
- The upfront fee does not result in a material right.
- The guaranteed minimum amount is substantive.
- The as-invoiced practical expedient cannot be applied because of the upfront fee.

CONCLUSION

The software entity recognizes the fixed fee of $10,000 and the minimum guaranteed amount of $480,000 ratably over the two-year contract period. It recognizes any variable usage-based fees above the monthly minimum guaranteed amount in the month in which the usage occurs.

ANALYSIS

- The fixed fee is attributable to the entire performance obligation and recognized over the two-year contract period. Similarly, the minimum guaranteed amount of $480,000 is recognized over the two-year contract period.
- The usage-based variable fee resulting from monthly usage above 2,000 transactions is allocated to the month in which the usage occurs because both variable consideration allocation exception criteria are met:
  - The usage-based fee relates specifically to the customer's usage of the SaaS platform in that month. The variable amounts earned in a month are not related to the customer's usage in a previous or later month because the minimum guarantee resets each month.
  - Allocating the usage-based fees to each month is consistent with the allocation objective because each month has a similar pricing structure and the allocation of fixed and variable fees depicts what the software entity would expect to receive for each month of service. Specifically, the variable amounts earned in a month are commensurate with the benefits received or transactions processed by the customer.
- In recognizing revenue, the software entity considers that:
  - If the contractual months are co-terminus with the software entity's monthly reporting periods, the software entity could recognize the amounts as the transactions are billed and achieve the same outcome for the reporting period.
  - If the contractual months are not coterminous with the software entity's monthly reporting periods, the software entity would need to estimate the transaction volumes for the contractual monthly periods to determine the revenue to recognize during the monthly reporting period and update that estimate each reporting period. The requirement to update the estimate could result in a cumulative catch-up adjustment during the month.

EXAMPLE 26: USAGE-BASED FEE WITH ANNUAL GUARANTEED MINIMUM IN WHICH THE ALLOCATION OBJECTIVE IS MET FOR THE YEAR (NOT MONTH OR QUARTER)

FACTS

Assume the same facts in Example 25 except the contract includes a guaranteed minimum of $240,000 for 24,000 transactions each year, that is, $480,000 for two years.

CONCLUSION

The software entity recognizes revenue as follows:

- The fixed fee of $10,000 and the minimum guaranteed amount of $480,000 using an appropriate measure of progress over the two-year contract period.

- Any variable usage-based fee above the annual minimum guaranteed amount is estimated at contract inception and updated at each reporting date during the year. The estimated variable amount and updates are recognized over the year using the same measure of progress as the fixed fees.

ANALYSIS

- The fixed fee is attributable to the entire performance obligation and recognized over the two-year contract period. Similarly, the minimum guaranteed amount of $480,000 is recognized over the two-year contract period.
- The usage-based variable fee resulting from annual usage above 24,000 transactions is allocated to the year in which the usage occurs because both variable consideration allocation exception criteria are met:
  - The usage-based fee relates specifically to the customer's usage of the SaaS platform in that year. The variable amounts earned in a year are not related to the customer's usage in a previous or later year because the minimum guarantee resets each year.
  - Allocating the usage-based fees to each year is consistent with the allocation objective because each year has a similar pricing structure and the allocation of fixed and variable fees depicts what the software entity would expect to receive for each year of service. Specifically, the variable amounts earned in a year are commensurate with the benefits received or transaction processed by the customer.
- The software entity **cannot** allocate the usage-based variable fee earned in a year entirely to a distinct period within that year, for example, a day, month, or quarter. This is because the variable amounts are specifically related to the **cumulative** transactions processed in a year. For example, if a customer exceeds the annual minimum in month seven, the usage-based variable amount earned in month seven depends on the purchases in the previous six months during that year. Therefore, the variable consideration earned in month seven relates to the software entity's efforts to transfer distinct services for all the previous months in the year rather than only month seven.
- The software entity estimates the usage-based variable amounts for the first year (not the two-year term, since the minimum guaranteed amount resets each year) at contract inception and recognizes that amount over the year. The software entity updates that estimate and revenue recognized at each reporting date, which may result in recognizing a cumulative catch-up adjustment.

BDO INSIGHT 30: ALLOCATION OF VARIABLE CONSIDERATION AND A DISCOUNT

Some contracts with customers include both variable consideration and a discount that is variable. In those cases, a question arises regarding the sequence of applying the guidance on allocation of variable consideration and on allocation of a discount.

A software entity first needs to determine whether the discount is variable consideration. Then, consistent with the TRG discussions on this matter, we believe a software entity must apply the guidance on variable consideration allocation and then apply the guidance on general allocation or on allocating a discount (that is not variable consideration) to the remaining transaction price. Determining the appropriate allocation and accounting requires the application of professional judgment based on facts and circumstances.

In the absence of specific guidance in ASC 606, we believe multiple approaches could be acceptable for determining the amount that represents a discount (rather than variable consideration) and is therefore not within the scope of the guidance on variable consideration allocation. Any approach selected must result in an allocation that is consistent with the criteria for the variable consideration allocation exception and the allocation objective.

Appendix A: BDO Blueprints



BDO's Professional Practice — Accounting Group has issued the following Blueprints, which provide detailed accounting guidance, interpretive insights, and practical examples on complex areas of U.S. GAAP:

- Revenue Recognition Under ASC 606
- Accounting for Software Costs
- Expense Disaggregation Disclosures Under ASC 220-40
- Statement of Cash Flows Under ASC 230
- Segment Reporting Under ASC 280
- Equity Method Investments Under ASC 323
- Impairments under ASC 350 and ASC 360
- Share-based Payments Under ASC 718
- Business Combinations Under ASC 805
- Control and Consolidation Under ASC 810
- Accounting for Leases Under ASC 842
- Issuer's Accounting for Complex Financial Instruments

BDO's Professional Practice — SEC Services Group has issued the following Blueprints to help SEC registrants apply the SEC's rules and regulations related to financial reporting:

- Guide to Going Public

In addition to Blueprints, BDO's Professional Practice has issued a suite of publications related to U.S. GAAP and SEC rules and regulations:

- Bulletins: short, timely communications delivering essential news, alerts, and developments in accounting and SEC reporting
- Snapshots: summaries of SEC rules and regulations
- SEC Reporting Insights, updated annually
- Conference Highlights: Current SEC & PCAOB Developments, updated annually
- Effective Date Reminders, updated annually
- Industry Supplements

For more on IFRS, visit BDO's IFRS and Corporate Reporting website.

Find our latest accounting and reporting publications on BDO's Accounting, Reporting and Compliance Hub (ARCH) at arch.bdo.com



Appendix B: Summary of Significant Changes

The following table lists the significant changes in this November 2026 edition of the Blueprint as compared to the October 2025 edition. The following substantive changes were made in this edition:

DESCRIPTION OF CHANGE
Added BDO Insight 15 and Example 9A related to SaaS platforms that incorporate AI
Added Examples 12, 13 and 14 illustrating the assessment of whether a reseller is the principal or agent
Added a new section “Determining Standalone Selling Price and Allocating Transaction Price” (step 4 of the model), including BDO Insights 25 through 30 and Examples 17 through 30

Contacts

JENNIFER KIMMEL
Professional Practice Principal – Accounting
203-905-6284 / **jkimmel@bdo.com**

ANGELA NEWELL
Deputy Chief Accountant
214-689-5669 / **ajnewell@bdo.com**

HANK GALLIGAN
National Technology Industry Leader
617-422-7521 / **hgalligan@bdo.com**

STEPHANIE HEWLETT
National Technology Industry Assurance Leader
801-456-5718 / **shewlett@bdo.com**

FASB publications excerpted in this publication were reprinted with permission. Copyright 2026 by Financial Accounting Foundation, Norwalk, Connecticut.

At BDO, our purpose is to be the people you count on to grow with confidence and achieve what matters most. Our professionals provide assurance, tax and advisory services for a diverse range of clients across the U.S. and in over 160 countries through our global organization. We are proud to be an ESOP company, inspiring an ownership mindset that enables our collective success.

BDO is the brand name for the BDO network and for each of the BDO Member Firms. BDO USA, P.C., a Virginia professional corporation, is the U.S. member of BDO International Limited, a UK company limited by guarantee, and forms part of the international BDO network of independent member firms. For more information, please visit: **www.bdo.com**.

© 2026 BDO USA, P.C. All rights reserved.