REVENUE RECOGNITION UNDER ASC 606 – AN INDUSTRY SUPPLEMENT TO A BDO BLUEPRINT

Identifying Performance Obligations in the Software Industry

November 2023

BDO

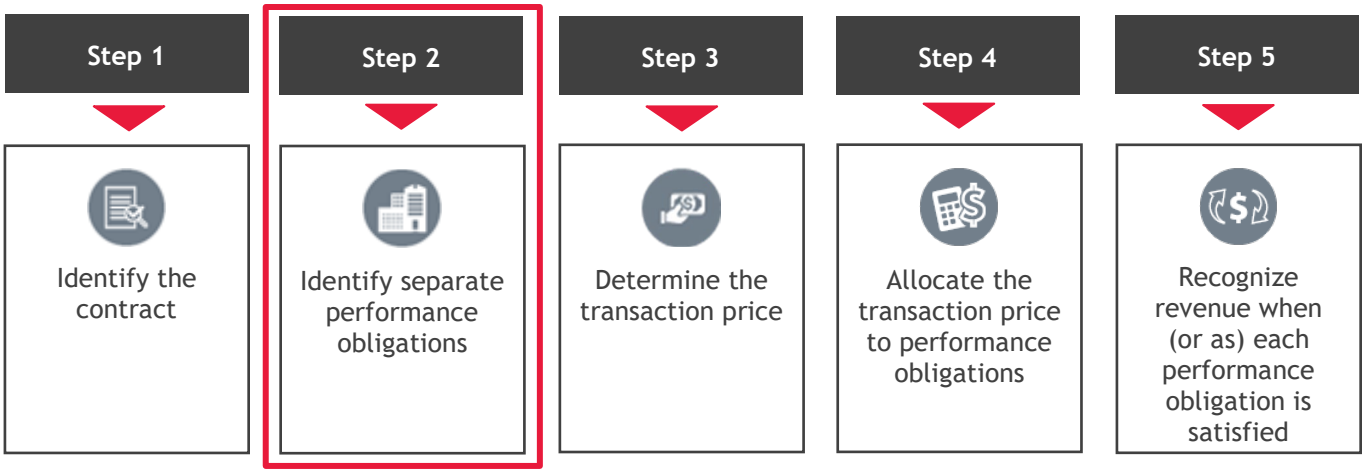
TABLE OF CONTENTS

IDENTIFYING PERFORMANCE OBLIGATIONS IN THE SOFTWARE INDUSTRY (STEP 2)

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, unique nature of software products and the continuous evolution of the software industry in terms of software products or 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 the considerations specific to the software industry in identifying the performance obligations in a contract with a customer (or Step 2). However, throughout this publication we make references to guidance included in Step 1 or Steps 3 through 5 to give better context or explain the effects of the conclusions reached in Step 2. We encourage you to read the interpretations and examples in this publication with our **Blueprint on revenue recognition**, 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 our **Blueprint on revenue recognition**.

OVERVIEW

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, which 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, which the customer can access through the internet and an offline version of the software application, which 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:

- An upfront fee for the entire license or SaaS subscription term
- Annual fees for each year during the license or SaaS subscription term
- A fee based on the number of active users
- A fee based on the number or dollar value of transactions processed using the underlying software
- Any combination of the above

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 INSIGHTS — GUIDANCE THAT APPLIES TO SOFTWARE LICENSES AND SAAS IS DIFFERENT

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

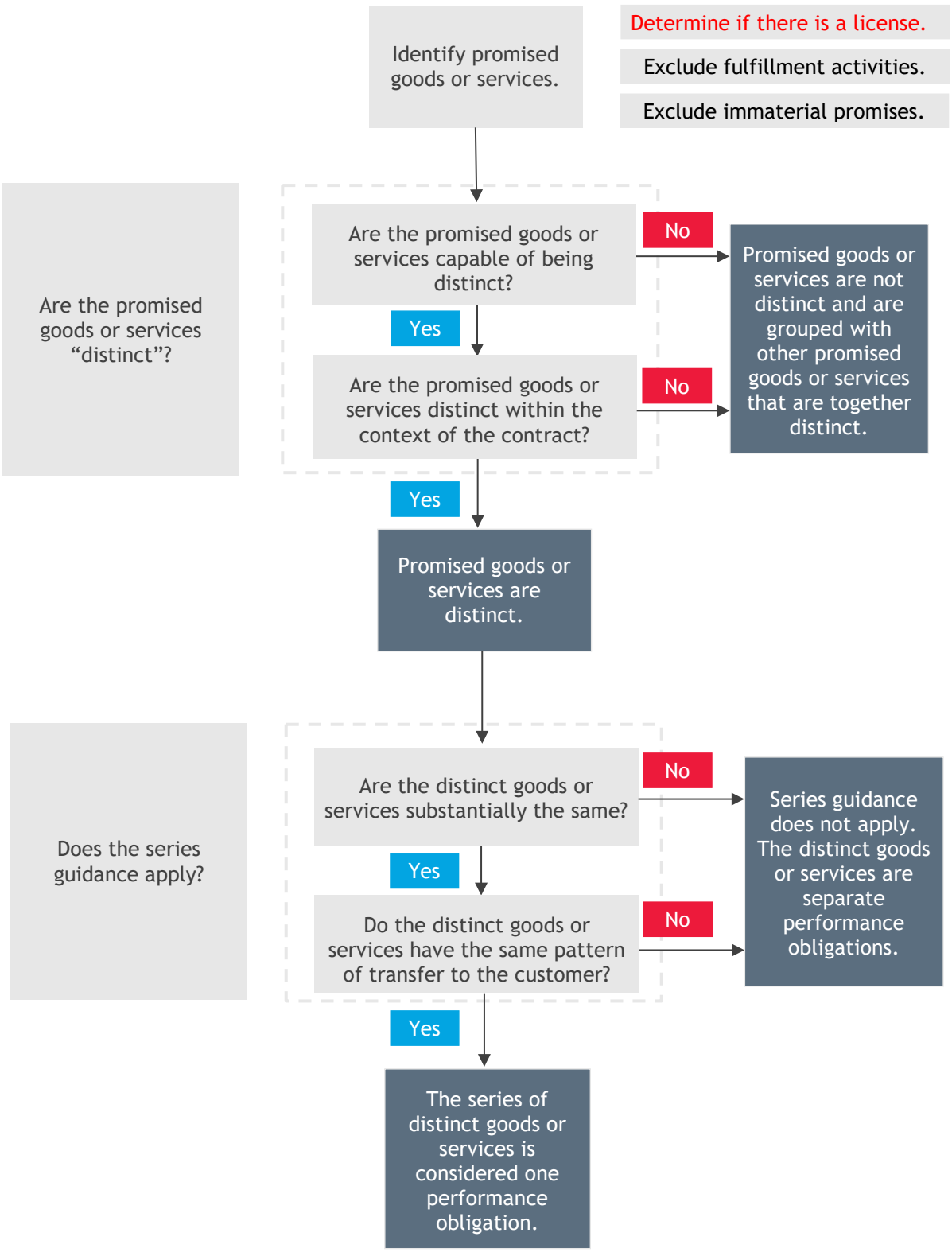
The licensing guidance does not apply to SaaS arrangements. Revenue from SaaS is recognized in the same manner as revenue from any other services under the five-step revenue recognition model. In particular, the exception from estimating certain sales- or usage-based fees described in ASC 606-10-55-65 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:



BDO INSIGHTS — 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 of 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:

- Software license
- Cloud-based service (for example, SaaS)
- PCS, which may include technical support and unspecified updates, upgrades, and enhancements
- Professional services (for example, implementation services, data migration services, customization services)
- Customer options (for example, a renewal right or a right to convert a software license to SaaS)
- Arranging for another party to grant a software license or provide SaaS as a reseller
- Hardware

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 contract with an unrelated party to host the software.

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.

[1] ASC 985-20 defines a hosting arrangement as *"[i]n 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."*

BDO INSIGHTS — 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.

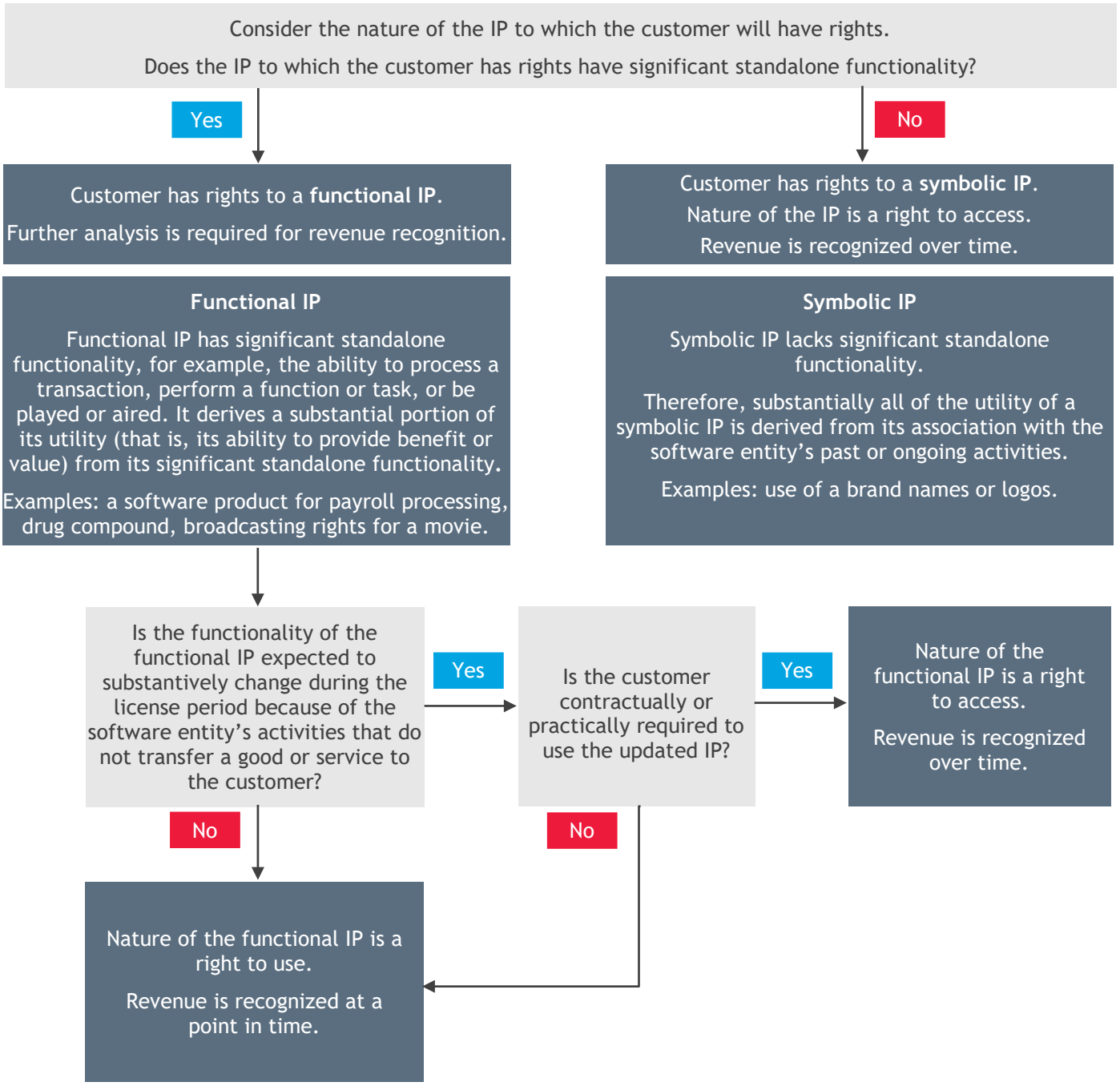
A software entity that grants a software license must apply the licensing guidance to determine the nature of its promise as either a right to access or a right to use the IP. This determination affects the conclusion about whether the software license transfers to a customer at a point in time or over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


Consider the nature of the IP to which the customer will have rights.
Does the IP to which the customer has rights have significant standalone functionality?
Yes
No
Customer has rights to a **functional IP**.
Further analysis is required for revenue recognition.
Customer has rights to a **symbolic IP**.
Nature of the IP is a right to access.
Revenue is recognized over time.
Functional IP
Functional IP has significant standalone functionality, for example, the ability to process a transaction, perform a function or task, or be played or aired. It derives a substantial portion of its utility (that is, its ability to provide benefit or value) from its significant standalone functionality.
Examples: a software product for payroll processing, drug compound, broadcasting rights for a movie.
Symbolic IP
Symbolic IP lacks significant standalone functionality.
Therefore, substantially all of the utility of a symbolic IP is derived from its association with the software entity's past or ongoing activities.
Examples: use of a brand names or logos.
Is the functionality of the functional IP expected to substantively change during the license period because of the software entity's activities that do not transfer a good or service to the customer?
Yes
Is the customer contractually or practically required to use the updated IP?
Yes
Nature of the functional IP is a right to access.
Revenue is recognized over time.
No
No
Nature of the functional IP is a right to use.
Revenue is recognized at a point in time.

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 (which 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 below 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

A software developer contracts 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 information technology). The installation service is routinely performed by other competitor entities and does not significantly modify the software. The software remains functional without the updates and the technical support.

The software developer identifies four performance obligations in the contract:

- The software license
- Installation services
- Software updates
- Technical support

In assessing the nature of its promise to transfer the software license, the software developer first concludes that the software 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.

Further, the software developer considers whether the functional IP license represents a right to access or a right to use its IP and concludes the following:

- 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 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 had 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.

Based on this analysis, 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, the software developer accounts for the software license performance obligation as a performance obligation satisfied at a point in time.

BDO INSIGHTS — 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 which 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.

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

BDO INSIGHTS — 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 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.

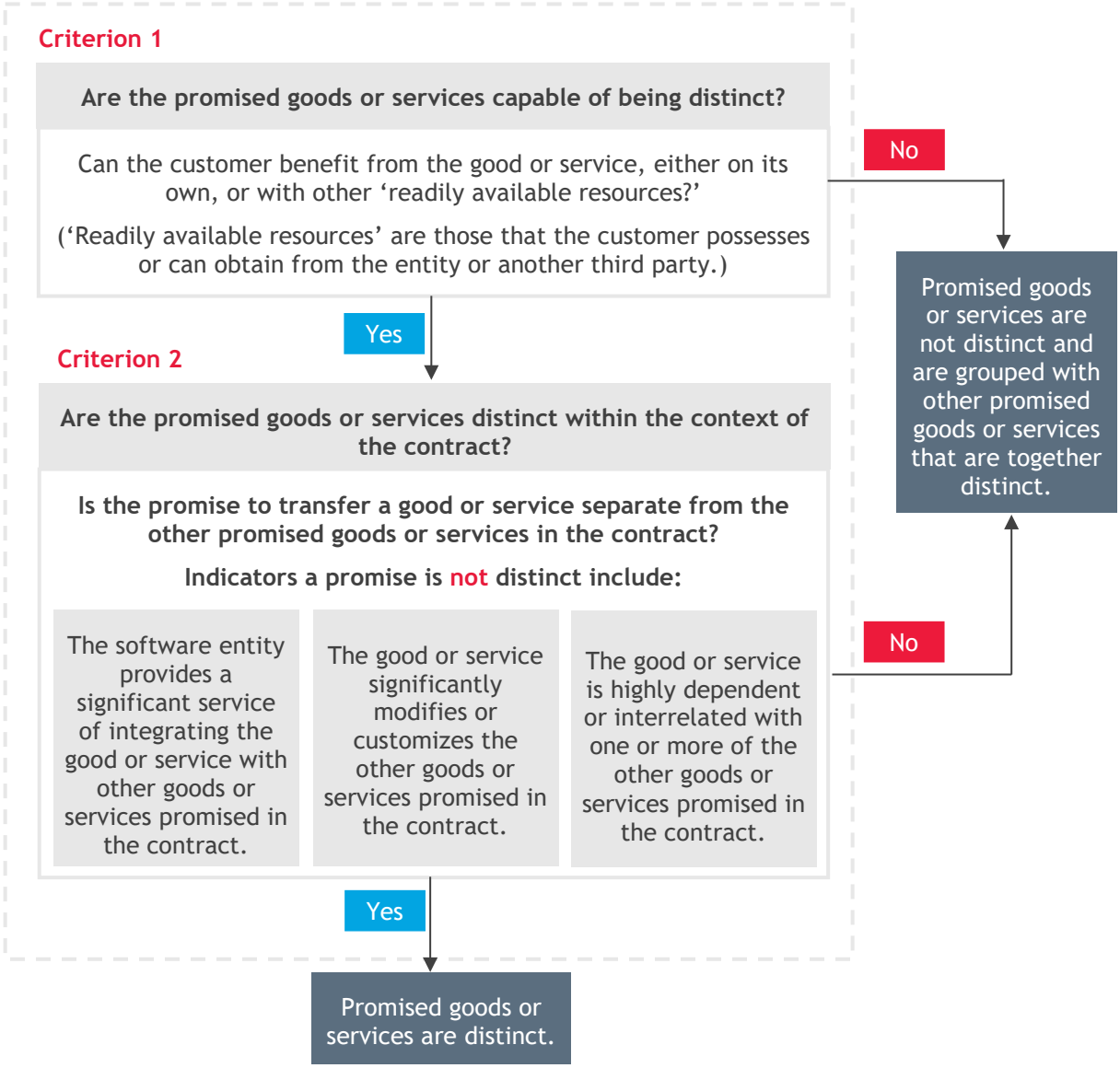
BDO INSIGHTS — IDENTIFYING IMMATERIAL PROMISES IN SOFTWARE ARRANGEMENTS

To determine whether a promise is immaterial in the context of a contract, a software entity must consider the customer's perspective in evaluating the quantitative and qualitative nature of the promise. 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.

ARE THE PROMISED GOODS OR SERVICES DISTINCT?

After identifying the promised goods or services in the contract, which may include licenses, 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 (including a license) 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 an 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 significant judgment and consideration of all facts and circumstances for each individual contract with a customer.

ASC 606 specifies three factors to assist an 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 modify or customize 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:

- Multiple promised goods or services that are outputs themselves
- A combined item(s) that consists of the multiple promised goods or services that are inputs to a combined item(s)

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 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 the following adaptation.

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

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.

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 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.

Therefore, the software developer concludes that there are three performance obligations in the contract:

- Software customization (comprised of the software license and the customized installation service)
- Software updates
- 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 impact the other.

BDO INSIGHTS — IDENTIFICATION OF DISTINCT GOODS OR SERVICES REQUIRES CAREFUL ANALYSIS

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 apply significant judgment to determine whether a promise in a contract is a distinct good or service (and, hence, constitutes a performance obligation) or must be combined ("bundled") with other promises in the

contract to create a single performance obligation. Subtle differences in contractual terms and conditions, unique facts and circumstances, and customer expectations can affect this analysis.

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 below 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

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.

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. This is 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 functionality of the anti-virus software (that is, they permit the software to protect the customer from a significant number of additional viruses that the software did not protect against previously) and 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. Therefore, the software entity accounts for the software license and the when-and-if available updates as a single performance obligation in the contract.

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 INSIGHTS — 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 that additional, albeit limited, scenarios may exist where a software license may be combined with updates provided over time and hence, revenue from the software license would be recognized over time rather than at a point in time. Significant judgment is required to analyze the nature of a software license sold with updates that would substantively change (or maintain) the functionality of the license and hence the benefit derived by the customer during the license term.

We believe the following key factors may be helpful in determining whether a software license and updates are a single performance obligation:

- The frequency of the updates
- Whether the updates are driven by factors within the software entity's control
- How significantly the updates impact the functionality of the software
- How often and how many customers elect not to adopt an update and continue to derive the intended benefit from the software license

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

A software entity provides a cloud-based business productivity and collaboration solution to customers, which 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.

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 an entity to transfer control of additional goods or services to a customer (for example, by requiring the 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 entity's IP and therefore do not define whether the entity satisfies its performance obligation at a point in time or over time and do not create an obligation for the 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

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.

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

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.

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 is integral to the functionality of the product and is thus combined with the product and firmware as a single bundled performance obligation or
- The application 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

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. In addition, 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.

The software entity concludes that 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

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.

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 INSIGHTS — 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 INSIGHTS — SERIES GUIDANCE IS MANDATORY IF IT APPLIES

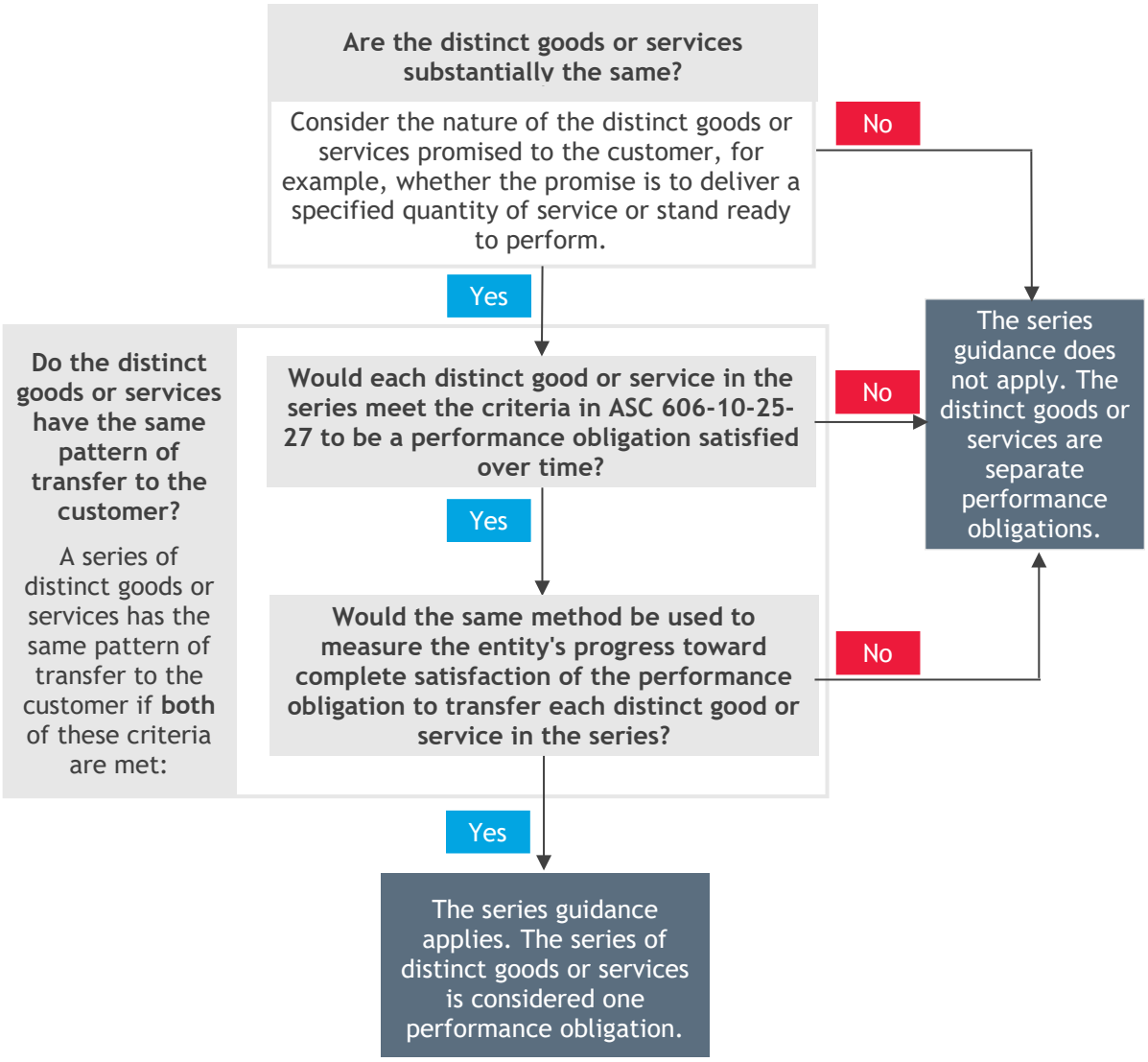
Although the series guidance simplifies the application of ASC 606 in many situations, it is not a practical expedient that software entities have a choice of applying. Rather, it is mandatory if the criteria for its application are met. Therefore, software entities need to consider whether the series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

BDO INSIGHTS — 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 our **Blueprint on revenue recognition** *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 significant 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 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, which 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 should consider 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 INSIGHTS — 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 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 can require significant 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 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 an entity, but the entity must still further develop its good, service, or IP. For example, a software entity's promise to transfer unspecified software upgrades at the entity's discretion, or a pharmaceutical entity's promise to provide when and-if-available updates to previously licensed IP to a drug formula based on advances in R&D.

Type B: Obligations in which the delivery of the underlying good or service is outside the control of the entity and the customer. For example, an entity's promise to remove snow from an airport's runways in exchange for a fixed fee for the year.

Type C: Obligations in which the delivery of the underlying good or service is within the control of the customer. For example, an 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, for example, a health club membership.

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 carefully 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 INSIGHTS — 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.

However, software entities must not default to that revenue recognition pattern for SaaS with usage-based pricing. Rather, a software entity must carefully 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 Example 9 for an illustrative fact pattern.

EXAMPLE 9: SAAS WITH USAGE-BASED PRICING

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.

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

- 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.

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.

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 into 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.

If an option does not provide a material right to the customer, the software entity has made a marketing offer which 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 users being $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 to 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). Prior to 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 INSIGHTS — SOFTWARE LICENSE OR SAAS WITH USAGE-BASED PRICING

A software entity may price its software license or SaaS based on a customer's usage. Determining whether usage-based pricing for a software license or SaaS 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 SaaS 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 Example 9 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. The 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) and the software entity must determine whether the customer option is a material right. If the customer is not required to pay for the SaaS or the payment is not commensurate with the standalone selling price of the SaaS, then the conversion option is a material

right. If however the payment for SaaS is commensurate with its standalone selling price, then the conversion option is a marketing offer (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 SaaS. While some apply the above discussed material right approach to determine whether the customer option is a material right, which 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 SaaS because the 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 SaaS. 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 SaaS is commensurate with the standalone selling price of SaaS, then the conversion option is effectively a marketing offer and not a separate performance obligation under either view because:

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

BDO INSIGHTS — CLOUD CONVERSION

Accounting for cloud conversion rights requires significant 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.

A customer option for cloud conversion may be explicit or implied by the 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 have the expectation at contract inception that it has an option to convert its software license to SaaS 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 covert a license to SaaS, then 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 good or service 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 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 prior to contract expiry. For example, a software entity may sell a software license with a one-year term and provide

[2] EITF 19-B, Issue 2

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, which requires an analysis of:

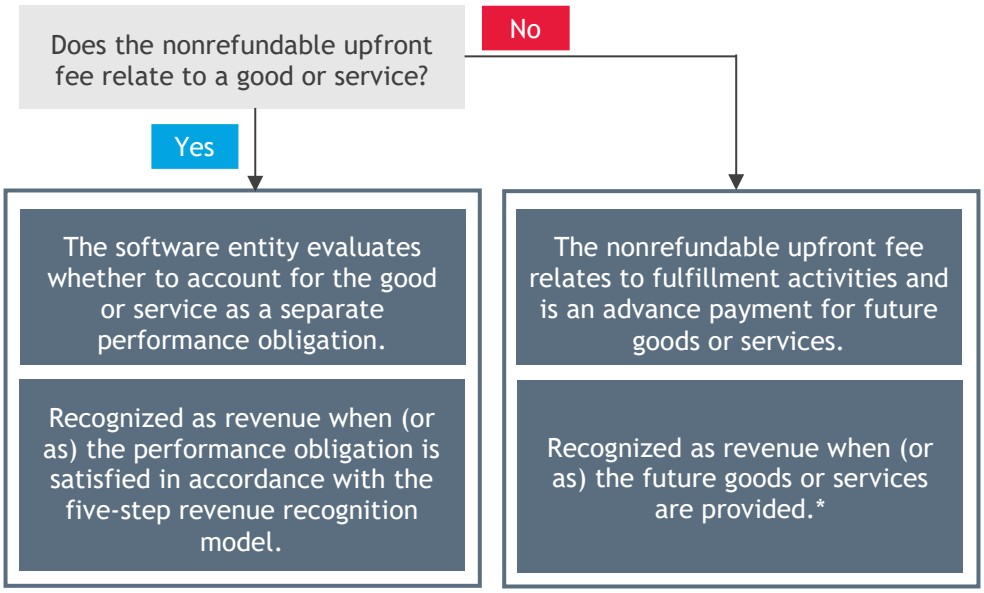
- Whether a performance obligation in the original contract has been satisfied before the modification date
- Whether the modification adds distinct goods or services, that is, separate performance obligations
- Whether the prices of the distinct goods or services are at their standalone selling prices

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 makes a cumulative effect adjustment for the effects of the modification. See our **Blueprint on revenue recognition** for a detailed discussion of the contract modification guidance.

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, which 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 for SaaS) 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 SaaS 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 entity.

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

A software entity enters a contract with a customer to provide SaaS for one year. The contract includes standard terms and conditions. The contract requires the customer to pay a nominal nonrefundable upfront fee in return for the entity 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.

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

A software entity contracts with a customer to provide SaaS 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.

The 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 FOR AN ARRANGEMENT THAT REQUIRES THE INVOLVEMENT OF A THIRD PARTY

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. The standard includes indicators and examples to assist with the analysis. See our **Blueprint on revenue recognition** 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 to the customer

BDO INSIGHTS — 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 INSIGHTS — 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).

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, in order 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 our **Blueprint on revenue recognition**.

Free Trials — Does a Contract Exist?

A software entity may offer certain services (for example, SaaS) 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 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, 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, for example, if a customer can unilaterally terminate the contract by paying a substantive penalty, 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, which is evidenced by the fact that compensation (a termination penalty) would be required to terminate the contract.

BDO INSIGHTS — ANALYZING TERMINATION PENALTIES

ASC 606 does not define the terms penalty or termination penalty. Additionally, ASC 606 does not contain bright lines for determining whether a termination penalty is substantive. In analyzing whether either party must "compensat[e] the other party" 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 prior to the stated term.

Additionally, 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 software license contract will likely not represent a substantive termination penalty.

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 carefully 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 have an impact on the assessment of the performance obligations and revenue recognition.

See the following two examples for illustrations.

EXAMPLE 12: CANCELLABLE FOUR-YEAR CONTRACT FOR SOFTWARE LICENSE WITH SUBSTANTIVE TERMINATION PENALTY

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 right to unilaterally 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 it 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.

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

Consider the same facts in Example 12 except that the customer is not required to pay any termination penalty.

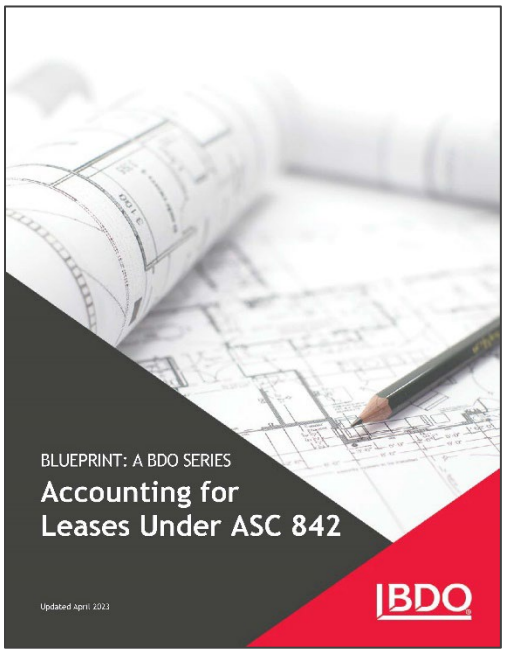
The software entity determines that the contract term is one year because the customer has the unilateral right to terminate the contract at the beginning of each year without paying any termination penalty. The customer has three options to renew the contract annually by not exercising the termination 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 Years 1-4.

APPENDIX A — BDO BLUEPRINTS

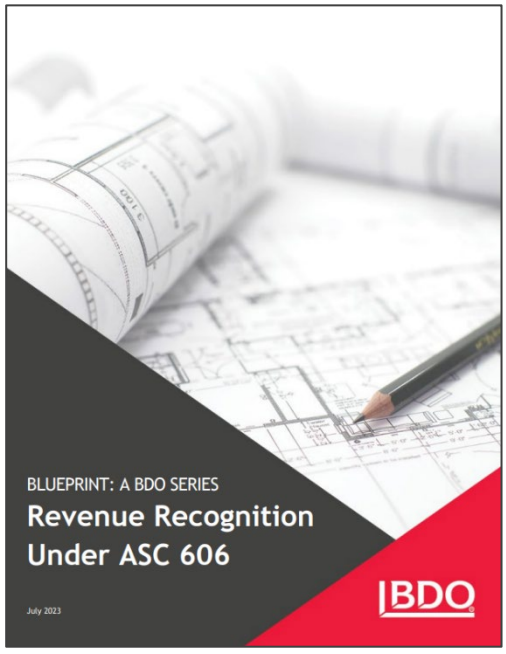
Other publications in BDO's Blueprint series are available on the **BDO Center for Accounting Standards and Reporting Matters**.



Accounting for Leases Under ASC 842

This Blueprint, *Accounting for Leases Under ASC 842*, guides professionals through the application of ASC 842.

The Professional Practice Group updated this Blueprint in April 2023 for FASB amendments to ASC 842 and BDO Insights.



Revenue Recognition Under ASC 606

This Blueprint, *Revenue Recognition Under ASC 606*, guides professionals through the application of ASC 606.

The Professional Practice Group updated this Blueprint in July 2023 for FASB amendments to ASC 606 and BDO Insights.

CONTACTS

BOBBI S. GWINN
Professional Practice Director – Accounting
214-665-0749 / **bgwinn@bdo.com**

ANGELA NEWELL
Professional Practice Partner – Accounting
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**