



Conga Order Management



Table of Contents

| | |
|--|----|
| Release Notes..... | 6 |
| Summer 2020 Release Notes..... | 6 |
| About Order Management..... | 7 |
| Key Terminology..... | 8 |
| What's New in Order Management Documentation..... | 13 |
| Oder Management for Administrators..... | 27 |
| Setting up Order Management..... | 28 |
| Before You Begin..... | 28 |
| Installing Order Management..... | 28 |
| Logging Into Order Management..... | 29 |
| Configuring Order Management Status..... | 30 |
| Configuring Order Management..... | 31 |
| Upgrading Order Management..... | 33 |
| Configuring In-Flight Order Changes and Cancellation..... | 37 |
| To configure In-Flight Order Change and Cancellation settings..... | 37 |
| Configuring Automation for In-Flight Order Changes and Cancellation..... | 39 |
| Configuring Order Workflow Rulesets..... | 40 |
| Configuring an Order Workflow Rule for Split Order..... | 41 |
| Configuring the Direct Order Workflow..... | 41 |
| Configuring the Accept and Configure Products Buttons..... | 42 |
| Changing the Flow on the Configure Products Button..... | 43 |
| Using a Contract Price List for an Order..... | 43 |
| Setting Up Partial Orders from Price Agreements..... | 43 |
| Configuring Price Agreements..... | 44 |
| Configuring the Contract Price List for Orders With Commitment Compliance..... | 48 |
| Setting Up Customer Purchase Orders..... | 49 |
| Configuring Customer Purchase Orders and Purchase Order Items..... | 49 |
| Setting Up Purchase Order Admin..... | 54 |
| Appendices..... | 55 |

| | |
|--|-----|
| Order System Properties | 55 |
| Order Management for Users..... | 58 |
| Logging in to Order Management..... | 60 |
| To log in to Order Management..... | 60 |
| Navigating the Order Management User Interface..... | 61 |
| Working in the Conga User Interface | 62 |
| Understanding the Order Management Lifecycle | 63 |
| Understanding Order Status..... | 63 |
| Capturing Orders..... | 75 |
| Order Management Scope | 76 |
| Creating Direct Orders for Accounts | 76 |
| Cloning an Order..... | 79 |
| Managing In-Flight Order Changes and Cancellation..... | 80 |
| Tracking Order Versions..... | 80 |
| Automating In-Flight Order Changes..... | 81 |
| Amending an Order | 81 |
| Cancelling an Order | 83 |
| Creating Order Workflows to Automate In-Flight Order Changes | 84 |
| Working With Customer Price Agreements..... | 93 |
| Navigating the Price Agreements User Interface..... | 94 |
| Creating Partial Orders from Price Agreements | 95 |
| Splitting an Order | 97 |
| Auto Split on Accept..... | 97 |
| Creating Order Workflows to Automate Split Order | 103 |
| Working With Distributed Order Fulfillment..... | 104 |
| Releasing Orders to Fulfillment Systems..... | 106 |
| Tracking Order Fulfillment..... | 108 |
| Activating an Order | 110 |
| Providing a Legal Entity for Downstream Processes..... | 110 |
| Billing for an Order..... | 111 |
| To create billing schedules on activation of an order | 111 |
| To create billing schedules manually..... | 111 |
| Working with Customer Purchase Orders | 112 |

| | |
|---|-----|
| Understanding Customer Purchase Order Status..... | 115 |
| Managing CPO Validation and Enrichment Rules..... | 116 |
| Creating Customer Purchase Orders | 123 |
| Accepting Customer Purchase Orders User Guide | 124 |
| Amending Customer Purchase Orders User Guide | 125 |
| Cancelling Customer Purchase Orders User Guide..... | 126 |
| Creating Sales Orders from a Customer Purchase Order..... | 127 |
| Intelligent Order Assistant (Max for OM)..... | 129 |
| Order Management (Out-Of-the-Box) Max Flow..... | 129 |
| Order management for SOAP API Developers | 131 |
| Getting Started | 132 |
| API Standards and Development Platforms..... | 132 |
| Field Types..... | 133 |
| Integrating Conga with External Systems..... | 134 |
| API Reference..... | 138 |
| Order Web Service..... | 138 |
| Customer Purchase Order (CPO) Web Service..... | 211 |
| Data Validation Service..... | 224 |
| Scenarios..... | 227 |
| Working with orders in Order Management..... | 227 |
| Order Management Feature by Release..... | 228 |
| Features by Release..... | 228 |
| Community & Learning Center Resources..... | 229 |
| Ready to get started? | 229 |

Learn to manage the life of the generated order before it creates or updates an asset.

Release Notes

Review the latest TurboEngines release notes that describe the system requirements and supported platforms, new features, enhancements, resolved issues, and known issues.

- [Summer 2020 Release Notes](#)

Summer 2020 Release Notes

Unable to render include or excerpt-include. Could not retrieve page.

About Order Management

An order is a document that serves as a confirmation of a purchase created for a customer before delivering goods or services. Conga Order Management allows customer sales and customer support representatives to manage the life of the generated order before it creates or updates an asset. Order Lifecycle Management offers a common, streamlined process for managing orders regardless of the channel used for their creation – direct sales, partner sales, telesales, digital commerce, and Electronic Data Interchange EDI – through the lifecycle of the order through fulfillment.

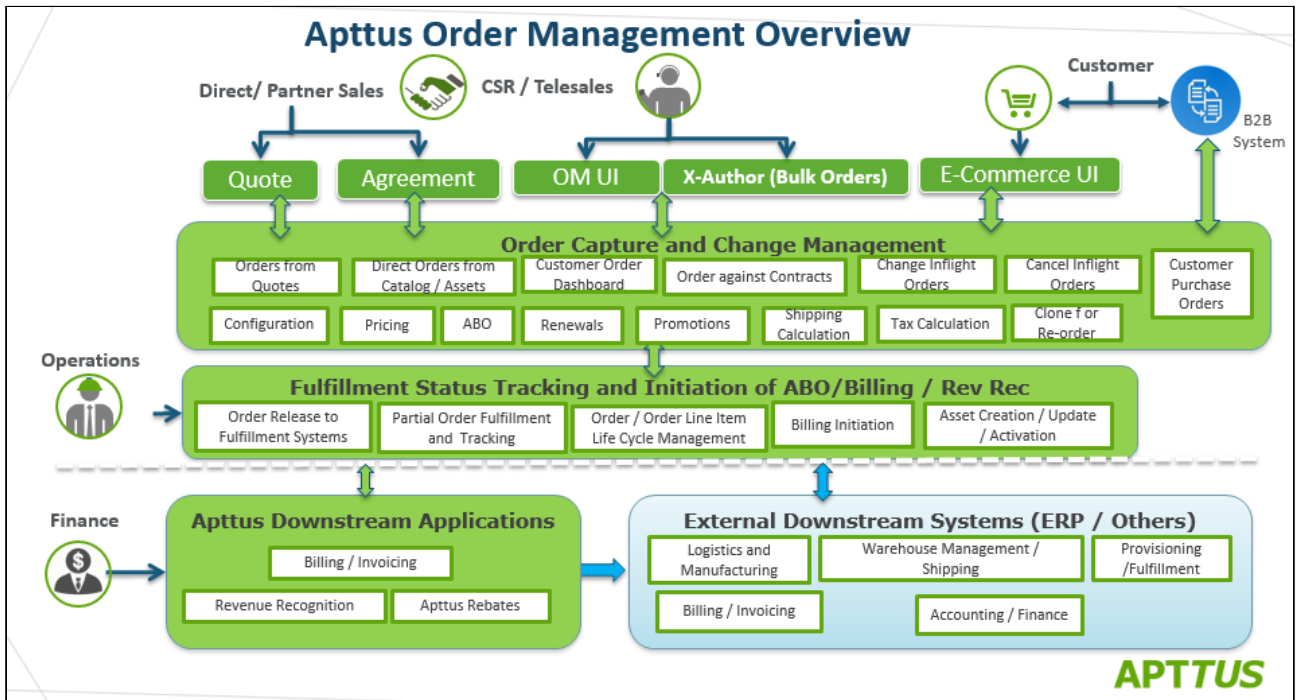
The three main stages of the lifecycle are:

- **Order Capture:** In this stage, users can create direct orders through the Order Management interface, create customer purchase orders from a quote, and create digital commerce orders.
- **Order Change:** In this stage, users can make changes to the configuration, pricing, shipping, promotions, change or cancel in-flight orders, and reorder before the order has created or updated an asset.
- **Order Fulfillment Tracking:** In this stage, the order is released to multiple fulfillment systems and locations. Order fulfillment can be tracked and managed at the order and order line item level.

Conga Order Management allows users to integrate with external systems, including ERP, logistics and manufacturing, accounting and finance, and warehouse management.

Advanced features within Order Management include the design of custom validation and enrichment rules for purchase orders, as well as automation of in-flight order changes.

The following diagram shows the interaction with different entities at various stages in the omni-channel sales process.



Conga Order Management allows a user to perform the following tasks:

- Managing order lifecycle and status
- Creating direct orders
- Creating partial orders from a customer price agreement (Quote/Agreement)
- Managing in-flight order changes and cancellation
- Managing distributed order fulfillment
- Activating an order
- Billing for an order
- Managing customer purchase orders
- Creating partial and split sales orders from customer purchase orders
- Creating data enrichment and validation rules for partial orders

Conga currently supports order creation through direct and partner sales and digital commerce interfaces. From the Order Management interface, customer support representatives can create orders on behalf of customers.

Key Terminology

It is important to understand how terms are used when working with Conga CPQ.

| Term | Description |
|---------------------|--|
| ABO | Asset-based ordering (ABO) functionality enables the customers to manage their existing subscriptions or install base using actions such as change, renew, swap, and terminate. |
| Administrators | Individual responsible for installing, configuring, and maintaining Order Management software, including creating direct orders, customer purchase orders, and managing partial order fulfillment and automation. |
| Assets | Assets define a purchased product or service. An asset is associated with an account. After being processed and fulfilled, the line items associated with new quotes, agreements, or orders result in the creation of new assets, which can then be viewed or managed from the customer's account. |
| Amend Order | An action a user can take from the order record page. You can use Amend Order to create a new version of the Order classified as an amendment with changes to the field values of the order. |
| Attributes | Features of a product, such as color, size, weight, and more. |
| Bundled Products | A combination of standalone products that offer added value to the customer while increasing overall sales. |
| Cart | A product and pricing view for the user to review all configuration and pricing information at a glance. |
| Catalog | A view that allows hierarchical categorization of products for users to search through and add to their configuration. |
| Clone | To replicate a field, record, template, etc. |
| Clone Order | An action a user can take from the order record page which creates a copy of the order record. |
| Contract Price List | A price list that helps the customer keep track of specific price agreements as applicable for that account. |

| Term | Description |
|------------------------------|---|
| Customer Purchase Order | A customer document that represents the initial offer of negotiated types, quantities, and prices for products or services. Customer Purchase Orders can be created by sales users or administrators as sales orders for long-term contracts. |
| In-Flight Order | The flow status of an order that is in the process of fulfillment. |
| Order | A document that serves as a confirmation of a purchase created for a customer before delivering goods or services. The Order object is used in Order Management to capture and track orders created from quotes, agreements, or other sources, such as customer purchase orders. |
| Order Fulfillment Line Items | A line item on the order record that comprises order fulfillment information for the record. Order fulfillment line items are displayed in the Order Fulfillment Line Items related list. |
| Order Line Item | A line item on the order record corresponding to a product, bundle, or service that is part of the order. Order line items are displayed in the Order Line Items related list. |
| Order Line Item Status | A column on the Order Line Items related list. This status refers to the current status of the corresponding line item. The possible status values are: <i>PartiallyFulfilled, InFulfilment, Pending, Draft, Accepted, In-Amendment, BeingAmended, Superseded, Activated, Pending Cancellation, BeingCancelled, Cancelled, and Fulfilled.</i> |
| Order Number | The unique ID of an order. |
| Order Source | A field on the order record that specifies the source from which an order is created. For example: Account, Quote, or an Agreement. |
| Order Status | A field on the order record that denotes the status of an order that you have created or updated. The possible status values are: <i>PartiallyFulfilled, InFulfilment, Pending, Draft, Accepted, In-Amendment, BeingAmended, Superseded, Activated, PendingCancellation, BeingCancelled, Cancelled, and Fulfilled.</i> |

| Term | Description |
|-------------------------|--|
| Partial Order | An order placed against a price agreement or customer purchase order for a partial amount of the committed quantities captured for an order. |
| Price Agreement | An agreement associated with a generated contract price list that represents the negotiated pricing and quantity commitments for one or more configurations of products and services. |
| Price Lists | Containers of items that are grouped in a price list. A price list controls which products are visible to the end user. A price list contains several price list items; each linked to a product. A product can be set up with one or more price list items. |
| Price Matrix (Matrices) | These are an advanced pricing concept used to define tiered pricing paradigms, or complex pricing structures with multiple criteria. Common examples are pricing tiers for a product based on user count or particular customer or transactional dimensions. |
| Price Rule | Represents a single rule in a price ruleset. |
| Price Rule Sets | Price Rulesets are a mechanism to allow particular families, categories or groupings of products to have either line item pricing adjustments applied or summary pricing adjustments applied. Typical examples of these are volume discounting rules or promotional pricing rules. |
| Product | A product or service that can be sold on its own as a standalone item, an option of other products, or as part of a bundled product. |
| Product Attribute Group | Represents a product attribute group that contains attributes shared by products. |
| Product Attribute Value | Represents the attribute values for a product class. For example, color has attribute values such as red, green, blue, orange and so on. |
| Product Group | A logical grouping of one or more product records. This construct allows you to create combinations of products with similar characteristics/qualities for use in a Rule. |

| Term | Description |
|---------------------|--|
| Quote or Proposal | A structured definition of a prospective sale that contains product configurations, pricing, and customer opportunity information. |
| Standalone Products | Refers to a device or software that is self-contained, one that does not require any other devices or software to function. |

What's New in Order Management Documentation

The following table lists changes in the documentation to support each release.

Summer '20

| Document | Topic | Description |
|-------------------|--|---|
| Summer 2020 REV A | Splitting an Order | Updated the description of the Manual Split. |
| Summer 2020 | Configuring Order Management Custom Settings | Updated the Order System Properties table with new custom settings (Split Order Criteria Fields & Split Order Threshold). |
| | Upgrading Apttus Order Management | Updated topic with the specific upgrade information. |
| | Configuring Order workflow Rule for Split Order | New Topic describing how to add Split Order to the Action field in the Order Workflow Rule Detail page. |
| | Splitting an Order | New Topic. |
| | Creating Order Workflows to Automate Split Order | New Topic. |
| | Integrating Apttus with External Systems | New Topic. |
| | Splitting an Order | New Topic. |
| | Splitting an Order by criteria | New Topic. |

| Document | Topic | Description |
|----------|--|----------------|
| | Splitting an Order by threshold | New Topic. |
| | Creating an Order | Updated Topic. |
| | Creating a Cart for an Order | Updated Topic. |
| | Synchronizing a Cart to an Order | Updated Topic. |
| | Order Web Service | Updated Topic. |
| | Accepting an Order | Updated Topic. |
| | Cancelling an Order | Updated Topic. |
| | Reverting a Cancelled Order | Updated Topic. |
| | Cloning an Order | Updated Topic. |
| | Customer Purchase Order (CPO) Web Service | Updated Topic. |
| | Accepting Customer Purchase Orders | Updated Topic. |
| | Amending Customer Purchase Orders | Updated Topic. |
| | Reverting Amended Customer Purchase Orders | Updated Topic. |
| | Cancelling Customer Purchase Orders | Updated Topic. |

| Document | Topic | Description |
|----------|--|----------------|
| | Reverting Cancelled Customer Purchase Orders | Updated Topic. |
| | Creating Orders from PO | Updated Topic. |

Spring '20

| Document | Topic | Description |
|------------|--|--|
| Spring '20 | Upgrading Apttus Order Management | Updated topic with specific upgrade information and cross-reference to CPQ topic. |
| | Displaying Order Fulfillment Line Items for an Order | Updated task topic steps. |
| | Configuring Order Workflow Rulesets | New topic describing how to add the required buttons for configuring workflow rulesets. |
| | About Order Management | Updated topic with additional key terminology. |
| | Creating Order Workflows to Automate In-Flight Order Changes | Updated topic with additional contextual information and tasks. Added a workflow ruleset use case. |
| | Creating Data Enrichment and Validation Rules | Updated topic to include a detailed use case. |
| | Order Management for Users | Updated topic. |
| | About Order Management | Updated topic. |
| | Getting Started | Renamed topic from "Overview." Updated topic. |

| Document | Topic | Description |
|----------|--|---|
| | API Supported Packages | Deleted topic. |
| | API Reference | Updated topic. |
| | Creating an Order | Updated topic. |
| | Creating a Cart for an Order | Updated topic. |
| | Synchronizing a Cart to an Order | Updated topic. |
| | Accepting an Order | Updated topic. |
| | In-Flight Order Changes and Cancellation | New contextual parent topic for Amend Order, Undo Amend Order, and Cancel Order APIs. |
| | Amending an Order using API | Updated topic. |
| | Reverting an Amendment to an Order | New topic. Content is taken from "Amending an Order." |
| | Cancelling an Order using API | Updated topic. |
| | Reverting a Cancelled Order | New topic. Content is taken from "Cancelling an Order." |
| | Cloning an Order using SOAP API | Updated topic. |
| | Fetching Price Agreements | Updated topic. |
| | Fetching Price Agreement Items | Updated topic. |

| Document | Topic | Description |
|----------|--|-------------------------------|
| | Creating a Partial Order from a Quote or Agreement | Renamed topic. Updated topic. |
| | Creating a Direct Order | Updated topic. |
| | Executing Order Workflow | Updated topic. |
| | Customer Purchase Order (CPO) Web Service | Updated topic. |
| | Accepting Customer Purchase Orders | Updated topic. |
| | Amending Customer Purchase Orders | Updated topic. |
| | Reverting Amended Customer Purchase Orders | Updated topic. |
| | Cancelling Customer Purchase Orders | Updated topic. |
| | Reverting Cancelled Customer Purchase Orders | Updated topic. |
| | Creating Orders from PO | Updated topic. |

Winter '19

| Document | Topic | Description |
|--------------------|---|---------------|
| Winter 2019 (1903) | Order Management for Administrators | Update topic. |

| Document | Topic | Description |
|----------|--|--|
| | About Order Management | Updated topic. |
| | Getting Started | Deleted topic. |
| | Setting Up Order Management | New topic to replace deleted topic "Getting Started." Updated topic content. |
| | Installing Order Management | Updated topic. Moved topic under "Setting Up Order Management." |
| | Logging in to Order Management | New topic. |
| | Configuring Order Management | New topic. |
| | Configuring Order Management Custom Settings | New topic. |
| | Configuring Order Management Status | New topic. |
| | Upgrading Apttus Order Management | New topic. |
| | Partial Order Fulfillment | Deleted topic. |
| | Displaying Order Fulfillment Line Items for an Order | New topic. |
| | Configuring In-Flight Order Changes and Cancellation | Topic renamed from "In-Flight Order Changes." Updated topic. Merged content that was previously part of the deleted topic "Automation of In-Flight Order Changes." |

| Document | Topic | Description |
|----------|---|---|
| | Automation of In-Flight Order Changes | Deleted topic. |
| | Configuring the Direct Order Workflow | Topic renamed from "Direct Order Configurations." Updated topic. |
| | Setting Up Partial Orders from Price Agreements | Topic renamed from "Generating Partial Orders from Quote or Agreement." Update topic. |
| | Configuring the Contract Price List for Orders With Commitment Compliance | Topic renamed from "Generating Contract Price List." Updated topic. |
| | Setting Up Customer Purchase Orders | Topic renamed from "Create Customer Purchase Order and Customer Purchase Order Items." Updated topic. |
| | Setting Up Purchase Order Admin | New topic for configuring the PO Admin (Admin UI for creating Enrichment and Validation Rules for Purchase Orders). |
| | About This Guide | Updated topic. |
| | About Order Management | Updated topic. |
| | Order Management in the Quote-to-Cash Process | Deleted topic. |
| | Getting Started | Deleted topic. |
| | Logging in to Order Management | New topic (replaced "Getting Started"). |
| | Navigating the Order Management User Interface | New topic. |
| | Understanding the Order Management Lifecycle | Topic renamed from "Getting Around with Order Lifecycle and Status." Updated topic. |

| Document | Topic | Description |
|----------|---|---|
| | Order Status | Deleted topic. |
| | Order Line Item Status | Deleted topic. |
| | Understanding Order Status | New topic (replaced "Order Status" and "Order Line Item Status"). |
| | Capturing Orders | Updated topic. |
| | Creating Orders from Quotes | Deleted topic. This information refers to a CPQ process. |
| | Creating E-Commerce Orders | Deleted topic. This information is redundant and covered in the API Guide. |
| | Creating Direct Orders for Accounts | Updated topic. |
| | Configuring and Pricing Orders | Deleted topic. Note on configuring and pricing added to Creating Direct Orders for Accounts . |
| | Finalizing Configurations | Deleted topic. |
| | Confirming Orders | Deleted topic. Content added to parent topic. |
| | Legal Entity on Order Line Item | Deleted topic. Content added to Activating an Order . |
| | Cloning an Order | Updated topic. |
| | Managing In-Flight Order Changes and Cancellation | Updated topic. |
| | Amending an Order | Renamed topic from "Changing an In-flight Order." Updated topic. |

| Document | Topic | Description |
|----------|--|---|
| | New UI for Amend Order | Deleted topic. |
| | Cancelling an Order | Updated topic. |
| | Creating Order Workflows to Automate In-Flight Order Changes | Renamed and updated topic. |
| | Versioning of an Order | Deleted topic. Content added to "Managing In-Flight Order Changes." |
| | Sample Scenarios for Amending Orders Using Workflow Rules | New topic. Content moved from Admin Guide. |
| | Order Management Workflows | Deleted topic. |
| | Agreement to Order | Deleted topic. This is a CLM task. |
| | Quote to Order | Deleted topic. This is a CPQ task. |
| | Direct Order Configuration | Deleted topic. Content added to Creating Direct Orders for Accounts and Releasing Orders to Fulfillment Systems . |
| | Asset Manager | Deleted topic. |
| | Total Order Amount on Order Header | Deleted topic. Content added to Capturing Orders . |
| | Working With Customer Price Agreements | Renamed and updated topic. |
| | Creating Partial Orders from Price Agreements | Renamed topic. |
| | View List of Quotes/Agreements | Deleted topic. |

| Document | Topic | Description |
|----------|---|---|
| | View List of Line Agreements from Price Agreements | Deleted topic. |
| | Rollup Order Line Item Information to Agreement Line Item | Deleted topic. Content added to Creating Partial Orders from Price Agreements . |
| | Rollup Order Line Item Information to Quote Line Item | Deleted topic. Content added to Creating Partial Orders from Price Agreements . |
| | Creating Partial Order from Agreement Enhancements | Deleted topic. Content added to Working With Customer Price Agreements . |
| | Working With Distributed Order Fulfillment | Updated topic. |
| | Releasing Orders to Fulfillment Systems | Renamed topic. Updated topic. |
| | Tracking Order Fulfillment | Renamed topic. Updated topic. |
| | Order Fulfillment | Deleted topic. Content added to Releasing Orders to Fulfillment Systems . |
| | Order Fulfillment Line Items | Deleted topic. Content added to Releasing Orders to Fulfillment Systems . |
| | Order Fulfillment Line Item Lifecycle | Deleted topic. Content added to Releasing Orders to Fulfillment Systems . |
| | Activating an Order | Updated topic. |
| | Billing for an Order | Updated topic. |

| Document | Topic | Description |
|----------|--|---|
| | Working with Customer Purchase Orders | Updated topic. |
| | Understanding Customer Purchase Order Status | Renamed topic from "Customer Purchase Order and PO Item Status Mapping." Updated topic. |
| | Managing CPO Validation and Enrichment Rules | Updated topic. |
| | Creating Data Enrichment and Validation Rules | Renamed topic from "Creating Rulesets." Updated topic. |
| | Creating Rulesets | Deleted topic. Content added to Creating Data Enrichment and Validation Rules |
| | Creating Rules | Deleted topic. Content added to Creating Data Enrichment and Validation Rules |
| | Mitigating Data Validation Issues | Updated topic. |
| | Creating Customer Purchase Orders | Updated topic. |
| | Accepting Customer Purchase Orders | Renamed topic from "Confirming or Accepting Draft Customer Purchase Orders." Updated topic. |
| | Amending Customer Purchase Orders | Updated topic. |
| | Cancelling Customer Purchase Orders | Updated topic. |
| | Creating Sales Orders from a Customer Purchase Order | Renamed topic from "Managing Customer PO." Updated topic. |

| Document | Topic | Description |
|----------|--------------------------------|--|
| | Creating a Single Order | Updated topic. |
| | Creating Multiple Sales Orders | Updated topic. |
| | Creating Partial Orders | Updated topic. |
| | About Order Management | Updated topic. |
| | Overview | Updated topic. |
| | Document Setup | Deleted topic. Will be added back when scenarios are documented. |
| | API Supported Packages | Updated topic. Updated table to reflect the latest package versions. |

Summer '19

| Document | Topic | Description |
|--------------------|---|--|
| Summer 2019 (1902) | Automation of In-Flight Order Changes | New topic. New feature for this release. |
| | Creating Order Workflow to Automate In-flight Order Changes | New topic. New feature for this release. |
| | Executing Order Workflow | New topic. New feature for this release. |

Spring '19

| Document | Topic | Description |
|--------------------|------------|---|
| Spring 2019 (1901) | No updates | No new topics are added for this release. The guide is updated to reflect product name changes. |

Winter '18

| Document | Topic | Description |
|--------------------|--|--|
| Winter 2018 (1803) | Total Order Amount on Order Header | New topic. New feature for this release. |
| | Legal Entity on Order Line Item | New topic. New feature for this release. |
| | Creating Partial Order from Agreement Enhancements | New topic. New feature for this release. |

Summer '18

| Document | Topic | Description |
|--------------------|--|--|
| Summer 2018 (1802) | Generating Partial Orders from Quote or Agreement | Updated admin level settings for generating partial orders from agreements/quotes. |
| | Create Customer Purchase Order and Customer Purchase Order Items | New topic. New feature for this release. |
| | Managing CPO Validation and Enrichment Rules | New topic. New feature for this release |
| | Mitigating Data Validation Issues | New topic. New feature for this release |
| | Managing Customer PO | New topic. New feature for this release |
| | Order Fulfillment Line Item Lifecycle | New topic. New feature for this release |
| | Creating Customer Purchase Orders | New topic. New feature for this release |

| Document | Topic | Description |
|----------|--|---|
| | Creating Partial orders from Customer Price Agreement (Quote/Contract) | New topic. New feature for this release |
| | Intelligent Order Assistant (Max for OM) | New topic. New feature for this release |

Oder Management for Administrators

This guide is designed to provide administrators with information on configuring Conga Order Management for use by sales and customer support representatives. This guide covers the most common use cases for Conga Order Management Administration and assumes a level of familiarity with basic Salesforce and Conga CPQ.

| Topic | Description |
|------------------|--|
| What's Covered | This guide walks the Order Management Administrators through the entire process of installing and configuring Conga Order Management. It provides conceptual information, step-by-step instructions, and use cases for the administration tasks provided by Conga Order Management. |
| Primary Audience | Order Management Administrators. |
| IT Environment | Refer to the latest <i>Order Management Release Notes</i> for information on System Requirements and Supported Platforms. |
| Updates | For a comprehensive list of updates to this guide for each release, see the What's New in Order Management Documentation topic. |
| Other Resources | <ul style="list-style-type: none"> • Conga Order Management Summer 2020 User Guide: Refer to this guide for installing and setting up Order Management in your organization. • Conga Order Management Summer 2020 Release Notes: Refer to this guide for the new feature, enhancements, resolved, and known issues. • Conga Order Management Summer 2020 SOAP API Guide: Refer to this guide for documentation of public-facing SOAP APIs for Order Management. • Conga CPQ Administrator Guide: Refer to this guide for setting up products, price lists, and constraint rules. |

This guide describes the following tasks:

- Installing Order Management
- Configuring Order Management settings
- Enabling and configuring settings for In-Flight Orders
- Automating In-Flight Order changes
- Configuring Direct Order settings (CPQ)
- Configuring settings for capturing orders from Price Agreements
- Configuring quotes and agreements to use a Contract Price List for Price Agreement orders

- Configuring Customer Purchase Order and Customer Purchase Order Item settings
- Configuring Custom Purchase Order Admin settings

Before using Order Management, you must be familiar with the following:

- Basic Salesforce administration knowledge
- Conga CPQ and Conga CLM administration
- Salesforce and Conga terms and definitions

Select one of the following topics for more information:

- [Setting up Order Management](#)
- [Configuring In-Flight Order Changes and Cancellation](#)
- [Configuring the Direct Order Workflow](#)
- [Setting Up Partial Orders from Price Agreements](#)
- [Setting Up Customer Purchase Orders](#)
- [Appendices](#)

Setting up Order Management

Conga Order Management setup requires you to install the Order Management package on Salesforce and configure a custom setting for Order Management. The following topics provide step-by-step instructions for setting up Order Management.

Before You Begin

Before configuring Order Management settings, make sure you are familiar with custom and admin settings for Conga Configure-Price-Quote (CPQ). Refer to the *CPQ Administrator Guide* for details.

Installing Order Management

Refer to the latest *Order Management Release Notes* for information on the packages required to install Conga Order Management.

Prerequisites

To use Conga Order Management, you must perform the following tasks prior to installing the Order Management package:

1. **Enable Salesforce CRM Content:** Refer to the *CPQ Administrator Guide* for steps to perform this task.
2. **Install other Conga packages:** Conga Order Management requires packages to be installed for Conga CPQ, as well as packages for any other Conga applications (such as Approvals, Contract Lifecycle Management, and so on) that you plan to use in conjunction with Order Management. Refer to the latest *Order Management Release Notes* and the *CPQ Administrator Guide* for which packages are required.


i Conga recommends downloading and upgrading Conga packages in a Salesforce sandbox **before** installing them in your production environment. For information on installing and upgrading in a sandbox, please contact Conga Support before you install any packages.

To install the Order Management package

1. Go to the **Resources > Install Center** tab on the [Conga Community Portal](#).
2. In the **My Packages** navigation link, click **Conga Order Management**.
3. Click **Install Now**.
4. Select the environment where you want to install the package:
 - Click **Install in Production** to install the packages in your production environment.
 - Click **Install in Sandbox** to install the package in your sandbox.
5. Log in to Salesforce.
6. On the Upgrade page, enter the password provided by Conga.
7. Conga recommends that you select **Install for All Users**.
8. If you want to **Install for Specific Profiles**, you must define the access level for all profiles. Select from one of the following options:
 - **No Access:** This is the default setting. Apply this access level to disable all object permissions.
 - **Full Access:** Apply this access level to assign users permissions to Read, Create, Edit, Delete, View All, and Modify All for all objects in the Order Management package.
9. Click **Set**.
10. Click **Upgrade**.

Logging Into Order Management

Log in to your [Salesforce.com](#) org to access Conga Order Management.

 Do not use the Back button on your browser when using Order Management.

Before logging in to Order Management, make sure you meet the following criteria:

- You have installed all required Order Management packages (included Conga CPQ packages, and packages for other integrated Conga applications).
- You have administrative privileges.
- You have login credentials provided by Conga.

To log in to Order Management

1. Go to <https://salesforce.com/>.

Or

If your organization is using a sandbox or test environment to access Conga Order Management (for example, if you are doing user acceptance testing), go to <https://test.salesforce.com/> instead.

2. From the toolbar at the top of the page, click **Login**. The login page opens.
3. Enter your user name and password, and click **Log in**.
4. Navigate to the Conga Order Management:
 - In Salesforce Classic: Click the App Menu and select **Conga Order Management**.
 - In Salesforce Lightning Experience: Click the App Launcher and select **Conga Order Management**.

Configuring Order Management Status

To ensure that the Order Management flow assigns the correct status for orders at different stages of the lifecycle, you must add and configure all applicable statuses for the Order and Order Line Item objects.

To configure Order object status

1. From the Order tab, go to **Quick Access > View Fields**.
2. From Custom Fields & Relationships, click **Status**.
3. Click **New**. The Status Picklist Values page is displayed.
 - a. In the picklist entry field, enter the following status values: *Pending, Activated, Fulfilled, Cancelled, Superseded, Draft, Partially Fulfilled, In Fulfillment, In*

Amendment, Being Amended, Pending Cancellation, Being Cancelled, Processing, and Completed.

4. Click **Save**.
5. Next to the Draft status, click **Edit**.
6. From the Picklist Edit page click the check box **Make this value default for the master picklist**.
7. Click **Save**.

To configure Order Line Item object status

1. Go to **Setup > Objects > Order Line Item**.
2. From Custom Fields & Relationships, click **Status**.
3. Click **New**. The Status Picklist Values page is displayed.
4. In the picklist entry field, enter the following status values: *Pending, Activated, Fulfilled, Cancelled, Superseded, Draft, Partially Fulfilled, In Fulfillment, In Amendment, Being Amended, and Pending Cancellation*.
5. Click **Save**.
6. Next to the Draft status, click **Edit**.
7. From the Picklist Edit page, select the check box **Make this value default for the master picklist**.
8. Click **Save**.

For detailed descriptions of Order and Order Line Item statuses, refer to the topic "Understanding Order Status" in the *Order Management User Guide*.

Configuring Order Management

Post-installation settings are part of your specific business requirements, meaning that your implementation of Conga Order Management will work out-of-the-box, but you can further customize the implementation using custom settings.

Refer to the topic "Recommended Post-Installation Configuration" in the *CPQ Administrator Guide* for detailed post-installation configuration steps for Conga CPQ (required to support Order Management).

Configuring Order Management Custom Settings

Use Order Management custom settings to configure various features of Order Management.

To access Order custom settings, go to **Setup > Custom Settings > Order System Properties**.

Configure the following Order Management custom settings for your organization.

| Setting | Description |
|--|---|
| Initiate Billing On Order Activation | Indicates whether billing should be initiated on order activation. |
| Create Asset On Order Activation | Indicates whether asset creation is delayed until order activation. |
| Enable Inflight Changes And Cancellation | Indicates whether inflight order changes and cancellation are enabled. |
| Agreement Fields For Partial Order2 | Enter the list of field API names from the agreement object for display when creating partial orders. Each field may be separated by a comma or a new line. |
| Agreement Fields For Partial Order | Enter the list of field API names from the agreement object for display when creating partial orders. Each field may be separated by a comma or a new line. |
| Agreement Item Fields For Partial Order2 | Enter the list of field API names from the agreement line item object for display when creating partial orders. Each field may be separated by a comma or a new line. |
| Agreement Item Fields For Partial Order | Enter the list of field API names from the agreement line item object for display when creating partial orders. Each field may be separated by a comma or a new line. |
| Quote Fields For Partial Order2 | Enter the list of field API names from the quote/proposal object for display when creating partial orders. Each field may be separated by a comma or a new line. |
| Quote Fields For Partial Order | Enter the list of field API names from the quote/proposal object for display when creating partial orders. Each field may be separated by a comma or a new line. |
| Quote Item Fields For Partial Order2 | Enter the list of field API names from the proposal line item object for display when creating partial orders. Each field may be separated by a comma or a new line. |

| Setting | Description |
|-------------------------------------|--|
| Quote Item Fields For Partial Order | Enter the list of field API names from the proposal line item object for display when creating partial orders. Each field may be separated by a comma or a new line. |
| Split Order Criteria Fields | Enter the list of field API names from Order line item objects to split items added during a large order process into multiple orders. Separate each field with a comma or a new line. |
| Split Order Threshold | Indicates the maximum number of a bundle or standalone products (represented as line numbers) that may be added to a split order. Defaults to 100. |

Displaying Order Fulfillment Line Items for an Order

By default, the Order Fulfillment Line Items related list is not displayed on the Order page. Perform the following task to add the related list to the Order page layout.

To display the Order Fulfillment and its line Items for an order

1. On the **Order Detail** page, go to **Edit Layout**.
2. In the **Order Layout**, select **Related Lists**.
3. Drag-and-drop the **Order Fulfillment Line Items** related list onto the page layout.
4. Click **Save**.

For details about Partial Order Fulfillment, refer to *Order Management User Guide*.

Upgrading Order Management

This section provides information for upgrading Conga Order Management to the latest version from the previous release.

i If you have not installed Conga Order Management, you can contact Conga Support to request an installation link, then perform the standard installation as described in [Installing Order Management](#).

i You must install the Conga Base Library package first when you upgrade Conga Order Management to Spring '21.

Preparing for Upgrade

Before you upgrade to Order Management to Spring '21 release, you must ensure the following:

- You have referred to the [Order Management Features by Release](#) to learn about any new features.
- You have [supported platforms and system requirements](#).
- You have access to the **Install Center** on the [Apttus Community Portal](#).
- You have administrator privileges for your Salesforce org.
- You need not back up your configurations. All configurations you performed since you installed the existing release will remain intact after the upgrade.

Upgrading to Order Management Spring '21 release

This section describes step-by-step instructions to upgrade from Summer 2020 to Spring'21 release.

Upgrading to Conga Order Management Spring '21 release.

1. Go to **Setup > Installed Packages** and ensure that your current Salesforce org has the following Summer 2020 packages installed.

| Product | Version Name Version Number |
|--|------------------------------------|
| Apttus Order Management | 1.0.0000 1.0 |
| Apttus Approvals Management (Required if you are using Approvals) | 11.1.0136 11.136 |
| Apttus Base Package | 1.1.77 1.77 |
| Apttus Configuration & Pricing | 12.1.1751 12.1751 Summer 2020 |
| Apttus Contract Management | 11.1.0532 11.532 |
| Apttus Contract-Configuration Integration (Required if you are using CLM) | 12.0.0130 12.130 |

| Product | Version Name Version Number |
|--|-------------------------------|
| Apttus CPQ API | 12.1.0101 12.101 |
| Apttus CPQ Admin (Required if you are using CPQ Admin Console) | 12.1.61 12.61 |
| Apttus CPQ Approvals Management (Required if you are using Approvals on CPQ objects) | 11.0.0016 11.16 |
| Apttus Promotion Management (Required if you are using Promotions) | 1.0.0000 1.0 |
| Apttus Proposal Management (Required if you are using Proposal Management) | 10.1.0221 10.221 |
| Apttus Quote/Proposal Approvals Management | 6.5.0004 6.4 |
| Apttus Quote/Proposal-Asset Integration | 6.5.0014 6.14 |
| Apttus Quote/Proposal-Contract Integration | 9.1.0057 9.57 |
| Apttus Quote/Proposal-Configuration Integration (Required if you are using CPQ and Proposal Management) | 12.1.0325 12.325 |

2. Ensure that you have the following packages and dependent packages to upgrade to Spring '21 release. These packages are required to utilize the new features and enhancements of Spring '21.

| Product | Latest Certified Version (Version Name Version Number) |
|---|---|
| Conga Order Management | 1.0.0000 1.0 |
| Conga Approvals Management (Required if you are using Approvals) | 12.0.0221 12.221 |

| Product | Latest Certified Version <i>(Version Name / Version Number)</i> |
|---|---|
| Conga Base Library | 2.0.148 2.148 |
| Conga Configuration & Pricing | 13.0.1882 13.1882 |
| Conga Contract Lifecycle Management | 12.0.0605 12.605 |
| Conga Contract-Configuration Integration (Required if you are using CLM) | 12.0.0136 12.136 |
| Conga CPQ API | 13.0.0120 13.120 |
| Conga CPQ Setup (Required if you are using CPQ Admin Console) | 13.0.97 13.97 |
| Conga CPQ Approvals Management (Required if you are using Approvals on CPQ objects) | 12.0.0020 12.20 |
| Conga Promotion Management (Required if you are using Promotions) | 1.0.0000 1.0 |
| Conga Proposal Management (Required if you are using Proposal Management) | 11.0.0242 11.242 |
| Conga Quote Approvals | 6.5.0004 6.4 |
| Conga Quote Asset Integration | 6.5.0014 6.14 |
| Conga Quote CLM Integration | 10.0.0069 10.69 |
| Conga Quote/Proposal-Configuration Integration (Required if you are using CPQ and Proposal Management) | 13.0.0354 13.354 |

3. Perform the upgrade. The upgrade procedure is the same as the installation procedure.

4. After the upgrade is complete, perform the post-upgrade tasks.

Performing the Post-Upgrade Tasks

Refer to the topic [Upgrading CPQ](#) in the *CPQ Administrator Guide* for information on required post-upgrade tasks.

Configuring In-Flight Order Changes and Cancellation

An In-Flight Order refers to an order that has been confirmed but has not yet reached fulfillment. By default, Conga Order Management does not have this functionality enabled. You can also configure Workflow Rulesets to automate the execution of workflows for In-Flight Orders Changes and Cancellation.

To enable In-Flight Order Changes and Cancellation, you must configure and make additions or changes to system properties, custom and object settings, and page layouts.

To configure In-Flight Order Change and Cancellation settings

Perform the tasks in the following table in the order described:

| Configuration | Description |
|--|---|
| 1. Enable In-Flight Changes and Cancellation | <ol style="list-style-type: none"> 1. Go to Setup > Custom Settings > Order System Properties. 2. Click Manage. 3. Click the Edit link to the left of System Properties. 4. Click the checkbox next to Enable Inflight Changes and Cancellation. 5. Click the checkbox next to Create Asset on Order Activation? 6. Click Save. |

| Configuration | Description |
|---|--|
| <p>2. Change the flow for the Amend Order action (optional if you want a different flow)</p> | <ol style="list-style-type: none"> 1. Go to Setup > Create > Objects > Order (Managed) > Configure Products (NG). 2. Click Edit. 3. Under Formula Options, copy the formula. 4. Go back to Order object. From Custom Fields & Relationships click New. 5. Click the Formula radio button, then click Next. 6. Enter a Field Label. Press tab to automatically populate the Field Name based on the label you entered. Select the Text radio button, then click Next. 7. Configure the Formula that defines this field. For example: <div data-bbox="571 797 1426 1050" style="border: 1px solid #ccc; padding: 10px; margin: 10px 0;"> <pre>IF (LEN(Apttus_Config2__PriceListId__c) > 0 , HYPERLINK("/apex/Apttus_Config2__OrderAmend?id=" &Id & "&flow=NGDefault", IMAGE("/resource/Apttus_Config2__Button_Configure", "Configure Products"), "_self"), NULL)</pre> </div> 8. Click Check Syntax to test the formula for errors. 9. Click Next. The Establish field-level security page is displayed. 10. Make changes to field-level security as needed. Click Next. 11. Select the page layout that should include this field and click Save. <p>For more information about Flow Settings, refer to the latest <i>CPQ Administrator Guide</i>.</p> |
| <p>3. Display the Cancel Order Lines action on the cart</p> | <p>Specify the Order Line Item and Order Line Status in the Config System Properties or include them in the Display Column settings for a flow.</p> <p>Go to Config Settings > System Properties > View Cart Custom Fields and set the API names as follows:</p> <ul style="list-style-type: none"> • Apttus_Config2__OrderLineItemId__c • Apttus_Config2__OrderLineStatus__c • Apttus_Config2__Quantity__c <p>Go to Config Settings > Display Column Settings > Select the Display Type and Flow > Add Order Line Item and Order Line Status and Save the custom settings.</p> |

| Configuration | Description |
|---|---|
| 4. Add the <i>Deleted</i> value to the Pricing Status field in the Line Items (Cart Line Item) object | <ol style="list-style-type: none"> 1. Go to Setup > Create > Objects > Line Item (Managed). 2. Edit the Pricing Status field. 3. Under Values, click New. 4. Add Deleted as a picklist value and click Save. |
| 5. Add the Amend Order, Undo Amend Order, Cancel Order, and Undo Cancel Order actions to the Order Detail page | <p>Go to Order Detail page > Edit Layout > Order Layout > Buttons > Search the required button > Drag and drop the button to the relevant section in order to make the following buttons available on the Orders UI and click Save.</p> <ul style="list-style-type: none"> • Amend Order • Undo Amend Order • Cancel Order • Undo Cancel Order |
| 6. Expose version details information on order and order line items | <ol style="list-style-type: none"> 1. Go to the Order Detail page > Edit Layout > Order Layout > Fields. 2. Drag and drop the Next Version field onto the appropriate section. 3. Repeat step 2 for Previous Version, and Version Number. 4. Click Save. |
| 7. Display the Change Status field on the Order Line items related list | <ol style="list-style-type: none"> 1. Go to the Order Line Item Detail page > Edit Layout > Order Line Item Layout > Fields. 2. Drag and drop the Change Status field onto the appropriate section. 3. Click Save. |
| 8. Add the <i>In Amendment</i> value to the Status field in the Product Configuration object | <ol style="list-style-type: none"> 1. Go to Setup > Create > Objects > Product Configuration (Managed). 2. Edit the Status field. 3. Under Values, click New. 4. Add In Amendment as a picklist value and click Save. |

Configuring Automation for In-Flight Order Changes and Cancellation

In-Flight Order Changes and Cancellation can be automated through the use of Order Workflow Rulesets and Rules that define when to automatically amend or cancel orders


based on specified criteria. No additional configuration is required to make Workflow Rulesets available to users of Order Management.

Refer to the *Order Management User Guide* for step-by-step instructions and examples of workflow automation rules. For detailed information on invoking the [execOrderWorkflow API](#), refer to the *Order Management API Guide*.

Configuring Order Workflow Rulesets

You can create and define rules under Order Workflow Rulesets to automate In-Flight Orders for certain business scenarios. The Order Workflow Ruleset object is included with Order Management out of the box but requires some configuration in order for you to create and manage rulesets.

Perform the following tasks to set up Order Workflow Rulesets.

 For your convenience in creating rules, you can add the Order Workflow Rulesets tab to your default view as you would add any other object tab.


| Configuration | Description |
|---|---|
| Add the Criteria button to the Order Workflow Ruleset Detail page | <ol style="list-style-type: none"> Go to Order Workflow Ruleset > Edit Layout > Order Workflow Ruleset Layout > Buttons. Drag and drop the Criteria button to the Custom Buttons area in the layout. Click Save. |
| Add the Item Criteria and Item Input buttons to the Order Workflow Rule Entry Detail page | <ol style="list-style-type: none"> Go to Order Workflow Rule Entry > Edit Layout > Order Workflow Rule Entry Layout > Buttons. Drag and drop the Item Criteria and Item Input buttons to the Custom Buttons area in the layout. Click Save. |

 In Salesforce Lightning, go to **Setup > Object Manager** to edit the page layouts for the Order Workflow Ruleset and Order Workflow Rule Entry objects.


Configuring an Order Workflow Rule for Split Order

You can create and define rules under Order Workflow Rulesets to automate the split order for certain business scenarios. This requires some configuration in order for you to create and manage rulesets.

To automate the split order under Order workflow rulesets, perform the following task.

 For your convenience in creating rules, you can add the Order Workflow Rulesets tab to your default view as you would add any other object tab.

| Configuration | Description |
|---|---|
| Add Split Order to the Action field in the Order Workflow Rule Detail page. | <ol style="list-style-type: none"> 1. Go to Setup > Custom Objects > Order Workflow Rule > Custom Fields & relationships > Action. 2. Under Values, click New. 3. Add Split Order as a picklist and click Save. |

 In Salesforce Lightning, go to **Setup > Object Manager** to edit the page layouts for the Order Workflow Ruleset and Order Workflow Rule Entry objects.

Configuring the Direct Order Workflow

A Direct Order is placed by an Order Management user either by creating a new order from the "Orders" tab or from the Order related list on an Account. To create an order in this way, you must configure the Order page with the appropriate status, buttons, and formulas to allow user to configure products and use contract pricing.

Add the **Accept** and **Configure Products (NG)** buttons to the Order page layout if they have not already been added. When you create a new order, depending on the Price List you select, the Configure Product (NG) button is displayed for the products associated with that Price List.

- [Configuring the Accept and Configure Products Buttons](#)
- [Changing the Flow on the Configure Products Button](#)
- [Using a Contract Price List for an Order](#)

Configuring the Accept and Configure Products Buttons

You can change the label and formula used for either of these buttons depending on your individual use case.

To control the visibility of Accept and Configure Products formula fields on the Orders page

1. From the Order page, go to **Quick Access > View Fields**.
2. Click **New**.
3. Click the **Formula** radio button and click **Next**.
4. Enter a mandatory **Field Label** (for example, "Accept (Pending)") and press **Tab** on your keyboard to auto-populate the **Field Name**.
5. Click the **Text** radio button and click **Next**. The Enter Formula page is displayed.
6. Enter code to be used in the formula. In this example, the Accept button is made visible when the order and its line items is in *Draft*, *Pending*, *In Amendment*, and *Pending Cancellation* status.

```
IF (OR(ISPICKVAL(Conga_Config2__Status__c, "Draft"),
ISPICKVAL(Apttus_Config2__Status__c, "Pending"),
ISPICKVAL(Conga_Config2__Status__c, "In Amendment"),
ISPICKVAL(Apttus_Config2__Status__c, "Pending Cancellation")), HYPERLINK("/
apex/Conga_Config2__OrderAccept?id="&Id, IMAGE("/resource/
Apttus_Config2__Button_Accept", "Accept"), "_self"), NULL)
```

7. Click **Check Syntax** to test the formula for errors.
8. Click **Next**. The Establish field-level security page is displayed.
9. Make changes to field-level security as needed. Click **Next**.
10. Select the page layout that should include this field and click **Save**.

Changing the Flow on the Configure Products Button

To control the flow and other properties for configuring products from a direct order, you can create a new formula field as described in the previous task. In the formula, change the value after "&flow=" to the flow you want the formula field to use when a user clicks the button from an order.

```
IF ( LEN( Conga_Config2__PriceListId__c ) > 0 , HYPERLINK("/apex/Apttus_Config2__OrderConfiguration?id=" &Id & "&flow=NGDefault", IMAGE("/resource/Conga_Config2__Button_Configure", "Configure Products"), "_self"), NULL)
```

For information on flows, cart views, and flow settings, refer to the *CPQ Administrator Guide*.

Using a Contract Price List for an Order

To identify a specific Contract Price List to use when configuring products for direct order, you can create a new formula field as described in the steps above. In the formula field, specify the contract numbers and any field that points to the agreement.

```
IF ( LEN( Apttus_QPConfig__PriceListId__c ) > 0 , HYPERLINK("/apex/Conga_QPConfig__ProposalConfiguration?id=" &Id& "&cntrNbr_1="&Apttus_Proposal__Account__r.Pricing_Agreement_Number__c, IMAGE ("/resource/Conga_QPConfig__Button_Configure", "Configure Products"), "_self"), NULL)
```

In the example above, the cntrNbr_1 parameter and the field which represents an agreement number on an account are passed to the URL.

For information on contract pricing, refer to the topics "Contract Pricing" and "Defining a Contract to be used in a Quote" in the *CPQ Administrator Guide*.

Setting Up Partial Orders from Price Agreements

Sales orders are generated using Price Agreements from quotes or agreements where the "Intent" field is equal to "Price Agreement." This allows Order Management users to place

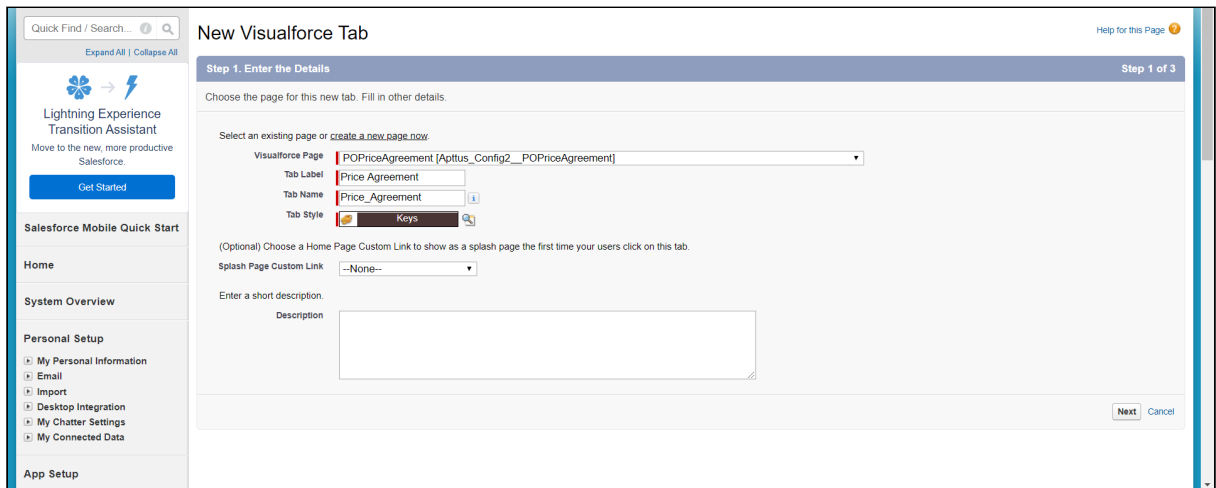
full or partial sales orders against a Price Agreement whose line items have been pre-negotiated at specific quantities and pricing.

Configuring Price Agreements

Administrators are responsible for setting up the tab that navigates the user to the Price Agreements user interface and configuring settings that customize the layout of pages in the UI. Complete all of the following tasks to set up Price Agreements for Order Management.

To create the tab for managing Price Agreements

1. Log in to your Salesforce org.
2. Go to **Setup > Tabs**.
3. Under Visualforce Tabs, Click **New**.
4. Click Visualforce Page drop-down and select **POPriceAgreement**.
5. Enter a label for the tab (for example, "Price Agreement").
6. Press tab to autofill the tab API name.
7. (Optional) Choose a tab style.



8. Click **Next**.
9. Select tab visibility for profiles and click **Next**.
10. Select which apps will have access to this tab. Click **Save**.

To configure Order System Properties for Price Agreements

1. Go to **Setup > Custom Settings > Order System Properties**
2. Click **Manage**.
3. Click **Edit** to the left of System Properties.
4. Enter values for the properties described in the following table:

| Property | Description | Example Value |
|---|--|--|
| Agreement Fields for Partial Order | Enter the list of field API names from the Agreement object to display as columns on the agreement list view page under Price Agreements. | Conga__Account__c, Apttus__Account__r. Name, Apttus_CMConfig__P riceListId__c, Apttus_CMConfig__P riceListId__r.Name, Apttus__FF_Agreeme nt_Number__c |
| Agreement Fields for Partial Order2 | Use this field to list additional Agreement field API names when the total number of characters for the "Agreement Fields for Partial Order" property is exceeded. | |
| Agreement Item Fields for Partial Order | Enter the list of field API names from the Agreement Line Item object to display as columns on the agreement line item list view page under Price Agreement Details. | Apttus__AgreementI d__c, Apttus__ProductId__r .Name, Apttus_CMConfig__ OrderedQuantity__c, Apttus__Quantity__c , Apttus_Approval__A pproval_Status__c, Apttus_CMConfig__B illToAccountId__r.Na me, Apttus_CMConfig__S hipToAccountId__r.N ame |

| Property | Description | Example Value |
|---|---|--|
| Agreement Item Fields for Partial Order | Use this field to list additional Agreement Line Item field API names when the total number of characters for the "Agreement Item Fields for Partial Order" property is exceeded. | |
| Quote Fields for Partial Order | Enter the list of field API names from the Quote/ Proposal object to display as columns on the quote list view page under Price Agreements. | Apttus_Proposal__Account__c, Apttus_Proposal__Account__r.Name, Apttus_QPConfig__PriceListId__c, Apttus_QPConfig__PriceListId__r.Name, Apttus_Proposal__Approval_Stage__c, Apttus_QPConfig__Intent__c, Apttus_QPConfig__ShipToAccountId__r.Name |
| Quote Fields for Partial Order2 | Use this field to list additional Quote/Proposal field API names when the total number of characters for the "Quote Fields for Partial Order" property is exceeded. | |
| Quote Item Fields for Partial Order | Enter the list of field API names from the Quote/ Proposal Line Item object to display as columns on the quote line item list view page under Price Agreement Details. | Apttus_QPConfig__EndDate__c, Apttus_QPConfig__StartDate__c, Apttus_Proposal__Product__r.Name, Apttus_Proposal__Proposal__c, Apttus_QPConfig__Quantity2__c, Apttus_QPConfig__OrderedQuantity__c |

| Property | Description | Example Value |
|--------------------------------------|--|---------------|
| Quote Item Fields for Partial Order2 | Use this field to list additional Quote/Proposal Line Item field API names when the total number of characters for the "Quote Item Fields for Partial Order" property is exceeded. | |

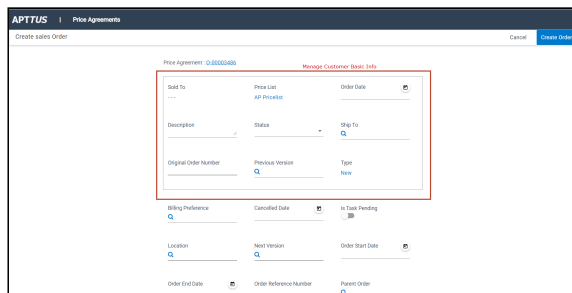
⚠ Field values must be comma-separated.

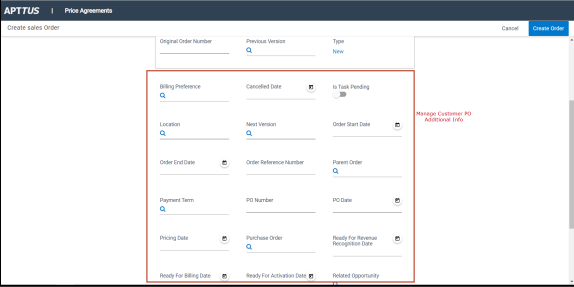
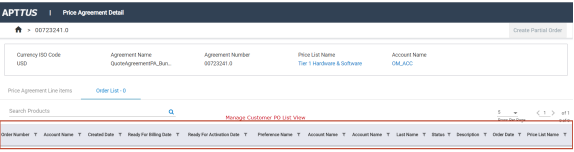
5. Click **Save**.

To configure fields sets for displaying information in the Price Agreement UI

1. Go to **Setup > Objects > Order**.
2. Under Field Sets, click **Edit** next to the field set you want to configure. Refer to the following table for the list of configurable field sets and their purpose:

| Field Set | Purpose | Notes |
|-------------------------------|--|--|
| Manage Customer PO Basic Info | This field set is used to display the list of fields in the header of the Create Sales Order screen. | Fields are displayed in the order they are added to the field set. |



| Field Set | Purpose | Notes |
|--|--|---|
| <p>Manage Customer Additional Info</p> | <p>This field set is used to display additional fields on the Create Sales Order screen.</p>  | <p>Fields are displayed in the order they are added to the field set.</p> |
| <p>Manage Customer PO List View</p> | <p>This field set is used to determine which columns are displayed in the Price Agreement Detail screen.</p>  | <p>Price Agreements can be sorted or filtered by column.</p> |

- When you are done adding fields to a field set, click **Save** to save your changes. Go to the appropriate page under Price Agreements to verify the fields are displayed.

Configuring the Contract Price List for Orders With Commitment Compliance

When setting up partial orders, you must configure the Quote/Proposal and Agreement objects to generate a Price List that refers to the product, bundle, and service line item pricing information on the quote or agreement originating the order. In this case, when the quote is accepted or the contract is finalized, a Contract Price List is generated that is part of the Price Agreement. Users can then create partial orders to fulfill quantities at the negotiated price for product and service line items (on the quote or contract). As an administrator, you need to add the "Price Agreement" option to the *Intent* picklist custom field on the Quote/Proposal and Agreement objects. When a quote or contract with "Intent = Price Agreement" is accepted or finalized, a new Price List is generated with the type

"Contract." Price List Items (PLIs) store information on the products and services copied over from the original quote or agreement.

- For PLIs with the list price, the new net unit price is copied to contract price field of the new PLI.
- For PLIs with matrix pricing (recurring), the matrix is copied from original PLI to the new PLI.
- For PLIs with matrix pricing (usage), the matrix is copied from usage price tiers of the line item to the new PLI.

To configure the Quote/Proposal object for partial orders

1. From Salesforce, click **Setup > Objects > Quote/Proposal**.
2. Under Custom Fields and Relationships, click **Intent**.
3. Under Values, click **New**.
4. Add the new picklist value "Price Agreement." Click **Save**.

To configure the Agreement object for partial orders

1. From Salesforce, click **Setup > Objects > Agreement**.
2. Under Custom Fields and Relationships, click **Intent**.
3. Under Values, click **New**.
4. Add the new picklist value "Price Agreement." Click **Save**.

Setting Up Customer Purchase Orders

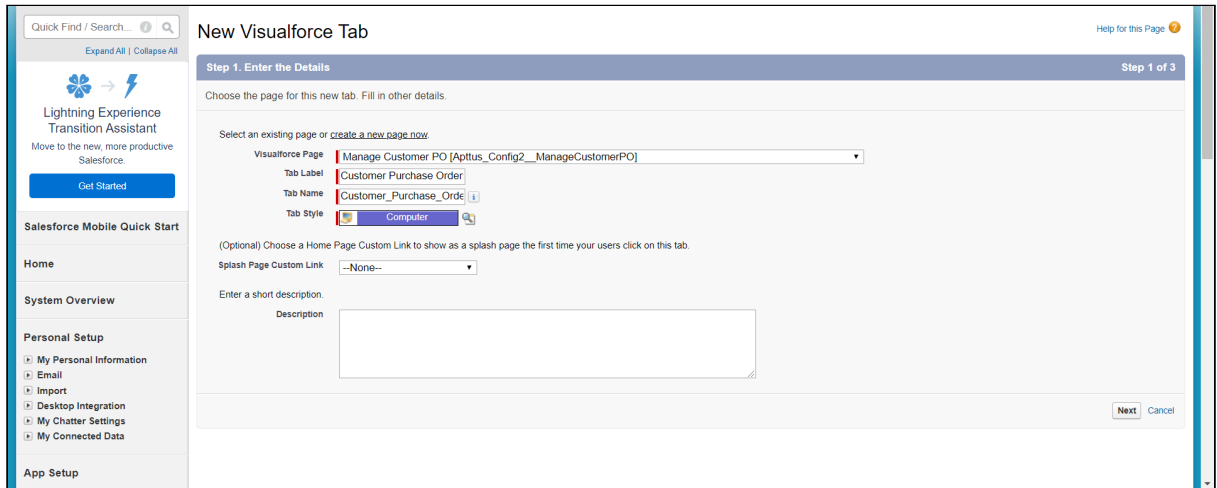
A Customer Purchase Order (PO) is a document that represents the initial offer of negotiated types, quantities, and prices for products or services. Customer Purchase Orders can be created by sales users or administrators as sales orders for long-term contracts.

Configuring Customer Purchase Orders and Purchase Order Items

Administrators are responsible for setting up Order Management to allow users to work with Customer Purchase Orders. To configure Order Management for Customer Purchase Orders, perform the following tasks in order.

To create a tab for managing Customer Purchase Orders and Items

1. Log in to your Salesforce org.
2. Go to **Setup > Tabs**.
3. Under Visualforce, click **New**.
4. Click the Visualforce Page drop-down and select **ManageCustomerPO**.
5. Enter a label for the tab (for example, "Customer Purchase Orders").
6. Press tab to autofill the tab API name.
7. (Optional) Choose a tab style.



8. Click **Next**.
9. Select tab visibility for profiles and click **Next**.
10. Select which apps will have access to this tab and click **Save**.

To enable default draft status for Customer Purchase Orders

i Repeat this task for the Customer Purchase Order Items object.

1. Go to **Setup > Objects > Customer Purchase Order** (In Lightning, Setup > Object Manager < Customer Purchase Order).
2. From Custom Fields & Relationships, click **Status**.
3. Click **New**. The Status Picklist Values page is displayed.
4. In the picklist entry field, enter the values *Draft*. Click **Save**.

- Next to the Draft status, click **Edit**.
- From the Picklist Edit page click the check box **Make this value default for the master picklist** and click **Save**.

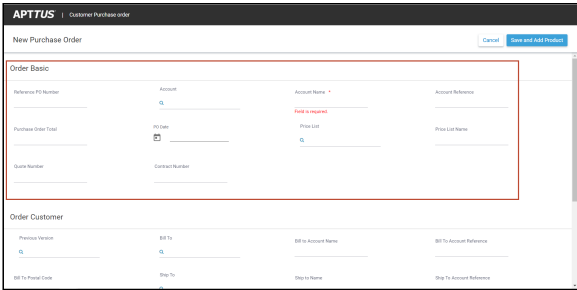
i The default status must be "Draft" to make the Edit PO button visible.

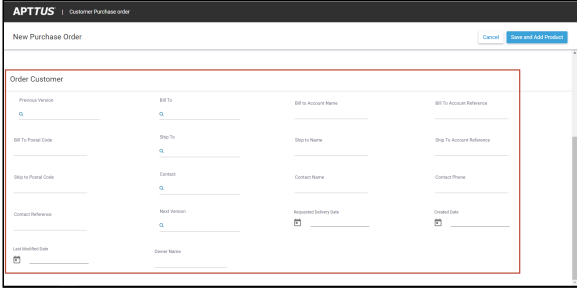
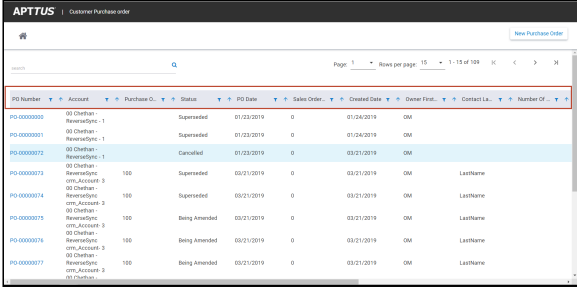
- Click **Save**.

To configure field sets for displaying information on Customer Purchase Orders

Configure field sets to display required fields in the PO Admin UI. For example, if a CPO needs to be validated against a quote, ensure that the quote and quote line item fields are present in the appropriate field set.

- Go to **Setup > Objects > Customer Purchase Order**.
- Under Field Sets, click **Edit** next to the field set you want to configure. Refer to the following table for the list of configurable field sets and their purpose:

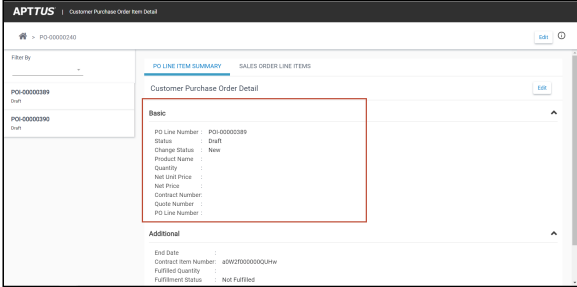
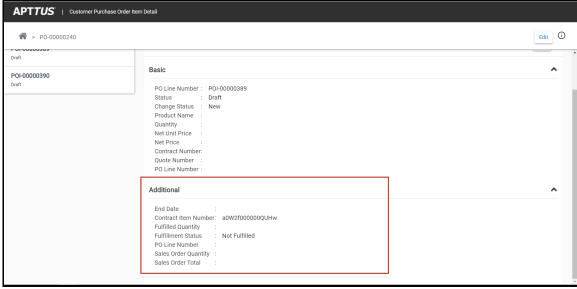
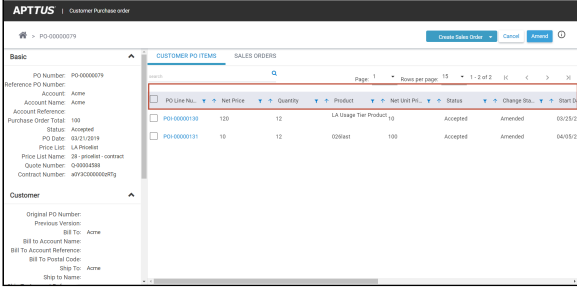
| Field Set | Purpose | Notes |
|-------------------------------|---|--|
| Manage Customer PO Basic Info | <p>Add fields to this fields set to display field values in the "Order Basic" section of the New Purchase Order screen.</p>  | Fields are displayed in the order they are added to the field set. |

| Field Set | Purpose | Notes |
|---|---|---|
| <p>Manage Customer PO Additional Info</p> | <p>Add fields to this field set to display field values in the "Order Customer" section of the New Purchase Order screen.</p>  | <p>Fields are displayed in the order they are added to the field set.</p> |
| <p>Manage Customer PO List View</p> | <p>Add fields to this field set to determine which columns are displayed in the list of purchase orders on the Customer Purchase Order screen.</p>  | <p>Customer Purchase Orders can be sorted or filtered by column.</p> |

- When you are done adding fields to the field set, click **Save** to save your changes. Click on the **Customer Purchase Order** tab to verify fields have been added correctly.

To configure field sets for displaying information on Customer Purchase Order Items

- Go to **Setup > Objects > Customer PO Item**.
- Under Field Sets, click **Edit** next to the field set you want to configure. Refer to the following table for the list of configurable field sets and their purpose:

| Field Set | Purpose | Notes |
|---|---|---|
| <p>Manage Customer PO Basic Info</p> | <p>Add fields to this fields set to display field values in the "Basic" section of the PO Line Item Summary > Customer Purchase Order Detail screen.</p>  | <p>Fields are displayed in the order they are added to the field set.</p> |
| <p>Manage Customer PO Additional Info</p> | <p>Add fields to this field set to display field values in the "Additional" section of the PO Line Item > Customer Purchase Order Detail screen.</p>  | <p>Fields are displayed in the order they are added to the field set.</p> |
| <p>Manage Customer PO List View</p> | <p>Add fields to this field set to determine which columns are displayed in the list of purchase order items on the Customer Purchase Order > Customer PO Items screen.</p>  | <p>Customer Purchase Order Items can be sorted or filtered by column.</p> |

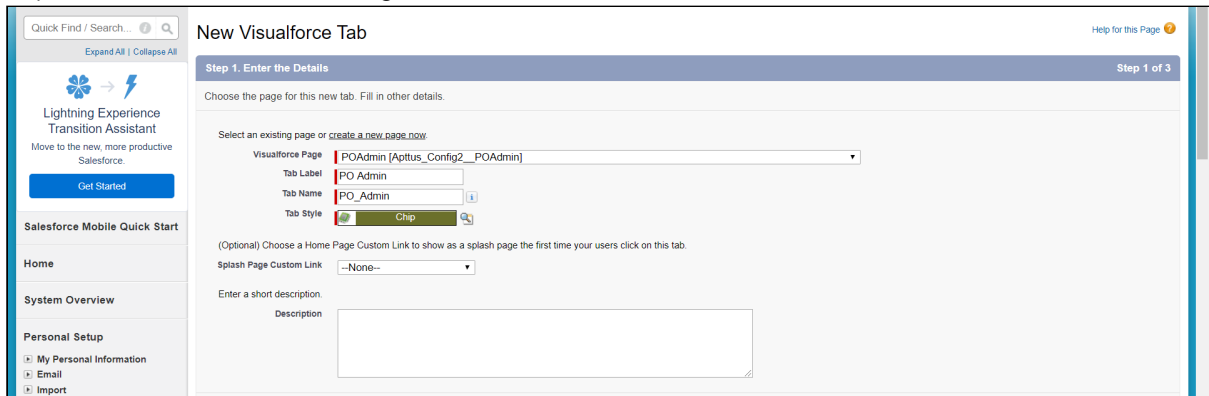
- When you are done adding fields to the field set, click **Save** to save your changes. Click on the **Customer Purchase Order** tab to verify fields have been added correctly.

Setting Up Purchase Order Admin

Administrators are responsible for setting up Order Management to allow users to create Enrichment and Validation rules for Customer Purchase Orders. Enrichment and Validation rules are created using the Purchase Order Admin (PO Admin) angular user interface. The PO Admin UI provides a step-by-step wizard that allows you to create Data Enrichment and Validation Rulesets which contain one or more Data Validation and/or Data Enrichment rules. For more information on Enrichment and Validation rules, refer to Managing CPO Validation and Enrichment Rules in the Order Management on Salesforce User Guide.

To create a tab for managing Customer Purchase Order Enrichment and Validation rules

1. Log in to your Salesforce org.
2. Go to **Setup > Tabs**.
3. Under Visualforce, click **New**.
4. Click the Visualforce Page drop-down and select **PO Admin**.
5. Enter a label for the tab (for example, "PO Admin").
6. Press tab to autofill the tab API name.
7. (Optional) Choose a tab style.



8. Click **Next**.
9. Select tab visibility for profiles and click **Next**.
10. Select which apps will have access to this tab and click **Save**.

To configure the Enrichment/Validation object for Customer Purchase Orders

1. Go to **Setup > Objects > Data Enrichment/Validation Ruleset** (in Lightning, Setup > Object Manager > Data Enrichment/Validation Ruleset).
2. Go to **Fields & Relationships > Source Data Object**.
3. Ignore the initial picklist values. Click **New** and enter the following with the correct namespace: *Conga_Proposal__Proposal__c*, *Conga_Proposal__Proposal_Line_Item__c*, *Conga__APTS_Agreement__c*, *Apttus__AgreementLineItem__c*, *Conga_Config2__Order__c*, *Conga_Config2__OrderLineItem__c*. Click **Save**.
4. Go to **Fields & Relationships > Target Object to Validate**.
5. Ignore the initial picklist values. Click **New** and enter the following with the correct namespace: *Conga_Config2__CustomerPurchaseOrder__c*, *Apttus_Config2__CustomerPOItem__c*. Click **Save**.
6. Go to **Fields & Relationships > Target Object Parent**.
7. Ignore the initial picklist values. Click **New** and enter the following with the correct namespace: *Conga_Config2__CustomerPurchaseOrder__c*, *Apttus_Config2__CustomerPOItem__c*. Click **Save**.

Appendices

Order System Properties

You can access these settings from **Setup > Develop > Custom Settings > Order System Properties**.

| Setting | Description |
|--|--|
| Initiate Billing On Order Activation | Indicates whether billing should be initiated on order activation. |
| Create Asset On Order Activation | Indicates whether asset creation is delayed until order activation. |
| Enable Inflight Changes And Cancellation | Indicates whether inflight order changes and cancellation are enabled. |

| Setting | Description |
|--|--|
| Agreement Fields For Partial Order2 | Enter the list of field API names from the agreement object for display when creating partial orders. Each field may be separated by a comma or a new line. |
| Agreement Fields For Partial Order | Enter the list of field API names from the agreement object for display when creating partial orders. Each field may be separated by a comma or a new line. |
| Agreement Item Fields For Partial Order2 | Enter the list of field API names from the agreement line item object for display when creating partial orders. Each field may be separated by a comma or a new line. |
| Agreement Item Fields For Partial Order | Enter the list of field API names from the agreement line item object for display when creating partial orders. Each field may be separated by a comma or a new line. |
| Quote Fields For Partial Order2 | Enter the list of field API names from the quote/proposal object for display when creating partial orders. Each field may be separated by a comma or a new line. |
| Quote Fields For Partial Order | Enter the list of field API names from the quote/proposal object for display when creating partial orders. Each field may be separated by a comma or a new line. |
| Quote Item Fields For Partial Order2 | Enter the list of field API names from the proposal line item object for display when creating partial orders. Each field may be separated by a comma or a new line. |
| Quote Item Fields For Partial Order | Enter the list of field API names from the proposal line item object for display when creating partial orders. Each field may be separated by a comma or a new line. |
| Split Order Criteria Fields | Enter the list of field API names from Order line item objects to split items added during a large order process into multiple orders. Separate each field with a comma or a new line. |

| Setting | Description |
|-----------------------|--|
| Split Order Threshold | Indicates the maximum number of a bundle or standalone products (represented as line numbers) that may be added to a split order. Defaults to 100. |

Order Management for Users

The *Order Management User Guide* is designed to provide customer sales and support representatives with information for managing the life of a generated order before it creates or updates an asset. This guide covers the major features that are provided with the Conga Order Management license, including changes to in-flight orders, partial orders from contract price agreements, and management of customer purchase orders. Many of the topics in this guide cover tasks that can be considered part of an overall workflow for Conga purchase and revenue management products – therefore familiarity with Conga Configure Price Quote (CPQ) products are highly encouraged to understand and execute Order Management concepts and tasks.

| Topic | Description |
|------------------|---|
| What's Covered | This guide walks sales users through the Order Management workflow and lifecycle, from the generation of order to creation of assets, fulfillment tracking, and integration with downstream services. It provides conceptual information, step-by-step instructions, and use cases for the features provided by Conga Order Management. |
| Primary Audience | <ul style="list-style-type: none"> • Customer Support Representative • Sales Representative |
| IT Environment | Refer to the latest <i>Order Management on Salesforce Release Notes</i> for information on System Requirements and Supported Platforms. |
| Updates | For a comprehensive list of updates to this guide for each release, see the What's New in Order Management Documentation topic. |

| Topic | Description |
|-----------------|---|
| Other Resources | <ul style="list-style-type: none"> • Conga Order Management Summer 2020 Administrator Guide: Refer to this guide for installing and setting up Order Management in your organization. • Conga Order Management Summer 2020 Release Notes: Refer to this guide for the new feature, enhancements, resolved, and known issues. • Conga Order Management Summer 2020 SOAP API Guide: Refer to this guide for documentation of public-facing SOAP APIs for Order Management. • CPQ Administrator Guide: Refer to this guide for setting up products, price lists, and constraint rules. |

This guide describes the following tasks:

- Managing orders in the Quote-to-Cash Process
- Managing order lifecycle and status
- Creating direct orders for accounts
- Creating partial orders from customer price agreements (quote/contract)
- Managing in-flight order changes and cancellation
- Managing distributed order fulfillment
- Activating an order
- Billing for an order
- Creating customer purchase orders

Before using Order Management, you must be familiar with the following:

- Basic Salesforce Administration knowledge
- Conga CPQ and Conga CLM administration
- Salesforce and Conga terms and definitions

DOC ID: OMSFSUM20UG20200804


Select one of the following topics for more information:

- [Logging in to Order Management](#)
- [Navigating the Order Management User Interface](#)
- [Understanding the Order Management Lifecycle](#)
- [Capturing Orders](#)
- [Managing In-Flight Order Changes and Cancellation](#)
- [Working With Customer Price Agreements](#)

- [Splitting an Order](#)
- [Working With Distributed Order Fulfillment](#)
- [Activating an Order](#)
- [Billing for an Order](#)
- [Working with Customer Purchase Orders](#)
- [Intelligent Order Assistant \(Max for OM\)](#)

Logging in to Order Management

Log in to your Salesforce.com org to access Conga Order Management.

 Do not use the Back button on your browser when using Order Management.

Before logging in to Order Management, make sure you meet the following criteria:

- All required Order Management packages (included Conga CPQ packages, and packages for other integrated Conga applications) have been installed by an administrator.
- You have login credentials provided by Conga.

To log in to Order Management

1. Go to <https://salesforce.com/>.

Or

If your organization is using a sandbox or test environment to access Conga Order Management (for example, if you are doing user acceptance testing), go to <https://test.salesforce.com/> instead.

2. From the toolbar at the top of the page, click **Login**. The login page opens.
3. Enter your user name and password, and click **Log in**.
4. Navigate to Conga Order Management:
 - In Salesforce Classic: Click the App Menu and select **Conga Order Management**.
 - In Salesforce Lightning Experience: Click the App Launcher and select **Conga Order Management**.

Navigating the Order Management User Interface

The Order Management User Interface (UI) is presented to the user similarly to many other Conga applications on Salesforce. The default landing page for Order Management takes you to the Order tab, which displays a list of Recent Orders and links and options in the Salesforce sidebar to the left (Salesforce Classic).

You can use standard Salesforce functionality such as global search, dashboard controls, tabs, and menus to navigate to various Order Management pages. Refer to Salesforce documentation for any differences in navigation using Salesforce Classic versus the Lightning Experience.

Conga Order Management provides a number of tabs to use its various features. Refer to the following table for brief descriptions of each tab:

i Note: Configuration of certain tabs require administrator privileges or user permissions. Contact your administrator if you do not have access to any of the tabs described below.

| Tab | Description | Help Topic(s) |
|------------------------------|---|---|
| Orders | Default landing page for the Order Management application. View, manage, and create new direct orders from this page. | Capturing Orders, Creating Direct Orders for Accounts |
| Order Workflow Rulesets | Use this tab to view, create, and manage Order Workflow Rulesets. | Creating Order Workflows to Automate In-Flight Order Changes |
| Order Fulfillments | Use this tab to view and manage Order Fulfillments. | Working With Distributed Order Fulfillment, Releasing Orders to Fulfillment Systems |
| Order Fulfillment Line Items | Use this tab to view, create, and manage Order Fulfillment Line Items. | Working With Distributed Order Fulfillment, Releasing Orders to Fulfillment Systems |

| Tab | Description | Help Topic(s) |
|-------------------------------------|---|---|
| Price Agreement | Use this tab to view Price Agreements created from Quote/Proposals and/or Agreements. Create partial orders from Contract Price List Items. | Working With Customer Price Agreements |
| Customer Purchase Orders | Use this tab to view and manage Customer Purchase Orders. Create and manage purchase orders and purchase order items, and create sales orders, split, and partial orders from purchase orders. View and add attachments to purchase orders. | Working with Customer Purchase Orders, Creating Sales Orders from a Customer Purchase Order |
| Config Settings | Use this tab to manage config settings and system properties. Refer to the <i>CPQ on Salesforce Administrator Guide</i> for more information. | |
| Data Validation/Enrichment Rulesets | Use this tab to create and manage Data Enrichment and Validation rulesets for Customer Purchase Orders. | Managing CPO Validation and Enrichment Rules |
| PO Admin | Use this tab to create and manage Data Enrichment and Validation rules for Customer Purchase Orders. | Managing CPO Validation and Enrichment Rules |

Working in the Conga User Interface

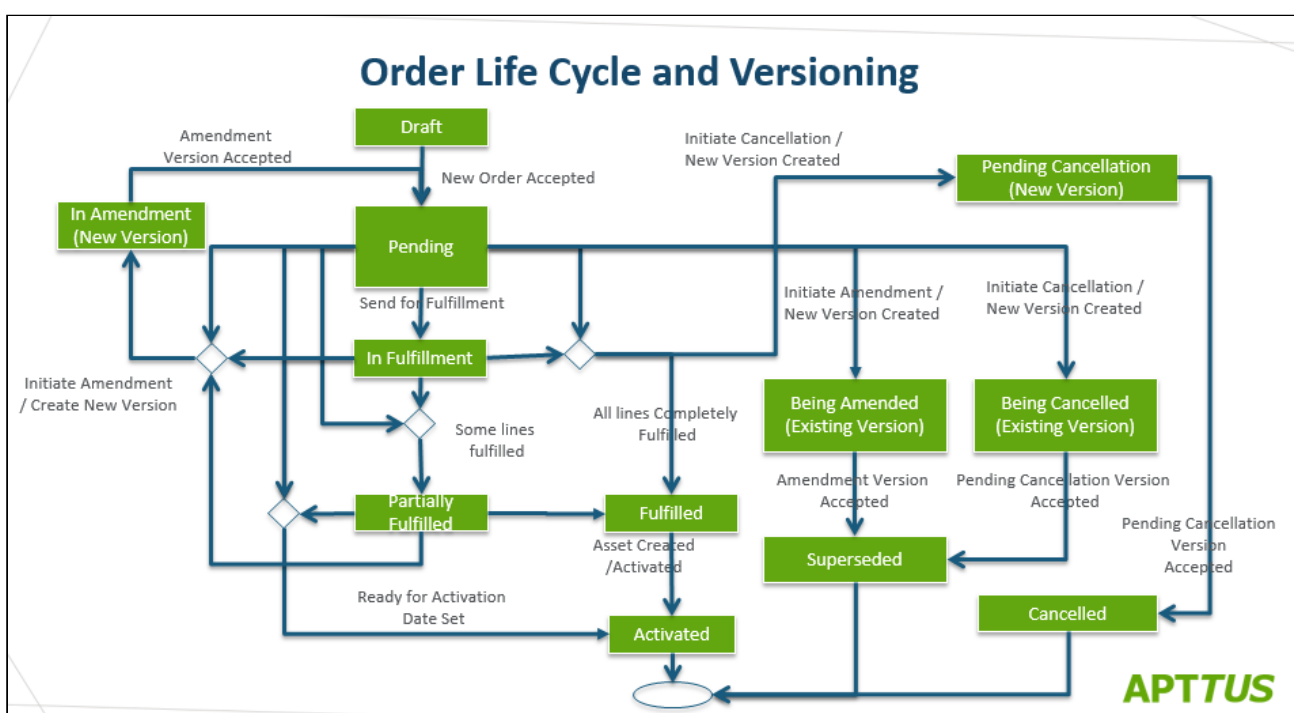
Several features of Order Management use a custom user interface provided by Conga. The following provides a brief summary of UI controls and features:

- To search for records using a type-ahead field (marked by the lookup (🔍) icon), enter text in the field and select a result to refresh the list.
- Click a column header name to sort the list by that column.
- Click the filter (🔽) icon in the column header, enter a filter string, and click **Apply** to filter the list by column value.
- Use the pagination controls to change the number of rows per page or to move between pages.
- When working with forms, use the date picker (📅) to select a date value for the field.

Understanding the Order Management Lifecycle

Using Order Management, customer support representatives can track an order through its entire lifecycle to asset creation. Order fulfillment status can be tracked at the order level or at the individual line item level.

The following diagram represents a high-level lifecycle flow for orders created in Conga Order Management.




Orders begin in **Draft** status, either through direct order creation or conversion from a Quote or Agreement (in Conga CPQ or CLM). From here, a **Pending** order can be confirmed or **Amended**. After an order is confirmed, it moves to fulfillment status, either as a **Partially Fulfilled** order, or **Fulfilled** and **Activated**. Orders can also be **Cancelled**, either initiated manually, or superseded by a new version.

Understanding Order Status

Order and Order Line Item status work as a mechanism to integrate with downstream systems and processes including fulfillment systems, asset management, billing, and revenue recognition.

- [Order Header Status](#)

- [Order Line Item Status](#)

 The majority of actions described in this topic that assign status to an order or order line items can also be executed using Order Service APIs. For information on how to use Order Service APIs, refer to the Conga *Order Management SOAP API Guide*.

Order Header Status

The following table describes Order Status at the order level and the events and actions that assign status to an order.

| Order Status | Description | Event / Action |
|--------------|--|---|
| Draft | When an order is initially created, its default status is "Draft." | <p>New Order Creating a new order from the Order page or a related list will create the order in "Draft" status. Draft orders are work-in-progress orders and can be opened and edited without creating new versions.</p> <p>Ensure that "Draft" is the default status when creating orders and order line items.</p> <p>You can change the quantity of an order from the cart even when the order is in "Draft" status.</p> <p>You can also clone an existing order to create a new order. For more information, see Cloning an Order.</p> |

| Order Status | Description | Event / Action |
|--------------|---|--|
| Pending | When an order is confirmed by a customer but is pending for validation. Such orders and all the order line items have a "Pending" status. | <p>Accept Order / Accept Quote / Activate Agreement</p> <p>An order is confirmed when the Accept button is clicked on the Order or Quote record, or when an Agreement is Activated. Accepting an order will change the status of the order and order line items to "Pending" if Auto Activation is not enabled.</p> <p>To generate orders in "Pending" status from quotes and agreements, ensure:</p> <ul style="list-style-type: none"> • Auto Create Orders for quotes is set to "True" Go to Custom Settings > Proposal System Properties to apply this property. • Auto Create Order for agreements is set to "True" Go to Custom Settings > Comply System Properties to apply this property. • Auto Activate Order is set to "False" <p>You are allowed to cancel an order when it is in "Pending" status.</p> <p>Undo Cancel Order</p> <p>When you click Undo Cancel Order, the order reverts the new version to the previous version, which then displays "Pending" status.</p> |

| Order Status | Description | Event / Action |
|----------------|--|--|
| In Fulfillment | The Order and Order Line Item status changes to "In Fulfillment " when order fulfillment begins. This means that one or more line items are in ('In Fulfillment') AND other line items are in "Pending" status | <p>Send for Fulfillment</p> <p>You can set the status to "In Fulfillment" when the Order and Order line items are exported to fulfillment systems to initiate the fulfillment process. It indicates that the order has been released to the fulfillment systems and the fulfillment process has begun.</p> <p>You are allowed to cancel an order when it is in "In Fulfillment" status.</p> <p>Undo Cancel Order</p> <p>When you click Undo Cancel Order, the order reverts the new version to the previous version, which then displays "In Fulfillment" status.</p> |

| Order Status | Description | Event / Action |
|---------------------|--|---|
| Partially Fulfilled | <p>When one or more line items are in "Partially Fulfilled" status.</p> <p>OR</p> <p>If one or more line items are IN ('Activated', 'Fulfilled') status AND one or more IN ('In Fulfillment' OR 'Pending') status.</p> | <p>Create Order Fulfillment Line Items</p> <p>You can manage and track partial fulfillment for an order by creating order fulfillment line items against a given order line item. This updates the "Fulfilled Quantity" on the order line item level and also updates the status of the order line item to "Partially Fulfilled" or "Fulfilled" or "Activated."</p> <p>Activate a Subset of Order Line Items</p> <p>A subset of order line items can also be manually activated by setting the "Ready for Activation Date" field on the corresponding line items. When a subset of the order line items is activated, the overall order status is set to "Partially Fulfilled."</p> |

| Order Status | Description | Event / Action |
|---------------------|---|---|
| <p>In Amendment</p> | <p>This status is set on an "In-Flight" order (in "Pending", "In Fulfillment", or "Partially Fulfilled" status) when changes have been made to the existing order.</p> <p>This status represents work-in-progress changes to the order that have not yet been confirmed (similar to a "Draft" order).</p> | <p>Amend Order</p> <p>Amending an order creates a new version of the order to make and track changes and a reference to the previous version. The previous version of the order and order line items are "Superseded," once the work-in-progress (In Amendment) changes are confirmed on "Accept" of the "In Amendment" order.</p> <p>Amending an In-Flight order allows you to make the following changes:</p> <ul style="list-style-type: none"> • Change the start date or end date of subscription on one or more order line items that are in "Pending," "In Fulfillment," or "Partially Fulfilled" status. • Make subscription date changes on the standalone item, bundles, and multiple charge line items. |

| Order Status | Description | Event / Action |
|--------------|-------------|--|
| | | <ul style="list-style-type: none"> • Cancel or add new line items, and change their quantities, pricing, and discounts. <p>The change in subscription start and/or end date will recalculate the price for the line item or bundle after the changes are applied.</p> <p>The changes to the order line items are also reflected in the corresponding assets. When <i>In-Flight Order Changes</i> is enabled, assets are only created on activation of the order line item.</p> <p>Note: Any order configured directly or generated from a quote, agreement, or E-Commerce can be amended.</p> <p>To change order line item quantities, the field "Is Quantity Modifiable" must enable at the line item level for product configuration.</p> <p>Undo Amend Order</p> <p>If you proceed with Undo Amend Order, it reverts the new version to the previous version, which then displays the status of the original order (for example, "Partially Fulfilled").</p> |

| Order Status | Description | Event / Action |
|---------------|---|--|
| Being Amended | <p>This status is set when an "In-Flight" order (in "Pending," "In Fulfillment," or "Partially Fulfilled" status) is in the process of being amended.</p> <p>This status is assigned to the previous version of the order when an existing order version is in the process of being amended. This status represents that this version is being amended and, once the new changes are confirmed, this version will be superseded by the new order.</p> | N/A |
| Fulfilled | <p>All line items in "Fulfilled" status</p> <p>OR</p> <p>One or more line items are IN ('Fulfilled') status</p> <p>AND</p> <p>One or more line items are IN ('Activated') status</p> | N/A |
| Activated | <p>The order fulfillment process is complete and ready for the initiation of additional related downstream processes (billing, revenue management, and so on).</p> | <p>Auto Activate Order Orders can be automatically activated by configuring the Auto Activate Order setting, or by providing a "Ready for Activation Date" on the Order or Quote header before accepting the corresponding order or quote.</p> <p>Activating All Order Line Items If all Order Line Items are independently fulfilled and activated, the entire Order will be activated.</p> |

| Order Status | Description | Event / Action |
|----------------------|--|---|
| Superseded | This status is set on the previous version of the order when the order version with the status "In Amendment" or "Pending Cancellation" is accepted. | <p>Accept Order</p> <p>When an order with the "In Amendment" or "Pending Cancellation" status is accepted, the previous version status is set to "Superseded."</p> |
| Cancelled | <p>This status is set when an order is cancelled and accepted. Only orders in "Pending" or "In Fulfillment" status can be cancelled.</p> <p>You cannot "Undo Cancel Order" when an order has "Cancelled" status.</p> | N/A |
| Pending Cancellation | This status is set on the new version of the order that is created for cancellation. The status of the order and its line items is set to "Pending Cancellation." | <p>Cancel Order</p> <p>When you cancel an order, a new version is created for cancellation. This version is set to "Pending Cancellation" for both the order and its line items. To complete cancellation, click Accept. The status of the order is then set to "Cancelled."</p> <p>The order line items also undergo versioning when you cancel or amend an order.</p> |
| Being Cancelled | This status is set on the original order when an order is cancelled. Line items remain in "Pending" status. When order cancellation is accepted, the original order status along with its order line items is changed to "Superseded." | N/A |

Order Line Item Status

The following table describes each Order Status at the order line item level. The status of orders and order line items can differ based on the stage of the order fulfillment.

| Order Line Status | Description | Event / Action |
|---------------------|--|--|
| Draft | The Order Line Item is initially created and the Order Status is "Draft." | Creating a New Order. |
| Pending | The Order is confirmed by a customer but is "Pending" for validation. | Accepting Quote/Order. When you perform the following actions, the order line item status changes to "Pending:" <ul style="list-style-type: none"> • Undo Cancel Order • Undo Amend Order |
| In Fulfilment | The Order Line Item has been sent for fulfillment. This Status helps in tracking the fulfillment process at the Order Line Item level. | Sending/Releasing Order to Fulfillment systems. |
| Partially Fulfilled | The Fulfilled Quantity is less than the Delta Quantity and the value is greater than zero. | Tracking Fulfillment for each Order Line Item by creating a corresponding Order Fulfillment Line Item. |

| Order Line Status | Description | Event / Action |
|-------------------|--|---|
| In Amendment | A change has been made to the Order Line Item. | <ul style="list-style-type: none"> • Changes to order line items are also reflected in the corresponding assets. • When the "In-Flight" order change capability is enabled, assets are only created on activation of the order line item. • Amending an order creates a new version of the order to make and track changes and has reference to the previous version. The previous version of the order and order line items have their status changed to "Superseded," once the work-in-progress ("In Amendment") changes are confirmed on Accept of the "In Amendment order". |
| Being Amended | <p>This status is set when an "In-Flight" order line item (order in "Pending," "In Fulfillment," or "Partially Fulfilled") is in the process of being amended.</p> <p>This status is assigned to the previous version of the order line item when an existing order version is in the process of being amended. This status represents that this version is being amended and, once the new changes are confirmed, this version will be superseded by the new order line item.</p> | Amend Order |

| Order Line Status | Description | Event / Action |
|----------------------|---|---|
| Superseded | This status is set on the previous version of the order when the order related to the "In Amendment" order version is accepted, or when an order with the "Pending Cancellation" status is accepted. | <p>Accept Order</p> <p>When the Accept action is used on an "In Amendment" or "Pending Cancellation" order, then the previous version is "Superseded."</p> |
| Fulfilled | If the Fulfilled Quantity on the Order Line Item is equal to the Delta Quantity. | <p>For the following events/actions:</p> <ul style="list-style-type: none"> • On fulfillment of all Order Line Items. • Directly updating the status of Order Line Item to "Fulfilled." |
| Activated | <p>The Order Line Item's status is set to <i>Activated</i> in any of the following three scenarios:</p> <p>Scenario 1: Line item activation is initiated once the order line item is "Fulfilled."</p> <p>Scenario 2: The Order is auto-activated.</p> <p>Scenario 3: The Order line item is manually activated.</p> <p>When Order Line Items are activated, the system initiates activation of the corresponding asset.</p> | <p>On Fulfillment of all Order Line Items.</p> <p>Providing a value for the Ready for Activation Date field on the Order.</p> |
| Pending Cancellation | This status is set when an order is cancelled. | Cancel Order |
| Cancelled | This status is set when the cancelled order is accepted. | Accept Order |

Capturing Orders

Conga Orders are captured using a number of processes, usually depending on which sales channel the order is originating from. The table below describes at a high level what types of orders are generally tied to which type of sale channels. In some cases, a sales channel, such as direct sales, may capture orders using one or more different processes. The table below covers the most common use cases.

In all cases, orders are captured and stored in Salesforce as Order records. Product, service, and bundle data is stored in Orders as Order Line Items.

| Sales Channel(s) | Order Management Process |
|---|--|
| Direct Sales, Partner Sales | <ul style="list-style-type: none"> • Captures an order from a Quote/Proposal • Captures an order from an Agreement (Contract) • Creates a direct order for an Account (Order Management UI) |
| Customer Service Representatives (CSR), Telesales | <ul style="list-style-type: none"> • Creates a direct order for an Account (OM UI) |
| E-Commerce, Partner Commerce | <ul style="list-style-type: none"> • Creates an order using Order Management APIs • Creates an order from a Customer Purchase Order |

i Total Order Amount on the Order Header

At any time during the ordering process, the total amount of the order is calculated and displayed in the Order Amount field on the order header.

Order Amount = SUM(Net Price) from order line items based on the following conditions:

- Any line item in "Cancelled" or "Pending Cancellation" status is not considered.
- If an option or child bundle price is rolled up to the parent bundle, the system considers the parent bundle net price to avoid double-counting.

The order amount is recalculated whenever the net price on an order line item is updated or when the status of the order line item is changed (to any status other than "Cancelled" or "Pending Cancellation").

Order Management Scope

It is important to consider that the most common use case for generating an order is through a Quote or Agreement. In these cases, orders are almost always automatically activated, meaning that the Order Management application plays a limited or no role in managing the order as it is usually automatically converted into an asset.

Order Management primarily focuses on providing users with the capability to make in-flight order changes, create partial orders from price agreements, and manage and create orders from customer purchase orders. You can create a direct order using Order Management, but even a direct order is mainly initiated by a CPQ or CLM user from the Quote or Agreement record.

For more information on creating and managing orders and assets from a quote, refer to the *CPQ User Guide* and supporting documentation.

For more information on creating and managing orders and assets from an agreement (contract), refer to the *CLM User Guide* and supporting documentation.

Creating Direct Orders for Accounts

Customer service representatives or telesales person can directly create an order for a given account rather than going through the quoting process. Customers typically request for creating orders directly instead of going through the quote creation process in the following scenarios:

| Request for Order Creation | Detail |
|---|---|
| Channel partners placing orders against the channel price list. | Channel partners (Retailers, distributors) generally have standard channel price lists. Channel partners can call customer service/telesales to place new orders. |
| Customers requesting new Orders at a standard price list. | Customers can call customer service/telesales to place new orders against the standard price list. |
| Order against pre-negotiated agreement/ rate card/price list. | Customer can place orders against an already negotiated price list, rate card, or agreement. In this case, orders will be priced based on the pre-negotiated price. |

| Request for Order Creation | Detail |
|---|--|
| Add-on/change orders for previously purchased products. | Customers can place add-on orders, change orders, or renewal orders against previously purchased products /services/subscriptions. |


- [To create a new order from the Orders tab](#)
- [To create a new Order from the Account page](#)

To create a new order from the Orders tab

1. Click the **Orders** tab. A list of recent orders is displayed.
2. Click **New**. The New Order page is displayed.
3. Enter values for the fields listed in the following table:


| Field | Description |
|----------------------|--|
| Type | Click the drop-down and select the order type: New, Add-On, or Renewal. |
| Status | Click the drop-down and select Draft status. |
| Source | Click the drop-down and select Account . |
| Sold To | Type the name of the Account in the search box and select it or click the lookup icon and search and select the account for this order. |
| Price List | Type the name of the Price List in the search box and select it or click the lookup icon and search and select the relevant Price List for this order. |
| Order Date | Click this field and select a date for the order from the date picker. |
| Auto Activate Order? | Click the check box to automatically activate the order when it is accepted. |

4. Click **Save**. A new order is created.
5. Click the **Configure Products** button to open the Product Catalog. After finalizing the cart, order line items are created for the order.

 For step-by-step help customizing the Configure Products button to your business needs, and other tasks and information about creating and finalizing product configurations, refer to the *CPQ Administrator* and *User Guides*.

6. Click **Accept**. The following changes are made depending on whether or not you decided to automatically activate the order:
 - If **Auto Activate Order** = False
 - The status of Order and Order Line Items is set to "Pending."
 - Asset Line Items are created in the "Pending" status.
 - If "Ready for Billing Date" is set, billing schedules are generated.
 - If **Auto Activate Order** = True
 - The status of Order and Order Line Items is set to "Activated."
 - Asset Line Items are created in the "Activated" status.
 - If "Ready for Billing Date" is set, billing schedules are generated.

To create a new Order from the Account page

 In order to create an order from an Account, the Orders (Sold To) related list must be added to the page layout. For more information, refer to the *Order Management Administrator Guide*.

1. Go to the **Accounts** tab (All Tabs > Accounts). The Accounts home page is displayed.
2. Find the Account for which you want to place an order. Click the **Account Name**. The Account Details page is displayed.
3. Hover over the **Orders (Sold To)** link above the Order header and click **New Order**.
4. Enter values for the Order fields as described in the previous task. The Sold To field is already populated with the Account Name.

 Do not forget to ensure that the order is in "Draft" status.

5. Click **Save**.
6. Click the **Configure Products** button to open the Product Catalog. Refer to the *CPQ User Guide* for complete steps on configuring products, applying pricing, quantity, promotions, and discounts, and finalizing the cart. After finalizing the cart, order line items are created for the order.
7. Click **Accept**. The following changes are made depending on whether or not you decided to automatically activate the order:

- If **Auto Activate Order** = False
 - The status of Order and Order Line Items is set to "Pending."
 - Asset Line Items are created in the "Pending" status.
 - If "Ready for Billing Date" is set, billing schedules are generated.
- If **Auto Activate Order** = True
 - The status of Order and Order Line Items is set to "Activated."
 - Asset Line Items are created in the "Activated" status.
 - If "Ready for Billing Date" is set, billing schedules are generated.

Creating Orders from Quotes and Agreements

You can also create direct orders from the quote or agreement page, however, in most cases, you will want to follow the Auto-Create Order workflow for generating orders in this way. Refer to the *CPQ* and *CLM User Guides* for these tasks and information on Asset-Based Ordering (ABO).

Cloning an Order

Clone an order when you want to save time to create a new order with existing data, order line items, and cart configuration (this is called a "deep clone").

i Ensure that you or an administrator have added the Clone button to the Order page layout. Refer to the *Order Management Administrator Guide* for more details.

To clone an order

1. Go to the Order Detail page of the order you want to clone.
2. Click **Clone Order With Line Items**. A new order is created. The following changes are made:
 - The status of the order and order line items are set to "Draft."
 - The fields Read for Activation Date and Ready for Billing date on the order header, and Asset Line Item on the order line item is reset.
 - Product Configurations are cloned along with corresponding cart and cart line items. Cart line items can then be configured and changed similar to drafting a new order.

i Asset-Based Ordering (ABO) Line Items cannot be cloned using this process. For more information ABO, refer to the *CPQ User Guide*.

Managing In-Flight Order Changes and Cancellation

An In-Flight Order is defined as any order that has been confirmed by the customer but has not yet reached fulfillment. In-Flight order changes can only be made to orders that are not automatically activated on the confirmation. The two most common scenarios for In-Flight order changes are as follows:

- Amending an order: Amend an order when you need to add, cancel, or change order line items, change order quantities, pricings, and discounts, or modify order start/end date and/or change subscription dates. The previous version of the order is "Superseded" when the new order created by the amendment is accepted.
- Cancelling an order: Cancelling an order cancels the In-Flight order and its line items. After accepting cancellation of the order, the previous version of the order is "Superseded" and the new version is created with the "Cancelled" status.

You can use the Undo Amend Order and Undo Cancel Order actions at any time before either is accepted to revert all changes.

- In-Flight order changes and cancellation require you to enable certain Order System Properties. Refer to the *Order Management Administrator Guide* for steps.
- When In-Flight Order Change is enabled, an asset is only created on activation of its corresponding order line item.

Tracking Order Versions

Conga maintains a version history when you work with new or existing orders. The following fields on the order header are populated when you amend or cancel an order:

- **Previous Version:** The previous version of the respective order before the order was amended or cancelled
- **Next Version:** The next version of the original order that has been amended or cancelled or is in the process of being amended or cancelled
- **Original Order Number:** The order number of the original order that was placed (before any amendments).
- **Version Number:** The version number that is associated with the respective order. The number increments by 1 each time the order is amended.

Automating In-Flight Order Changes

You can automate In-Flight order changes by creating Order Workflow Rulesets. Order Workflow Rulesets comprise specific Order Workflow Rules that execute in-flight order changes when certain criteria are met. The following are the main triggering events that execute these rules:

- A quote is accepted and converted to an order.
- An agreement is activated and converted to an order.
- A direct order or order amendment is accepted.
- The "Ready for Workflow" field is set to true on the order header for orders in "Pending", "In Fulfillment," or "Partially Fulfilled" status. (If the field is not present, add it to the Order page layout.)
- The [execOrderWorkflow](#) API is called directly to manually trigger one or more workflow rules.

For more information on APIs, refer to the *Conga Order Management SOAP API Guide*.

Amending an Order

You can amend an order after it is accepted by the customer but before it has reached fulfillment. Amending an order creates a new version of the order to make and track changes and a reference to the previous version. The previous version of the order and order line items are "Superseded," once the work-in-progress (In Amendment) changes are confirmed on "Accept" of the "In Amendment" order.

You can make the following changes when you amend an In-Flight order:

- Change the start date or end date of subscription on one or more order line items that are in "Pending," "In Fulfillment," or "Partially Fulfilled" status.
- Make subscription date changes on the standalone item, bundles, and multiple charge line items.
- Cancel or add new line items, and change their quantities, pricing, and discounts.

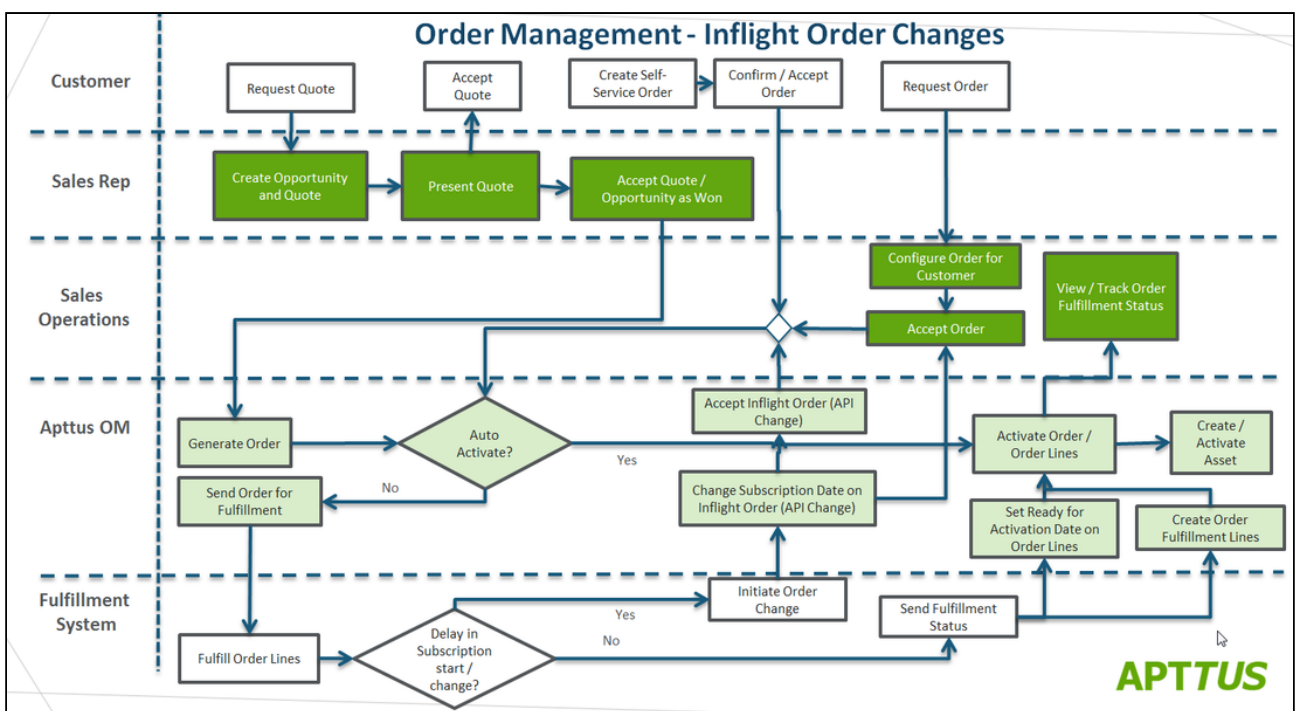
The change in subscription start and/or end date will recalculate the price for the line item or bundle after the changes are applied.

The changes to the order line items are also reflected in the corresponding assets. When In-Flight Order Changes is enabled, assets are only created on activation of the order line item.

Note: Any order configured directly or generated from a quote, agreement, or E-Commerce can be amended.

To change order line item quantities, the field "Is Quantity Modifiable" must be enabled at the line item level for product configuration.

The following diagram demonstrates the In-Flight Order Changes workflow.




i If you do not see the Amend Order button then In-Flight Order changes may not have been configured for your product. You or an administrator must complete the proper configuration to continue. Refer to the topic [Configuring In-Flight Order Changes and Cancellation](#) in the *Order Management Administrator Guide*.

To amend an order

1. Navigate to the order from the quote, agreement, account, or Orders tab. The Order Detail page is displayed.

2. Verify that the order is in "Pending," "In Fulfillment," or "Partially Fulfilled" status and click **Amend Order**. The Cart page is displayed.
3. Make changes to the product configuration based on your requirements. To cancel any order line items:
 - a. Select one or more order line items in the cart.
 - b. Click the More menu under the Reprice/Finalize button.
 - c. Click **Cancel Order Lines**. The cart is refreshed to display the remaining line items.
4. **Reprice** and **Finalize** the cart. Changes made to cart line items are copied to the order line items. Order Management creates a new version of the order in "In Amendment" status. The status for the previous version is changed to "Being Amended."
5. After approval of the in-flight order changes, click **Accept** to confirm the amended order. The status of the new order is changed to "Pending." That status of the previous version is changed to "Superseded."
6. When all amendments are complete, [activate the order](#).

-  • When a new line item is added to the cart associated with an amended order, the order line item status defaults to "Draft." When the order status is updated based on the combined status of order line items, the order is set to "Draft" status when there is at least one order line item in "Draft" status.
- Cancel Order Lines is not available from the cart menu if it is not configured properly in Config Settings. Refer to [Configuring In-Flight Order Changes and Cancellation](#) in the *Order Management Administrator Guide* for step-by-step information on configuring In-Flight order settings.

To undo an amend order

1. Navigate to the amended order.
2. To roll back the amendment and all order line item changes, click **Undo Amend Order**. The order is reverted to the previous version with its original status ("Pending," "In Fulfillment," or "Partially Fulfilled").

Cancelling an Order

You can cancel an order after it is accepted by the customer but before it is released to fulfillment systems. When you cancel an order, the status of the order and its line items is changed to "Pending Cancellation" and a new version is created. The previous order version status is set to "Being Cancelled." When you decide to cancel an order, you must accept the cancellation of the order to change the order and its line items from "Pending Cancellation"

to "Cancelled". Any order amount that was rolled up to the Order Amount on the order header is subtracted from the total.

To cancel an order

1. Navigate to the order from the quote, agreement, account, or Orders tab. The Order Detail page is displayed.
2. Verify that the order is in "Pending" or "In Fulfillment" status and click **Cancel Order**. A new version of the order is created with the status "Pending Cancellation." The status for the previous version is changed to "Being Cancelled."
3. Click **Accept** to confirm order cancellation. The status of the cancelled order and its line items is changed to "Cancelled." That status of the previous version is changed to "Superseded."

To undo a cancelled order

1. Navigate to the cancelled order.
2. To roll back cancellation of the order, click **Undo Cancel Order**. The order is reverted to the previous version with its original status ("Pending" or "In Fulfillment"). The order amount is recalculated and displayed in the Order Amount field on the order header.

Creating Order Workflows to Automate In-Flight Order Changes

When you are working with a large number of orders and you must create rules for different business scenarios (for example, pro-rating subscriptions, or cancelling orders due to a delay in provisioning), you can use Order Workflows to automate in-flight changes to orders based on certain criteria.

Order Management allows you to automate in-flight order changes and cancellation using Order Workflow rulesets and rules. Creating Order Workflows comprises two main tasks. First, you define a **Workflow Ruleset** to provide the business and trigger context, and any criteria related to the trigger. You then create a **Workflow Rule** and **Workflow Rule Entries** within that ruleset to instruct the system which action or actions to execute based on further criteria.

i You can only create one workflow rule per ruleset. Each workflow rule can contain one or more workflow rule entries.

When a workflow ruleset is triggered, the specified workflow rule executes based on the criteria that triggers one or more of its rule entries. For example, you could create a workflow rule that automatically amends the order to change the subscription date of one or more line items when the agreement is accepted.

Executing Order Workflow Rulesets

Order Workflow Rulesets trigger based on any of the following events:

- Automatic:
 - An order is created from a quote or agreement.
 - A partial order is created from a price agreement.
 - An order is accepted.
- Manual: Ready for Workflow is enabled and the order record is saved.

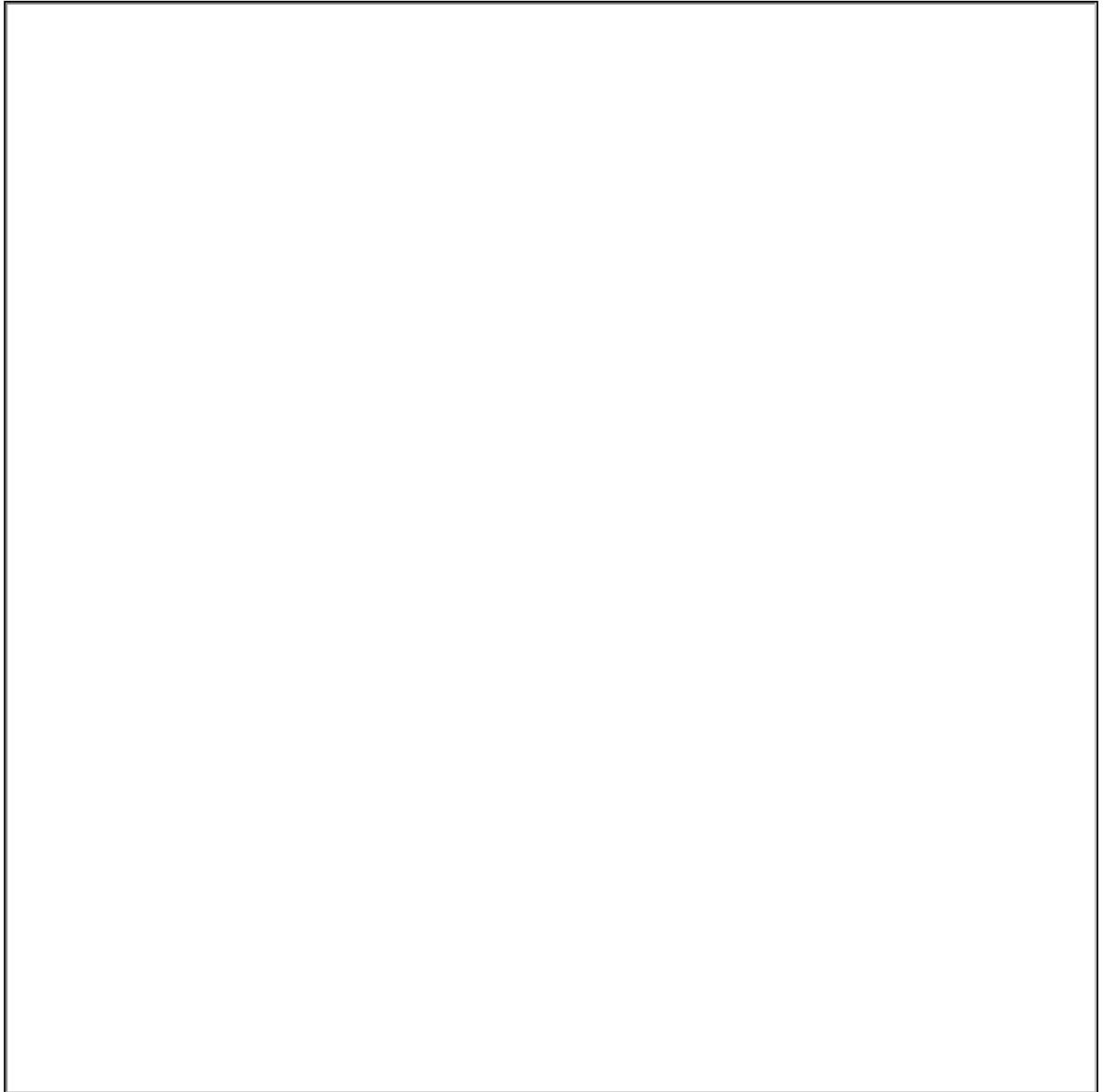
i After a workflow ruleset is executed, the Ready for Workflow flag is disabled.

Perform the following tasks to set up a Workflow ruleset for automating in-flight order changes and cancellation.

- [To create an Order Workflow Ruleset](#)
- [To define criteria for the Workflow Ruleset](#)
- [To create an Order Workflow Rule](#)
- [To create an Order Workflow Rule Entry](#)
- [To define criteria for line item workflow rule entries](#)
- [To define the input for line item workflow rule entries](#)
- [Use Case: Automatic Subscription Rollover](#)

To create an Order Workflow Ruleset

1. Click **All Tabs** (☰) > **Order Workflow Rulesets** > **New** (in Salesforce Lightning, go to **App Launcher** > **Order Workflow Rulesets** > **New**). The New Order Workflow Ruleset page is displayed.



2. Enter the **Ruleset Name**.
3. Click the **Type** drop-down and select **API Execution**.
4. Enter a **Sequence** value (begins at "1"). This defines the sequence in which to execute the ruleset when an event triggers more than one ruleset.

 To avoid errors, specify a unique sequence value for every workflow ruleset.

5. Click the **Business Context** Type drop-down and select the context object (default is "Order").
6. Depending on your requirements, click the **Trigger Context Type** drop-down and select the appropriate trigger:

- **Order**
 - **Order Fulfillment**
7. Depending on your requirements, add the appropriate **Trigger Event(s)** from the **Available** box to the **Chosen** box.
 8. Select the **Active** checkbox.
 9. Click **Save**. The ruleset is created.

To define criteria for the Workflow Ruleset

Perform the following task to define the triggering criteria for a workflow ruleset.

1. From the Workflow Ruleset Detail page, click **Criteria**. The Order Workflow Ruleset Edit page for Criteria Edit is displayed.
2. Click **Edit**. The Criteria Edit page is displayed.

Order Workflow Ruleset Edit
Order Automation Tech Pubs Test

Criteria Edit

| Field | Operator | Map To | Value | Offset |
|----------|----------|----------|-------|--------|
| --None-- | --None-- | --None-- | | + |

[Advanced Options](#)

OK Cancel

3. Click the **Field** drop-down and select a field on the Order object.
4. Click the **Operator** drop-down and select an operator for the expression.
5. Click the **Map To** drop-down and select one of the following:
 - **None**: Evaluate the expression based only on the Field > Operator > Value defined.
 - **Function**: Date fields only. Specify days or months and the offset value.
 - **Order**: Map the field on the Order object to the value of another standard or custom field on the Order object.
6. Enter or select a **Value** for the expression.
7. Click the **Add Row** (+) icon to add another expression. Repeat steps 3 – 6 to define additional criteria.
8. Click **Advanced Options** to add filter logic if necessary (for example: "(1 OR 2) AND 3").
9. Click **OK** to save the criteria and return to the Workflow Ruleset Detail page.

To create an Order Workflow Rule

i You can create only one workflow rule per ruleset. Each workflow rule can contain one or more workflow rule entries.

1. Click **New Order Workflow Rule** from the Order Workflow Ruleset page. The New Order Workflow Rule page is displayed.
2. Enter a **Sequence** value (begins at "1").
3. Click the **Action** drop-down and select one of the following actions to be executed when the Workflow Ruleset is triggered:
 - **Amend Order**: Automatically amend the order. The order must be manually accepted after it is amended.
 - **Amend and Accept Order**: Automatically amend and accept the order.
 - **Cancel Order**: Automatically cancel the order. The user must manually accept the order cancellation.
 - **Cancel and Accept Order**: Automatically cancel and accept the order cancellation.
4. (Optional) Enter a **Description**.
5. Click the lookup icon and select the ruleset to associate with the rule (automatically filled when the rule is created from a ruleset detail page.)
6. Click **Save**.

To create an Order Workflow Rule Entry

1. Click **New Order Workflow Rule Entry** from the Order Workflow Rule page. The New Order Workflow Rule Entry page is displayed.
2. Enter a **Sequence** value (begins at "1"). This defines the sequence in which to execute the workflow rule entries when the workflow ruleset is triggered.
3. Click the **Context Type** drop-down and select **Order Line Item**.
4. Click the **Action** drop-down and select one of the following actions to be executed when the rule entry criteria is fulfilled:
 - **Amend Order Items**: Amend the order line item specified by the rule entry criteria.
 - **Cancel Order Items**: Cancel the order line item specified by the rule entry criteria.
5. Click **Save**.

To define criteria for line item workflow rule entries

Perform the following task to define the criteria that determine which Order Line Items are amended or cancelled when the workflow rule is executed.

1. From the Order Workflow Rule Entry page, click **Item Criteria**. The Order Workflow Rule Entry Edit page for Criteria Edit is displayed.
2. Click **Edit**. The Criteria Edit page is displayed.

| Field | Operator | Map To | Value | Offset |
|----------|----------|----------|-------|--------|
| --None-- | --None-- | --None-- | | + |

[Advanced Options](#)

OK Cancel

3. Click the **Field** drop-down and select a field on the Order Line Item object. Standard and custom fields on the Order Line Item object are available for selection.
4. Click the **Operator** drop-down and select an operator for the expression.
5. Click the **Map To** drop-down and select one of the following:
 - **None**: Evaluate the expression based only on the Field > Operator > Value defined.
 - **Function**: Date fields only. Specify days or months and the offset value.
 - **Order Line Item**: Map the field on the Order Line Item object to the value of another standard or custom field on the Order Line Item object.
6. Enter or select a **Value** for the expression.
7. Click the **Add Row** (+) icon. Repeat steps 3 - 6 to define additional criteria.
8. Click **Advanced Options** to enter filter logic (for example, "(1 OR 2) AND 3").
9. Click **OK** to save the Item Criteria and return to the Order Workflow Rule Entry page.

To define the input for line item workflow rule entries

Perform this task to define the action taken on one or more Order Line Items defined by the Item Criteria when the workflow rule is executed.

1. From the Order Workflow Rule Entry page, click **Item Input**. The Order Workflow Rule Entry Edit page for Input Edit is displayed.
2. Click **Edit**. The Input Edit page is displayed.

| Field | Operator | Map To | Value | Offset |
|----------|----------|----------|-------|--------|
| --None-- | equal to | --None-- | | + |

3. Click the **Field** drop-down and select a field on the Order Line Item object.
4. Click the **Operator** drop-down and select an operator for the expression.
5. Click the **Map To** drop-down and select one of the following:
 - **None:** Evaluate the expression based only on the Field > Operator > Value defined.
 - **Function:** Date fields only. Specify days or months and the offset value.
 - **Order Line Item:** Map the field on the Order Line Item object to the value of another standard or custom field on the Order Line Item object.
6. Enter or select a **Value** for the expression.
7. Click the **Add Row** (+) icon. Repeat steps 3 - 6 to add another input.
8. Click **Advanced Options** to enter filter logic (for example, "(1 OR 2) AND 3").
9. Click **OK** to save the Item Input and return to the Order Workflow Rule Entry page.

Use Case: Automatic Subscription Rollover

Description: This use case describes how to create a Workflow Ruleset to handle automatic subscription rollover for in-flight order changes.

Prerequisite: In-Flight Order Changes and Cancellation must be configured in Order Management Settings. Refer to the *Order Management Administrator Guide* for more information.

You are tasked with modifying an in-flight order for subscription products. The product originally sold from January 1st to December 31st at a \$1200 price point for a quantity of 10. However, fulfillment order the order is complete on January 31st. You need the subscription term to roll over, meaning that the order must be automatically modified to start on February 1st and end on January 31st the next year.

To create the Order Workflow Ruleset

1. Click **All Tabs** (☰) > **Order Workflow Rulesets** > **New** (in Salesforce Lightning, go to **App Launcher** > **Order Workflow Rulesets** > **New**). The New Order Workflow Ruleset page is displayed.

2. Enter "Autoroll" as the **Ruleset Name**.
3. Click the **Type** drop-down and select **API Execution**.
4. Enter "1" for the **Sequence**.
5. From the the **Business Context Type** drop-down, select **Order**.
6. Click the **Trigger Context Type** drop-down and select **Order**.
7. Select the **Active** check box.
8. Click **Save**. The ruleset is created.

To create the Order Workflow Rule

1. Click **New Order Workflow Rule** from the Order Workflow Ruleset page. The New Order Workflow Rule page is displayed.
2. Enter "1" for the **Sequence**.
3. Click the **Action** drop-down and select **Amend and Accept Order**.
4. Enter "Auto Rollover Subscription on Fulfillment" in the **Description** field.
5. Click the lookup icon and select the **Autoroll** ruleset you created in the first task.
6. Click **Save**.

To create the Order Workflow Rule Entry and define criteria

1. Click **New Order Workflow Rule Entry** from the Order Workflow Rule page. The New Order Workflow Rule Entry page is displayed.
2. Enter "1" for the Sequence value.
3. Click the **Context Type** drop-down and select **Order Line Item**.
4. Click the **Action** drop-down and select **Amend Order Items**.
5. Click **Save**.
6. From the Order Workflow Rule Entry page, click **Item Criteria**. The Order Workflow Rule Entry Edit page for Criteria Edit is displayed.
7. Click **Edit**. The Criteria Edit page is displayed.
8. Click the **Field** drop-down and select **Status**.
9. Click the **Operator** drop-down and select **equal to**.
10. Enter "Fulfilled" for the **Value**.
11. Click **OK** to save the Item Criteria and return to the Order Workflow Rule Entry page.

To define input for the Order Workflow Rule Entry

For the input, perform the following tasks to shift the start date by a fixed number of days while keeping the end date the same.

1. From the Order Workflow Rule Entry page, click **Item Input**. The Order Workflow Rule Entry Edit page for Input Edit is displayed.

2. Click **Edit**. The Input Edit page is displayed.
3. Click the **Field** drop-down and select **Start Date**.
4. Click the **Operator** drop-down and select **equal to**.
5. Click the **Map To** drop-down and select **Function**.
6. Click the drop-down to the right of **Map To** and select **Today**.
7. Click the **Add Row (+)** icon to add another line.
8. On the new row, click the **Field** drop-down and select **End Date**.
9. Click the **Operator** drop-down and select **equal to**.
10. Leave the **Map To** value as **--None--** and leave the **Value** field blank.
11. Click **OK**.

Result:

- The "Auto Roll" Order Workflow Ruleset is created.
- The corresponding rule and rule entry is created. Criteria and inputs are specified to ensure that when a subscription order line item is fulfilled, the start term is modified to automatically rollover the subscription term without changing the end date.

Sample Scenarios for Amending Orders Using Workflow Rules

Refer to the following sample scenarios for amending orders using workflow rules.

1. Auto Roll
 - a. To shift the start date and end date of the order by a fixed number of days/ months while keeping the length of the subscription term same. In this case, just set the "Start Date" to a new date, set "End Date to Null" and do not modify the term using workflow rule entries.
 - b. For example, Start Date = Today, End Date = Null.
 - c. For example, Start Date = Provisioning Date, End Date = Null.

| Field | Operator | Map To | Value | Offset | |
|------------|----------|----------|-------|----------|-----|
| Start Date | equal to | Function | Today | --None-- | AND |
| End Date | equal to | --None-- | | | + - |

OK Cancel

2. Pro Rata
 - a. To move the start date of the subscription while keeping the end date unchanged. In this case, just specify the new start date and do not set End Date or Term.

- b. For example, Start Date = Provisioning Date + 2 Days.
- c. For example, Start Date = Provisioning Date, End Date = Previous End Date.

| Field | Operator | Map To | Value | Offset |
|------------|----------|-----------------|---------------------------------|---------------------|
| Start Date | equal to | Order Line Item | Provisioning Date(custom field) | Offset (+) Days 2 + |

OK Cancel

3. Quantity Changes

- a. To increase/decrease ordered quantity based on provisioning fulfillment inputs.
- b. For example, Quantity = InventoryAvailableQuantity.
- c. For example, Quantity = Provisioned Quantity.

| Field | Operator | Map To | Value | Offset |
|----------|----------|-----------------|----------------------------------|--------|
| Quantity | equal to | Order Line Item | Available Quantity(Custom field) | + |

OK Cancel

4. Cancelling Order Lines

- a. To cancel order lines as inventory is not available for fulfillment. In this case, set the status to "Pending Cancellation".

| Field | Operator | Map To | Value | Offset |
|--------|----------|----------|----------------------|--------|
| Status | equal to | --None-- | Pending Cancellation | + |

OK Cancel

Working With Customer Price Agreements

When business needs cannot be met by converting a quote or contract into a single order, you can use Order Management to create partial orders against a negotiated **Price Agreement**. In this case, when the quote is accepted or the contract is finalized, a **Contract Price List** is generated that is part of the Price Agreement. Users can then create partial orders to fulfill quantities at the negotiated price for product and service line items (on the quote or contract). Using Price Agreements, you can also track commitment compliance – how many line items were initially committed to the order initially versus how many have already been ordered or fulfilled – so that you can then take action based on that information.

Contract Price Lists are created from a quote or agreement where Intent = Price Agreement. You can then create partial orders from the line items stored in that Price List.

The Price List and Price List Item (PLI) objects store line item information from the quote or agreement in the following ways:

- For PLIs with the list price, the new net unit price is copied to the contract price field of the new PLI.

- For PLIs with matrix pricing (recurring), the matrix is copied from the original PLI to the new PLI.
- For PLIs with matrix pricing (usage), the matrix is copied from the usage price tiers of the line item to the new PLI.

When a partial order is generated, product attributes and related line information is copied from the source line item (quote or agreement) to the target line item:

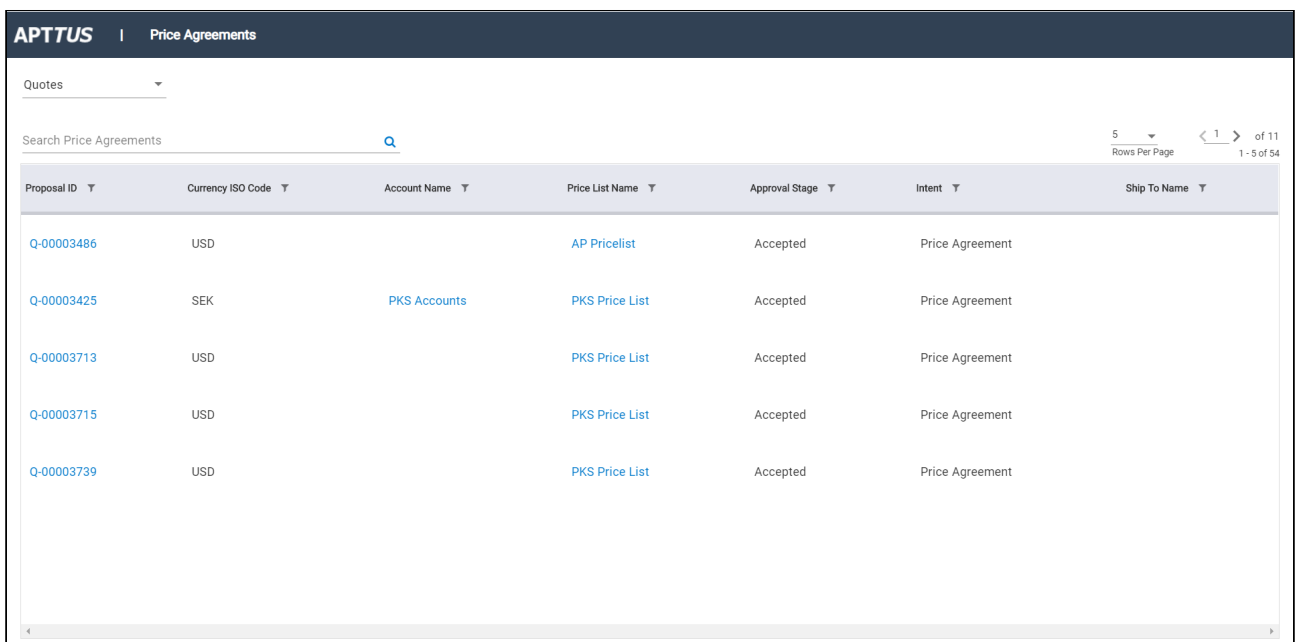
- Quote to order: When attributes are present on the quote line items, the order is created from the quote, and CopyBundleConfigurationFromSource = true, the attributes are copied to the order line items.
- Agreement to order: When attributes are present on the agreement line items, the order is created from the agreement, and CopyBundleConfigurationFromSource = true, the attributes are copied to the order line items.

Creating partial orders from Price Agreements allows you to:

- Select one or more line items and the subset of quantity
- Select items from one or multiple quotes or contracts
- Track committed quantity and ordered quantity
- Track commitment compliance

Navigating the Price Agreements User Interface

You can create partial orders using the Price Agreements UI. Click the **Price Agreement** tab to open the Price Agreements page.




The screenshot shows the APTTUS Price Agreements user interface. At the top, there is a header with the APTTUS logo and the text 'Price Agreements'. Below the header, there is a search bar labeled 'Search Price Agreements' with a magnifying glass icon. To the right of the search bar, there is a dropdown menu for 'Rows Per Page' set to '5' and a pagination indicator showing '1 of 11' and '1 - 5 of 54'. The main content area is a table with the following columns: Proposal ID, Currency ISO Code, Account Name, Price List Name, Approval Stage, Intent, and Ship To Name. The table contains five rows of data, all with an 'Accepted' approval stage and 'Price Agreement' intent.

| Proposal ID | Currency ISO Code | Account Name | Price List Name | Approval Stage | Intent | Ship To Name |
|-------------|-------------------|--------------|-----------------|----------------|-----------------|--------------|
| Q-00003486 | USD | | AP Pricelist | Accepted | Price Agreement | |
| Q-00003425 | SEK | PKS Accounts | PKS Price List | Accepted | Price Agreement | |
| Q-00003713 | USD | | PKS Price List | Accepted | Price Agreement | |
| Q-00003715 | USD | | PKS Price List | Accepted | Price Agreement | |
| Q-00003739 | USD | | PKS Price List | Accepted | Price Agreement | |

The default view of the Price Agreements page displays a list of Quotes with Intent=Price Agreement. Click the drop-down from the upper left-hand corner of the page to switch between **Quotes** and **Agreements**.

You can perform the following actions on the Price Agreements page:

- Use the Search Price Agreements type ahead to filter the list of price agreements by keyword.
- Click the filter icon (▼) next to a column heading to filter the list of price agreements by that field.
- Click a column heading to sort the list of price agreements by that column.
- Click the drop-down in the upper right-hand corner to change the number of price agreements displayed in the list per page.
- Use the pagination controls in the upper right-hand corner to navigate the list of price agreements.


 If you cannot locate the Price Agreements tab, then it is not configured for your org. Refer to the *Order Management Administrator Guide* for configuration steps. Administrators can also configure which columns are displayed in the list of price agreements.

Creating Partial Orders from Price Agreements

Perform the following task to create a partial order from a Price Agreement (quote or agreement).

To create a partial order

1. Click the **Price Agreement Tab**. The Price Agreements page is displayed.
2. Search for and click the **Proposal/Agreement ID**. The Price Agreement Detail page is displayed. The Price Agreement Line Items tab is displayed under the price agreement summary. Each Price List item displays, start date, end date, the negotiated quantity, remaining quantity to order, and the quantity to order.
3. Use the Search Products type ahead to filter the list of line items.
4. Enter the **Qty to Order** for a line item. After you enter any quantity, the **Create Partial Order** button is enabled.
5. Click **Create Partial Order**. The Create Sales Order page is displayed.
6. Enter values in the fields displayed on the page to create the partial order.
7. Click **Create Order**. The Price Agreement Detail page is displayed. The Order List is displayed with the newly-created partial order and any other partial orders made against this price agreement.
8. Click the Home icon (🏠) to return to the Price Agreements page. If required, repeat this task to create another partial order.

 Refer to the *Order Management Administrator Guide* for steps to configure which fields are displayed in the partial order form, price agreement list, and order list.

Rolling Up Order Line Item Information to Quote and Agreement Line Items

Prerequisite: Order Line Item must be linked to Quote Line Item or Agreement Line Item.

Information is rolled up from the order line items to the quote or agreement line items when a partial order is created from the quote or agreement. For example: Whenever there is a change in the ordered line item for a given agreement line item, the Ordered Quantity is updated. If there is 1 ordered line item for a given agreement line item, the Ordered Quantity is 1. As per this feature:

- Quantity from the order line item is rolled up to the ordered Quantity on the proposal line item.
- Fulfilled quantity is rolled up from the order line item to the fulfilled quantity on the agreement line item when the fulfilled quantity is updated on the order line item
- Fulfillment status is updated on the agreement line item when the fulfilled quantity is updated on the order line item as stated below:

- a. Fulfilled Quantity = 0: Not Fulfilled
 - b. Fulfilled Quantity < Quantity and > 0 : Partially Fulfilled
 - c. Fulfilled Quantity > = Quantity: Fulfilled
- Order line items in all status except (Cancelled, Superseded, Being Amended, Being Cancelled) are rolled up.

Splitting an Order

This feature allows you to divide an existing order (parent order) into multiple orders (child orders) during the provisioning process. You can split any large order that was created from a Quote, Agreement, or Direct order that has not been amended.

Using this feature you can:

- Split the order on the initiation of acceptance based on customer needs.
- Split the order post creation and before activation using a select set of criteria.
- Split the order before it is activated and track the original order id on the split order.
- Split the order manually by selecting the order lines.
- Amend or cancel the orders that have been split.

Consider the following scenarios when deciding to split an order:

- You have a requirement for handling orders comprising different products.
- Splitting the order based on product availability and delivery timeline.
- Splitting the order to expedite the provisioning process.

Use one of the following methods to perform order split:

1. **Manual Split:** You can perform the manual split only through API. For more information, refer to the [Splitting an Order using API](#) topic in *Order Management SOAP API Guide*.
2. **Auto Split on Accept:** To automate the order split using the criteria on order acceptance. You can perform split automation using one of the following criteria.
 - a. Using Global Criteria
 - b. Using Select workflow Level Criteria
3. **Order Workflow Ruleset**

Auto Split on Accept

You can automate the order split on acceptance using one of the following criteria.

- **Splitting the order on accepting using Global criteria:** This splits the order on Quote acceptance, Agreement activation, or order acceptance based on the global criteria defined in the Order System Properties.
- **Splitting the order on accepting using the selected workflow level criteria:** This split the specific order on Quote acceptance, Agreement activation, or Order acceptance based on the selection of order lines defined in the order workflow rule criteria.

Splitting the order on accepting using global criteria

To split the order using global criteria:

- Create an Order Workflow Ruleset and Order Workflow Rule. Refer to [Creating Order Workflows to Automate In-Flight Order Changes](#) for step-by-step instructions.

Note:

- When creating an Order Workflow Ruleset, ensure that **Record Updated** is chosen as a Trigger Event.
- If the Order Workflow Rule Entry is not defined, the split order action is performed based on the Global criteria defined in the Custom settings > Order System Properties.

Usecase: Using Global criteria

Description: This use case describes how to create a Workflow Ruleset to handle the split order using global criteria.

A large manufacturing company that sells hardware and software products. Its backend process is handled in different fulfillment systems for the product lines. Sales representative quotes for 10 hardware items and 5 software items for one enterprise in a single quote. But, expecting to generate two individual orders for each Line of Business for fulfillment.

To create the Order Workflow Ruleset:

1. Click **All Tabs (+) > Order Workflow Rulesets > New** (in Salesforce Lightning, go to **App Launcher > Order Workflow Rulesets > New**). The New Order Workflow Ruleset page is displayed.
2. Enter "Split Rule" as the Ruleset Name.
3. Click the Type drop-down and select **API Execution**.
4. Enter "1" for the Sequence.

5. From the Business Context Type drop-down, select **Order**.
6. Click the Trigger Context Type drop-down and select **Order**.
7. Add the Order Updated Trigger Event(s) from the Available box to the Chosen box.
8. Select the Active checkbox.
9. Click Save. The ruleset is created.

To create the Order Workflow Rule:

1. Click New Order Workflow Rule from the Order Workflow Ruleset page. The New Order Workflow Rule page is displayed.
2. Enter "1" for the Sequence.
3. Click the Action drop-down and select Split Order.
4. Enter "Split order on accepting using <criteria name>" in the Description field.
5. Click the lookup icon and select the Split Rule ruleset you created in the first task.
6. Click Save.
7. Go to Order custom settings, go to **Setup > Custom Settings > Order System Properties**, and enter **Line_of_Business__c** in the Split Order Criteria fields.

Refer to the following screens for the sample setup for the given use case.

The screenshot displays two screenshots from the Order Management system. The top screenshot shows the 'Order Workflow Ruleset Detail' for 'Split rule-ST'. The ruleset is configured with the following details:

- Ruleset Name: Split rule-ST
- Type: API Execution
- Sequence: 1
- Business Context Type: Order
- Trigger Context Type: Order
- Trigger Event: Record Created
- Active:
- Owner: CM.QA.(Summer.20)
- Created By: CM.QA.(Summer.20), 6/10/2020 11:47 PM
- Last Modified By: AssetsQA.ABO, 6/21/2020 9:59 PM

The bottom screenshot shows the 'Order Workflow Rule Detail' for 'OR-0000020'. The rule is configured with the following details:

- Rule Number: OR-0000020
- Sequence: 1
- Action: Split Order
- Description: Sample
- Created By: CM.QA.(Summer.20), 6/10/2020 11:48 PM
- Ruleset: Split rule-ST
- Number Of Entries: 0
- Last Modified By: AssetsQA.ABO, 6/22/2020 2:42 PM

Order System Properties Edit

Provide values for the fields you created. This data is cached with the application.

Edit Order System Properties

Save Save & New Cancel

Order System Properties Information

Name System Properties ⓘ

Initiate Billing On Order Activation ?

Create Asset On Order Activation ?

Enable Inflight Changes And Cancellation ?

Agreement Fields For Partial Order2

Agreement Fields For Partial Order Aptus__Account_c,Aptus__Account_

Agreement Item Fields For Partial Order2

Agreement Item Fields For Partial Order Aptus__AgreementId_c,Aptus__Produ

Quote Fields For Partial Order2

Quote Fields For Partial Order Aptus_Proposal_Account_c,Aptus_P

Quote Item Fields For Partial Order2 Aptus_QPConfig_EndDate_c,Aptus

Quote Item Fields For Partial Order

Split Order Criteria Fields Line_of_Business_c

Split Order Threshold

Result: The above setup triggers the parent order first and then splits into two new child orders. The parent order remains empty.

Splitting the order on accepting using the selected workflow level criteria

To split the order using select workflow level criteria:

- Create an Order Workflow Ruleset and Order Workflow Rule. Refer to [Creating Order Workflows to Automate In-Flight Order Changes](#) for step-by-step instructions.

Use Case: Splitting an order using the selected workflow level criteria

A large Hitech company sells perpetual licenses, software, and professional services. Their professional services deliverables are managed using a PSA tool. Sales representative quotes for 100 different licenses, 10 software, 3 professional services offerings in a single quote. On quote acceptance, the expectation is that the professional service items should carve out as a separate order for integration with the PSA system.

To create the Order Workflow Ruleset:

1. Click All Tabs (+) > Order Workflow Rulesets > New (in Salesforce Lightning, go to App Launcher > Order Workflow Rulesets > New). The New Order Workflow Ruleset page is displayed.
2. Enter "Split Rule" as the Ruleset Name.
3. Click the Type drop-down and select API Execution.
4. Enter "1" for the Sequence.
5. From the Business Context Type drop-down, select Order.
6. Click the Trigger Context Type drop-down and select Order.
7. Select the Active checkbox.
8. Click Save. The ruleset is created.

To create the Order Workflow Rule:

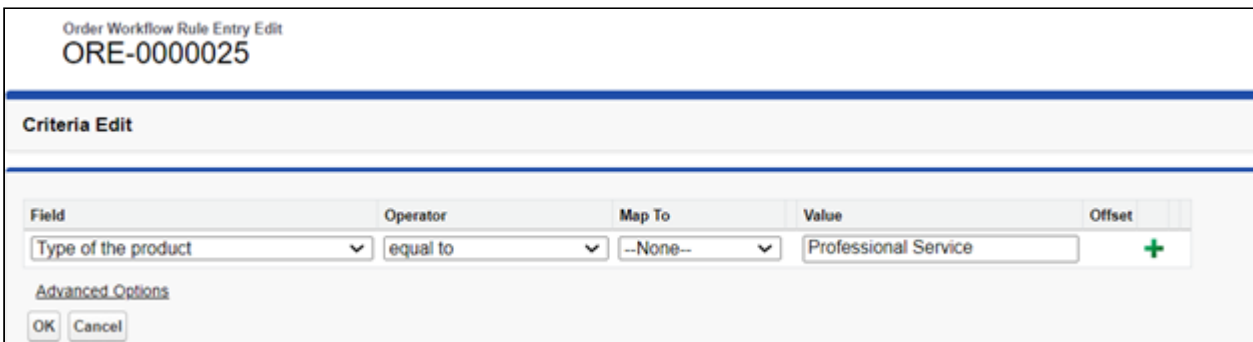
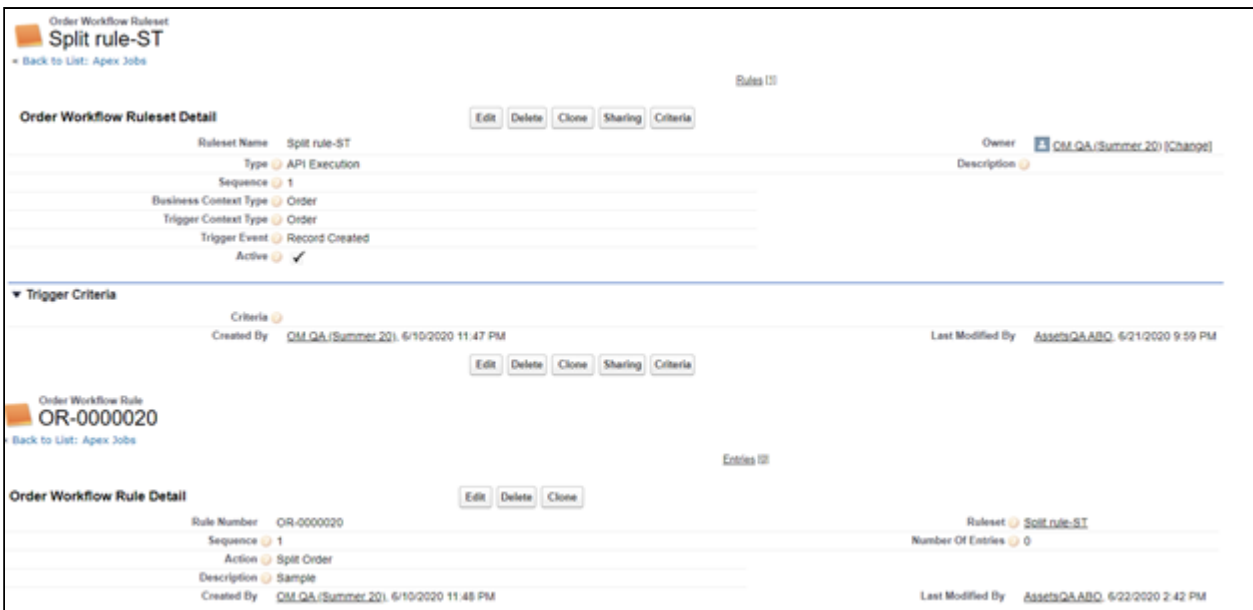
1. Click New Order Workflow Rule from the Order Workflow Ruleset page. The New Order Workflow Rule page is displayed.
2. Enter "1" for the Sequence.
3. Click the Action drop-down and select Split Order.
4. Enter "Split order on accepting using select workflow level criteria" in the Description field.
5. Click the lookup icon and select the Split Rule ruleset you created in the first task.
6. Click Save.

To create the Order Workflow Rule Entry and define criteria:

1. Click New Order Workflow Rule Entry from the Order Workflow Rule page. The New Order Workflow Rule Entry page is displayed.
2. Enter "1" for the Sequence value.
3. Click the Context Type drop-down and select Order Line Item.
4. Click **Save**.

5. From the Order Workflow Rule Entry page, click Item Criteria. The Order Workflow Rule Entry Edit page for Criteria Edit is displayed.
6. Click **Edit**. The Criteria Edit page is displayed.
7. Click the Field drop-down and select Type of Product.
8. Click the Operator drop-down and select **equal to**.
9. Click the Map To drop-down and select None.
10. Click the Value drop-down and select Professional Service.
11. Click **OK** to save the Item Criteria and return to the Order Workflow Rule Entry page.

Refer to the following screens for the sample setup for the given use case.



Result:

- The "Split Rule" Order Workflow Ruleset is created.
- The corresponding rule and rule entry is created. Criteria and inputs are specified to create a child order with professional service products. The license and software lines remain on the parent order.

Creating Order Workflows to Automate Split Order

The Order Workflow Rulesets enables the automatic triggering of the split action on quote acceptance, Agreement Activation (Agreement Flow), or Order acceptance (Direct Order Flow) for a specific selection of order lines as defined by the criteria on the workflow rule.

Perform the following tasks to set up a Workflow Rule for automating the split order process.

- [To create an Order Workflow Rule for a Split Order](#)
- [To create an Order Workflow Rule Entry](#)

To create an Order Workflow Rule for a Split Order

i You can create only one workflow rule per ruleset. Each workflow rule can contain one or more **Workflow Rule Entries**.

1. Create an Order Workflow Ruleset and define the criteria for the workflow ruleset. Refer to [To create a new Order Workflow Ruleset](#) and [To define criteria for the Workflow Ruleset](#) for step by step instructions.
2. Click **New Order Workflow Rule** from the Order Workflow Ruleset page. The New Order Workflow Rule page is displayed.
3. Enter a **Sequence** value (begins at "1").
4. Click the **Action** drop-down and select **Split Order**.
 - **Split Order**: Automatically split an order as per the flow defined in the order workflow rule.
5. Enter a **Description** (Optional).
6. Click the lookup icon and select the ruleset to associate with the rule (automatically filled when the rule is created from a ruleset detail page.)
7. Click **Save**.

To create an Order Workflow Rule Entry

1. Click **New Order Workflow Rule Entry** from the Order Workflow Rule page. The New Order Workflow Rule Entry page is displayed.
2. Enter a **Sequence** value (begins at "1"). This defines the sequence in which to execute the workflow rule entries when the workflow ruleset is triggered.

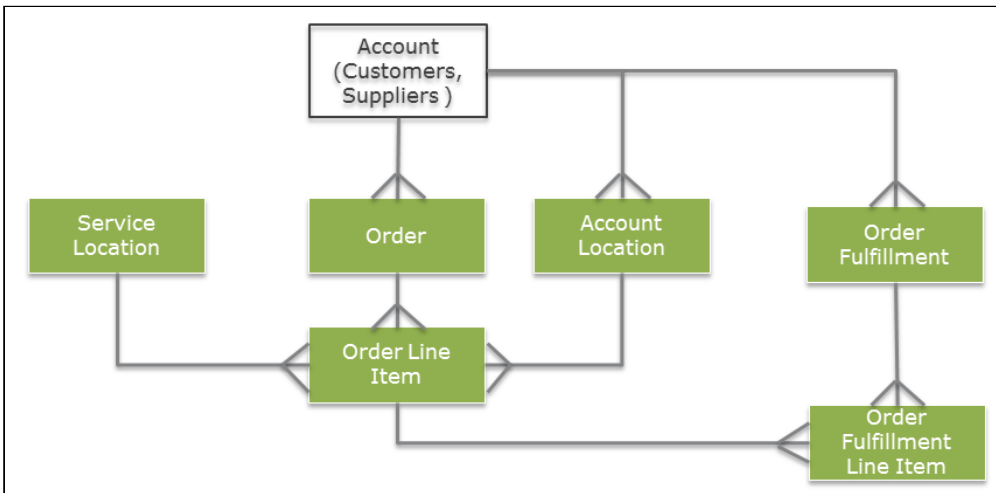
3. Click the **Context Type** drop-down and select **Order Line Item**.
4. Click **Save**.

Working With Distributed Order Fulfillment

Order fulfillment refers to the order life cycle process from the time the order is confirmed by the customer to the time the order is activated. This includes the following key processes:

- Order release to fulfillment systems
- Order fulfillment status tracking with partial order fulfillment
- Order and Order Line Item activation

The following diagram provides an overview of the data Conga captures to manage the order and order line items through fulfillment and activation.



The following objects are used in the distributed order fulfillment process to track and fulfill order line items from the beginning of order fulfillment to activation.

| Data | Details |
|------------------|--|
| Account Location | Specifies the buyer's location where the products or services are installed. This is an optional field that can be used as part of an integration with fulfillment systems. |
| Service Location | Specifies the warehouse, plant, factor, fulfillment center, service center of the seller from which the products or services are delivered. This is an optional field that can be used as part of an integration with fulfillment systems. |

| Data | Details |
|-----------------------------|--|
| Order Fulfillment | Order Fulfillment captures the details of the fulfillment coming from the fulfillment systems. A given fulfillment is for one or more orders for a given account for fulfillment completed on a specified date. |
| Order Fulfillment Line Item | Order Fulfillment Line item captures the quantity fulfilled against a given order line item. This is the information provided by the fulfillment systems and is used to track the total quantity fulfilled against the order line item. A given order line item can have multiple order fulfillment line items across one or more order fulfillment. |

The following fields are used in the distributed order fulfillment process to track and fulfill order line items from the beginning of order fulfillment to activation.

| Data | Details |
|--|--|
| Order Line Item > Ready for Fulfillment Date | Set this field on the parent order to apply it to all order line items associated with that order. (Note: Setting this date does not have any effect on Order Fulfillment Line Items or fulfillment status. |
| Order Line Item > Fulfilled Quantity | This field is automatically updated when the Order Fulfillment Line Item > Fulfilled Quantity for the given Order Line Item is saved. |
| Order Fulfillment > Account | The Account associated with one or more Order Fulfillment Line Items over one or more orders. |
| Order Fulfillment > Fulfillment Date | The date that corresponds to a single Order Fulfillment. |
| Order Fulfillment Line Item > Fulfilled Quantity | Set this field and save it to automatically updated the Fulfilled Quantity field value on the corresponding Order Line Item. |
| Order Fulfillment Line Item > Fulfillment In Progress? | Enable this flag on an Order Fulfillment Line Item to track stages of fulfillment for the corresponding Order Line Item. The Order Line Item Fulfilled Quantity and Status will not be changed regardless of fulfillment quantity as long as this flag is enabled. |

| Data | Details |
|--------------------------------------|---|
| Order Fulfillment Line Item > Status | Set this field to indicate the fulfillment status of an Order Fulfillment Line item. The Status is automatically updated to "Fulfilled" when the Fulfilled Quantity is greater than or equal to the Delta Quantity on the corresponding Order Line Item and "Fulfillment in Progress?" is disabled. |

Releasing Orders to Fulfillment Systems

After an Order is activated, the order must be fulfilled by delivering the purchased products or services to the customer. This process is known as Order Fulfillment. Conga Order Management supports distributed order fulfillment. This means that a single order can be fulfilled from different locations, warehouses, factories, or at different times in the fulfillment process. This is represented by the **Order Fulfillment** object that is associated with a specific **Account**.

Similarly, a single Order Fulfillment can have multiple **Order Fulfillment Line Items** that are all part of the same order or from multiple orders. Fulfillment is tracked for one or more orders in this way through the Fulfillment Line Items.

You can use the field **Service Location** on the Order Line Item to track from which warehouse or fulfillment location the order will get fulfilled. Provide the value for Service Location manually on the Order Line Item or using workflow rules.

Working with Order Fulfillment

Order Fulfillments are created based on input from fulfillment systems, such as SAP ERP, Oracle ERP, and so on. Order Fulfillment and Order Fulfillment Line Items can be created manually, but in most cases, these objects will be created through integrations with fulfillment and provisioning systems.

One Order Fulfillment can correspond to multiple orders for a given account. In this case, the Order Fulfillment captures all order line items to be fulfilled for a given customer on a particular date from the group.

Order Fulfillment
F-00000005

< Back to List: Custom Object Definitions

Order Fulfillment Line Items (5)

Order Fulfillment Detail [Edit] [Delete] [Clone]

Fulfillment Number F-00000005 Owner [User Name] [Change]

Fulfilled Date 7/9/2019 Account [Account Name]

Created By [User Name] 7/9/2019 2:50 AM Last Modified By [User Name] 7/9/2019 2:50 AM

[Edit] [Delete] [Clone]

Order Fulfillment Line Items [Order Fulfillment Line Items Help]

| Action | LineItem Id | Line Number | Order | Order Line Item | Fulfilled Quantity |
|----------------|---------------|-------------|------------|-----------------|--------------------|
| [Edit] [Del] | FI-0000000008 | 1 | | OI-0000008828 | 1.00000 |
| [Edit] [Del] | FI-0000000009 | 1 | Q-00003029 | OI-0000008882 | 1.00000 |
| [Edit] [Del] | FI-0000000010 | 1 | Q-00003029 | OI-0000008882 | 1.00000 |
| [Edit] [Del] | FI-0000000011 | 1 | Q-00003510 | OI-0000009716 | 2.00000 |
| [Edit] [Del] | FI-0000000013 | 1 | Q-00003510 | OI-0000009716 | 8.00000 |

Show 5 more » | Go to list (17) »

It is important to understand that one Order Fulfillment is not related to an order in any way. The fields to consider on the Order Fulfillment object are the **Fulfilled Date** and the **Account** associated with the fulfillment. Fulfillment of any actual orders is completed and tracked through the Order Fulfillment Line Items that comprise the particular Order Fulfillment.

i All order line items from a given order do not need to be fulfilled for given order fulfillment. Order can be fulfilled through multiple order fulfillment. Refer to [Tracking Order Fulfillment](#).

Working with Order Fulfillment Line Items

Order Fulfillment Line Items are created based on the input from fulfillment systems as the quantities for Order Line Items are fulfilled. A given Order Fulfillment Line Item corresponds to a given Order Line Item. It represents the quantity fulfilled for a given Order Line Item on a given date.

One Order Line Item can have multiple Order Fulfillment Line items across multiple Order Fulfillments. The **Fulfilled Quantity** field on an Order Fulfillment Line Item is aggregated for a given line item and is updated on the **Fulfilled Quantity** field for the corresponding Order Line Item. When the Fulfilled Quantity of an Order Fulfillment Line Item is saved, the Fulfilled Quantity on the linked Order Line Item is updated and the status of the Order Line Item is updated to one of two statuses:

- If the Fulfilled Quantity is greater or equal to the Delta Quantity, the Order Line Item status is updated to "Activated."
- If the Fulfilled Quantity is greater than zero, and less than the Delta Quantity, the Order Line Item status is updated to "Partially Fulfilled."

Order Fulfillment Line Item
FI-0000000008

[← Back to List: Custom Object Definitions](#)

[Edit Layout](#) | [Print](#)

Order Fulfillment Line Item Detail

| | |
|---|---|
| LineItem Id FI-0000000008 | Order Fulfillment F-00000005 |
| Line Number 1 | Order |
| Fulfillment In Progress? <input type="checkbox"/> | Status Fulfilled |
| Fulfilled Quantity 1.00000 | Order Line Item OI-0000008828 |
| Created By [User Name], 7/19/2019 5:15 AM | Last Modified By [User Name], 7/22/2019 3:30 AM |

Tracking Order Fulfillment

The Order fulfillment process calls for differing lifecycles for each of its order fulfillment line items depending on how the order is being fulfilled. As a result, Conga Order Management provides a partial order fulfillment to control this process at different levels, such as:

- The entire order is activated and fulfilled as a whole.
- Some of the line items in the order are activated while other line items are still undergoing fulfillment.
- Individual line items are partially fulfilled at different times (for example, part of the quantity for a given order line item is fulfilled and tracked as partially fulfilled).

✓ Scenarios for Partial Order Fulfillment:

- Company X receives an Order for 40 'platinum configuration' servers. The warehouse has only 30 units available. In this case, the Order is partially fulfilled with 30 units ready for delivery.
- A company producing Automobile spare parts has two production units for its products. It receives an Order for 10 different products. Since each product is manufactured at a different time in the production unit, the Order is fulfilled as and when the products complete the production process from start to finish.

As fulfillment line items are created and fulfilled, order line items associated with one or more orders are fulfilled, eventually resulting in the activation of order line items and orders as part of the process.

Extending Order Fulfillment Tracking

For most cases, the fulfillment of line item quantities followed by activation of the order line item is not sufficient for integration with fulfillment systems. When fulfillment scenarios require a series of intermediate steps in the fulfillment process you can enable the **Fulfillment in Progress?** field on the Order Fulfillment Line Item object. When this flag is enabled, any update to the Fulfilled Quantity on that fulfillment line item will not roll up to the corresponding order line item.

The **Status** field on the Fulfillment Order Line Item object designates the current stage of the fulfillment process. When all intermediate steps of the fulfillment process are complete, disable the **Fulfillment in Progress?** flag. This sets the status of the Fulfillment Line Item to "Fulfilled," which then triggers an update in the status of the corresponding Order Line Item.

i The Status field is not present on the Order Fulfillment Line Item page by default. An administrator must add it to the page layout to make any manual changes to status.

Using Service Locations to Track Partial Order Fulfillment

You can use the field **Service Location** on the Order Line Item object to track partial order fulfillment by shipping or service location. When you extend order fulfillment in this way, set the Service Location as part of the fulfillment process using workflow rules. For example, you may want to set a rule that help you to categorize order related to the USA under dollar (\$) currency price lists and not under INR currency price lists.

i The Service Location field is not present on the Order Line Item page by default. An administrator must add it to the page layout to make any manual changes to Service Location.

Activating an Order

Activating an order and its associated order line items indicates completion of the order fulfillment process and initiation of additional related downstream processes. Downstream processes include asset activation, billing initiation, revenue recognition, and so on.

Orders and Order Line Items are activated in the following ways:

- Auto Activate the order on generation from a quote/proposal: The **Auto Activate Order** flag is set to "true" on the quote/proposal. This generates an order with the status "Activated."
- Auto Activate the order on generation from an agreement: The **Auto Activate Order** flag is set to "true" on the agreement. This generates an order with the status "Activated."
- Set the Ready for Activation Date on the order: The **Ready for Activation Date** is specified on the order. This activates the order and its line items and initiates any downstream processes (if enabled), such as billing.
- Set the Ready for Activation Date on the order line item: The **Ready for Activation Date** is specified on an order line item. This activates the order line item but does not activate the corresponding order unless all of its line items are activated.
- Auto Activate the order on confirmation (Accept action) of a direct order: The **Auto Activate Order?** field is set to "true" on the order header.
- Activate order and order line items based on the line item fulfilled quantity:
 - Once the **Fulfilled Quantity** for a given line item is greater than or equal to the **Delta Quantity**, the Order Line Item status is set to "Fulfilled."
 - Once all the Order Line Items are "Activated", the corresponding order is also marked as "Activated."

Providing a Legal Entity for Downstream Processes

The **Legal Entity** field on the Order object can be specified to enable billing and revenue recognition for related legal entities. When an order line item is activated and an asset is created, the Legal entity value specified in the order header is copied to the **Legal Entity** field for the corresponding Asset Line Item.

ⓘ When a quote or agreement has a value for Legal Entity, that value is automatically copied when the quote or agreement is converted to an order. When you place a direct order or the field is not present in the quote or agreement record, you must specify a Legal Entity to enable any downstream processes.

Billing for an Order

Conga billing and invoicing can be initiated for the orders either before or after activation of the order. When billing is initiated depends on Order Management settings.

To create billing schedules on activation of an order

1. Go to **Setup > Custom Settings > Order System Properties**
2. Click **Manage**.
3. Click **Edit**.
4. Click the check box next to **Initiate Billing On Order Activation?** and click **Save**.

To create billing schedules manually

Prerequisite: The custom setting **Initiate Billing On Order Activation?** must be disabled.

1. Go to the order record and set the **Ready for Billing Date** field.
2. Click **Save**.

ⓘ If the **Ready for Billing Date** is set to a date before the order is activated, the billing schedules are created when the order becomes active.

Use Case: Billing for Standalone Recurring product

Tier1 system sells 'SecureDevice' (\$100/unit) which is billed monthly for a contract period of one year. The contract is in effect from 04/20/2016. The customer wants an invoice on 15th of every month.

Steps:

1. Add a product 'SecureDevice' with **List Price** as \$100 and **Price Type** as *Recurring*.
2. Under the Tax and Billing tab on PLI, set the **Billing Rule** as *Bill in Advance* and **Billing Frequency** as *Monthly*.
3. Create a new Billing Preference and set **Billing Cycle Start** to *Billing Day of the month*.
4. Set *Billing Day of the month* to 15th of the month.
5. Create a Quote/Proposal. Set **Start Date** as 04/20/2016 and **End Date** as 04/19/2017.
6. Add the product to your cart and **Finalize**.
7. **Present** the proposal and **Accept** it after the reviews.
8. Order and Asset are created when the proposal is accepted. Activate the order.

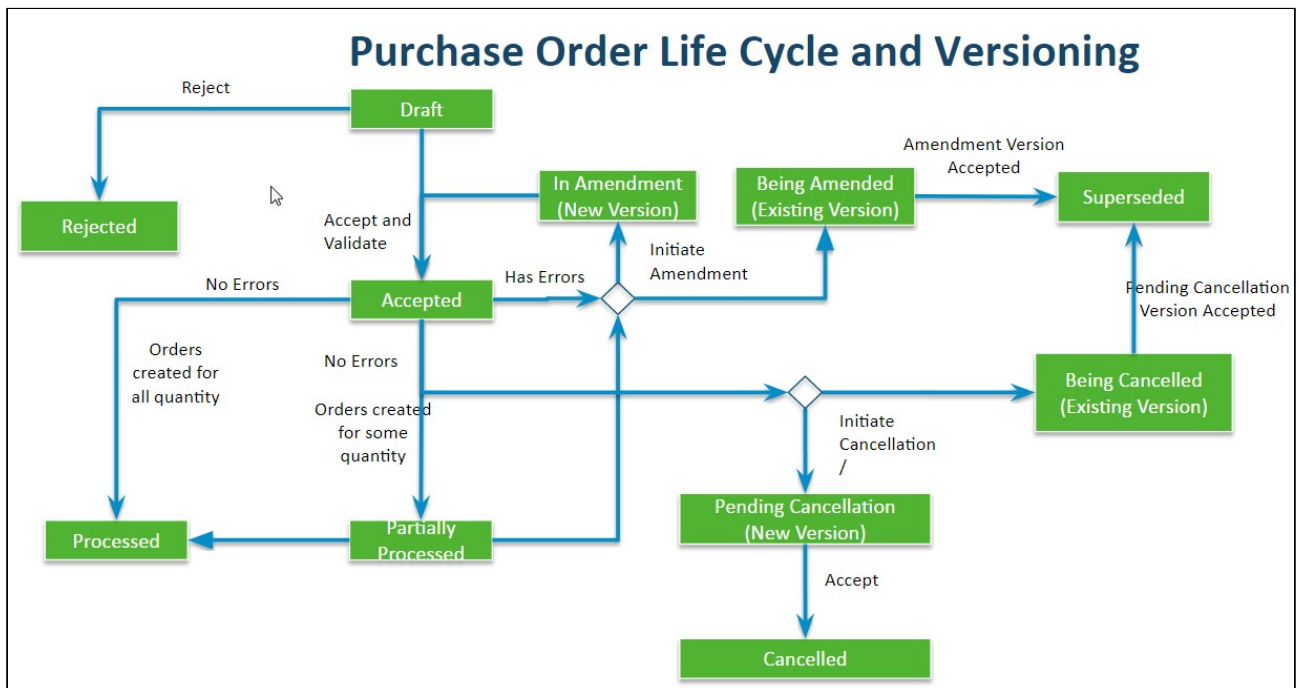
Resulting Billing Schedules

| Period Start Date | Period End Date | Ready for Invoice Date | Amount |
|-------------------|-----------------|------------------------|---------|
| 4/20/2016 | 5/14/2016 | 4/20/2016 | \$83.33 |
| 5/15/2016 | 6/14/2016 | 5/15/2016 | \$100 |
| 6/15/2016 | 7/14/2016 | 6/15/2016 | \$100 |

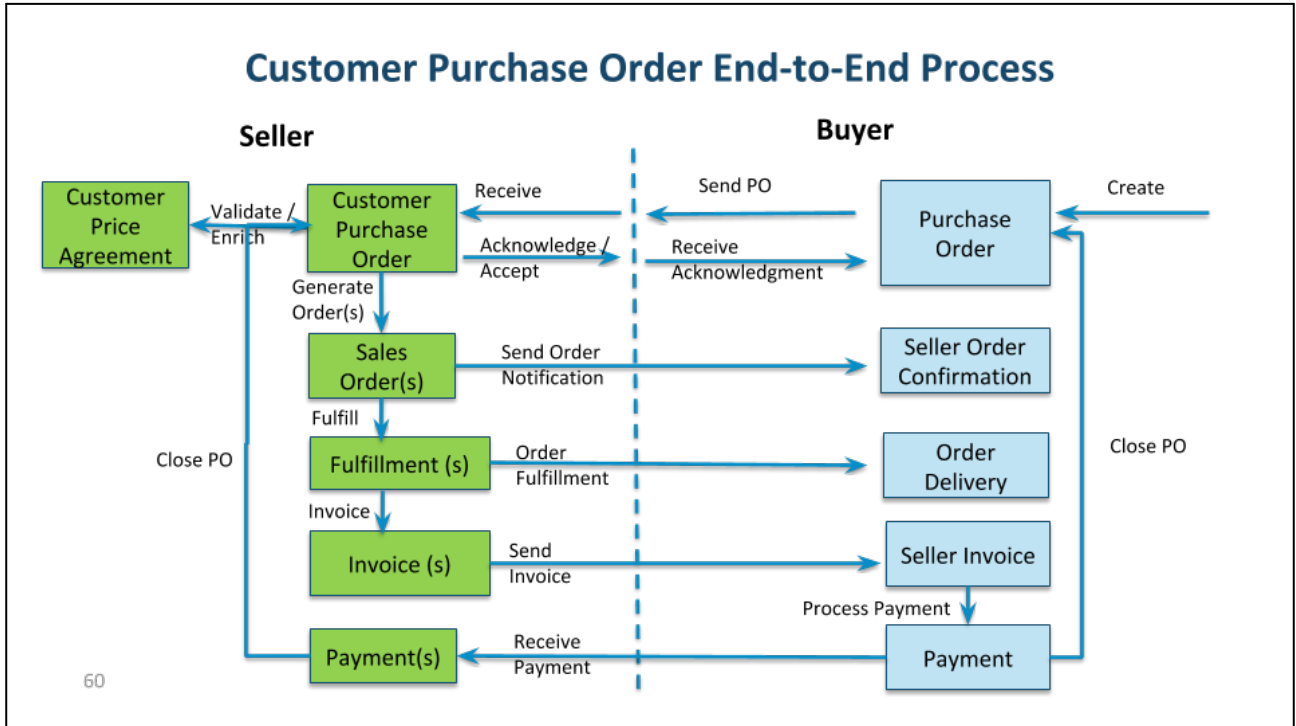
Working with Customer Purchase Orders

In Order Management, a **Customer Purchase Order** is a customer document (usually PDF or email) that represents the initial offer of negotiated types, quantities, and prices for products or services. Customer Purchase Orders can be created by sales users or administrators as sales orders for long-term contracts.

The following diagram illustrates the customer purchase order lifecycle.



When a customer places a purchase order, they submit a document that references the quote or agreement line item numbers from an associated customer price agreement. When Conga Order Management receives that purchase order the purchase order is validated against the line items and negotiated pricing in the price agreement, and any data enrichment rules are applied. The user must then mitigate any of the existing validation errors or warnings and accept the purchase order. Sales orders can then be placed against the purchase order and follow the same lifecycle from that point as with any other order processed by Conga Order Management.



Refer to the following table for the tasks described in the portion of the user guide.

| Function | Description |
|--|--|
| Capture Customer Purchase Orders | <ul style="list-style-type: none"> • Capture raw purchase orders from customers • Add PDF or Email as attachments to Customer POs |
| Manage Validation / Enrichment Rules | <ul style="list-style-type: none"> • Create and manage rules to validate, match, and enrich PO • Add PO items against customer’s existing contracts, Quotes, and other master data |
| Validate PO against Quotes/Contracts | <ul style="list-style-type: none"> • Validate or enrich PO and PO items against customer’s existing contracts, Quotes, or other master data |
| PO Exception or Issue Logs | <ul style="list-style-type: none"> • Capture errors or issues identified during data validation and enrichment of customer PO and PO Items |
| View PO header and line item validation issues | <ul style="list-style-type: none"> • View data validation issues for header and line items |

| Function | Description |
|---|---|
| Fix data validation and match issues | <ul style="list-style-type: none"> • Allow users to fix data validation or matching issues • Maintain version history of changes for audit trail |
| Create or Generate Sales Orders | <ul style="list-style-type: none"> • Convert the PO to a single sales order • Auto convert PO to multiple sales orders based on split criteria • Create partial orders from a PO |
| Amend or Cancel Inflight Customer PO | <ul style="list-style-type: none"> • Amend in-flight PO to add a line, change lines, cancel lines • Cancel in-flight PO |
| Track orders and fulfillment against PO | <ul style="list-style-type: none"> • Track ordered quantity by PO items • Track fulfilled quantity by PO items |

Understanding Customer Purchase Order Status

As a Customer Purchase Order moves through the order lifecycle, the purchase order and its line items are assigned a status. The following table describes the mapping between Customer Purchase Order (PO) and PO Item statuses:

| PO Status | PO Item Status |
|-----------|--|
| Draft | <ul style="list-style-type: none"> • All items are in "Draft" status. |
| Accepted | <ul style="list-style-type: none"> • All items are in "Accepted" status and the Customer PO is not in "In Amendment" status. • One or more items are in "Accepted" status while remaining items are in "Cancelled" status and Customer PO is not in "In Amendment" status. • PO Items and header may have errors. |

| PO Status | PO Item Status |
|----------------------|---|
| Partially Processed | <ul style="list-style-type: none"> Any one item is in "Partially Processed" status and Customer PO is not in "In Amendment" status. One or more items are in "Processed" status while remaining items are in "Accepted" status and Customer PO is not in "In Amendment" status. |
| Processed | <ul style="list-style-type: none"> All PO items are in "Processed" status. One or more items are in "Processed" status while remaining items are in "Cancelled" status. |
| Rejected | <ul style="list-style-type: none"> All PO Items are in "Rejected" status. PO Items and header may have errors. |
| In Amendment | <ul style="list-style-type: none"> PO Is explicitly amended and the current version is the latest one. PO items can be in "Draft," "In Amendment," "Pending Cancellation," "Cancelled," "Partially Processed," or "Processed" statuses. PO Items and header may have errors. |
| Pending Cancellation | <ul style="list-style-type: none"> All PO Items are in "Pending Cancellation" status. |
| Cancelled | <ul style="list-style-type: none"> All PO Items are in "Cancelled" status. |
| Being Cancelled | <ul style="list-style-type: none"> Next version exists in "Pending Cancellation" status. |
| Being Amended | <ul style="list-style-type: none"> Next Version exists in "In Amendment" status. |
| Superseded | <ul style="list-style-type: none"> Next version in "Accepted" status. |

Managing CPO Validation and Enrichment Rules

When you create a Customer Purchase Order in Order Management, you need to ensure that the Customer Purchase Order (CPO) you create contains Purchase Order Items (PO Items) that match line items coming from one or more referenced quotes, contracts, or price agreements. To accomplish this, you must configure Validation rules that validate data entered in the CPO against existing data in the price agreement, as well as Data Enrichment rules that automatically enter data for specific fields onto the CPO rather than

having to do it manually. When these rules trigger, they display errors or warnings that can be correct by the user creating the CPO before any orders are created.

You can create CPO Validation and Enrichment rules using the Purchase Order Admin (PO Admin) user interface. The PO Admin UI provides a step-by-step wizard that allows you to create a **Data Enrichment and Validation Ruleset** which contains one or more **Data Validation** and/or **Data Enrichment** rules that are triggered based on specified criteria when you are creating your CPO. You can then review and activate these rules prior to accepting the CPO.

Refer to [Setting Up Purchase Order Admin](#) in the *Order Management Administrator Guide* for step-by-step instructions for setting up the PO Admin angular UI.

Use the PO Admin UI to create rulesets containing the following rule types:

- **Data Enrichment Rules:** Use these rules to enrich the data in your CPO. For example, the PO item account must be the same as the Proposal Line Item or Source Object Account. You can define the data enrichment rule in order to input this data whenever you create a customer purchase order instead of feeding this data manually.
- **Data Validation Rules:** Use these rules to set up data validations for purchase orders or their items so that whenever you specify the values for quantity or fulfilled quantity, a data validation rule is triggered to ensure that the values fall within the specified limits. You can have two types of validations:
 - Warning: This type of data validation rule allows you to continue the operation despite the system warning. However, it warns you to take precautions while creating sales orders.
 - Error: This type of data validation rule does not allow to bypass the error and stops the operation until you mitigate the error before continue creating the sales orders.

Perform the following tasks in this section to create and manage CPO Validation and Enrichment rules.

- [Creating Data Enrichment and Validation Rules](#)
- [Mitigating Data Validation Issues](#)

Creating Data Enrichment and Validation Rules

You can use the Purchase Order Admin (PO Admin) user interface to create and manage Data Enrichment and Validation rules for your Customer Purchase Orders.

A **Data Enrichment and Validation Ruleset** defines the target object (Customer PO or Customer PO Item), the source object (Quote/Proposal, Contract, Order, Proposal Line Item, Agreement Line Item, or Order Line Item), the criteria to be applied to the target and source


objects, and any additional filter logic. When criteria for a ruleset is fulfilled, the **Enrichment and Validation Rules** defined in the ruleset are triggered and any actions specified in its rules are executed. Perform the following task to create a Data Enrichment and Validation Ruleset and its rules.

 Do not click your browsers "Back" button when using the PO Admin.




- [To create Enrichment and Validation rules using the PO Admin](#)
- [Use Case: Creating a Data Enrichment/Validation Ruleset for a Customer Purchase Order](#)


To create Enrichment and Validation rules using the PO Admin

1. Navigate to the **PO Admin** tab. The Customer Purchase Order Admin page is displayed.

 If you cannot find the PO Admin tab, your administrator must add the tab in Salesforce. Refer to the *Order Management Administrator Guide* for steps.

2. From the upper right-hand corner in the header, click **Create New Rule**. The Data Enrichment and Validation Setup wizard is displayed.
3. Enter a **Ruleset Name** and the **Sequence**, which indicates the order in which this ruleset will be validated on confirmation of the customer purchase order (starts at "1").
4. From the **Target Object To Validate** drop-down, select either **Customer Purchase Order** or **Customer PO Item**.
5. If you selected Customer PO Item as the object to validate, click the **Target Object Parent** drop-down and select **Customer Purchase Order**.
6. From the **Source Data Object** drop-down, select the source object on the price agreement you want to match to the customer purchase order.
7. Click **Next** to display the Specify Target Data Filter Criteria screen.
8. Click **Add** and define filter criteria to be applied to the target Customer PO or PO Item:
 - a. Select a **Field** on the Target object (for example, Status).
 - b. Select an **Operator**.
 - c. Enter or select a **Value** for the field (for example, Draft. In this case, whenever the system finds a status in Draft, the rule is triggered for the Customer Purchase Order).

9. Click **Add** to add another filter to the target and repeat step 8. You can remove defined filter criteria by clicking the Delete icon ().
10. Add any Filter Logic as needed.
11. Click **Next** to display the Specify Source Data Filter Criteria screen.
12. Click **Add** and define the filter criteria to match a field on the source object (Quote/ Proposal, Proposal Line Item, Agreement, Agreement Line Item, Order, or Order Line Item) with a field on the Customer PO or PO Item, or to validate a field on the source object against a field on the Customer PO or Customer PO Item:
 - a. Select a **Field** on the Source object (for example, Proposal Line Item).
 - b. Select an **Operator**.
 - c. To match a field on the source object with a field on the Customer PO, click the **Value Type** drop-down and select **Field Value**. Then select one or more values from the **Target Field** drop-down.
 - d. To validate a field on the source object against a field on the Customer PO, click the **Value Type** drop-down and select **Constant**. Enter the value for the field on the target object.
13. Click **Add** to add another filter to the target and repeat step 12. You can remove defined filter criteria by clicking the Delete icon ().
14. Click **Save and Close** to save the ruleset or click **Next** to display the Specify Enrichment Rules screen.
15. Define an Enrichment rule to set a value on the Customer PO or PO Item to a value from the selected Source Object field:
 - a. Select the **Target Object Field** (for example, Contract Item Number).
 - b. Select the **Source Object Field** (for example, Record ID).
 - c. Click the toggle to make the field a required field on the Customer Purchase Order or PO Item.
 - d. Enter the **Message** to be displayed to the user when an error occurs (for example, when either the target or source object field is not found).
16. Click **Add** to add another rule to the ruleset and repeat Step 15. You can remove a rule by clicking the Delete icon ().
17. Click **Save and Close** to save the ruleset or click **Next** to display the Specify Validation Rules screen.
18. Define a Validation rule to validate a value in the CPO or PO Item against the master data in the Source object field:
 - a. Select the **Target Object Field** (for example, Quantity).
 - b. Select the **Match Operator** (for example, "less than or equal to").
 - c. Select the **Source Object Field** (for example, Quantity (2)).
 - d. Select the **Error Type** to be triggered when the rule does not evaluate to true: Warning or Error.
 - e. Enter the **Message** to display to the user when an error occurs.

19. Click **Add** to add another rule to the ruleset and repeat step 18. You can remove a rule by clicking the Delete icon ().
20. Click **Next** to display the Review screen. You can review the entire Data and Enrichment Validation Rule Setup on this page. If you need to make any changes, click **Previous** to return to the appropriate step.
21. Click **Done** to save the ruleset and return to the PO Admin page.
22. Click **Activate** to activate the ruleset.

 When a rule is executed and any errors are found, they are recorded in the log.

Use Case: Creating a Data Enrichment/Validation Ruleset for a Customer Purchase Order

Description: The requirement for this Customer Purchase Order is to create and apply Data Enrichment and Validation rules as part of a ruleset that enriches the PO using data on the quote line item. The ruleset then validates that the quantity entered in the PO is the same as the quantity on the quote line item. This use case is an example of a ruleset that can be applied to multiple purchase orders with varying combinations of products and services.

In the following example, an administrator or sales user/customer support representative with appropriate permissions creates a Data Enrichment/Validation Ruleset and the Enrichment and Validation rules that comprise it.

Prerequisites:

- The Customer Purchase Order and Purchase Order Admin user interfaces must be configured.
- Data Enrichment/Validation Ruleset seed data must be configured.

 Refer to *Order Management on SFDC Administrator Guide* for configuration steps.

To create a Data Enrichment/Validation Ruleset to enrich the Customer Purchase Order and validate order quantities against quote line items

1. From the upper right-hand corner in the header, click **Create New Rule**. The Data Enrichment and Validation Setup wizard is displayed.
2. Enter "Enrich PO and validate quantity" as the Ruleset Name.
3. Enter "1" in the **Sequence** field.
4. Click the **Target Object To Validate** drop-down and select **Customer PO Item**.
5. Click the **Target Object Parent** drop-down and select **Customer Purchase Order**.
6. Click the **Source Data Object** drop-down and select **Proposal Line Item**.
7. Enter the **Description** "Enrich PO and validate quantity against QLI."
8. Click **Next** to display the Specify Target Data Filter Criteria screen.
9. Click **Add** and define the following filter criteria:
 - a. Click the **Field** drop-down and select the field "Status."
 - b. Click the **Operator** drop-down and select "in." Value type is "Constant."
 - c. Click the **Value** drop-down and select "In Amendment" and "Draft."
10. Click **Next** to display the Specify Source Data Filter Criteria screen.
11. Click **Add** and define the following filter criteria:
 - a. Click the **Field** drop-down and select the field "Line Item Id."
 - b. Click the **Operator** drop-down and select "equal to." Value type is "Field Value."
 - c. Click the **Value** drop-down and select "Quote Line Item."

12. Click **Next** to display the Specify Enrichment Rules screen.
13. Click **Add** and define an Enrichment rule to set the value for the Product field on the Customer PO Item to the value of the same field on the Proposal Line Item:
 - a. Click the **Target Object Field** drop-down and select **Product**.
 - b. Click the **Source Object Field** and select **Product**.
14. Click **Add** and define an Enrichment rule to set the value for the Ship To field on the Customer PO Item to the value of the same field on the Proposal Line Item:
 - a. Click the **Target Object Field** drop-down and select **Ship To**.
 - b. Click the **Source Object Field** and select **Ship To**.
 - c. Click the toggle to make **Ship To** a required field on the Customer Purchase Order or PO Item.
 - d. Enter the **Message** "Ship To not found at the source on the line item" (this displays a message to the user when the Ship To field has no value on the corresponding quote line item).
15. Click **Add** and define an Enrichment rule to set the value for the **Net Price** field on the Customer PO Item to the value of the same field on the Proposal Line Item:
 - a. Click the **Target Object Field** drop-down and select **Net Price**.
 - b. Click the **Source Object Field** and select **Net Price**.
16. Click **Next** to display the Specify Validation Rules screen.
17. Define a Validation rule to validate **Quantity** of the PO Item against master data in the Proposal Line Item record:
 - a. Click the **Target Object Field** drop-down and select **Quantity**.
 - b. Click the **Match Operator** drop-down and select **equal to**.
 - c. Click the **Source Object Field** drop-down and select **Quantity (2)**.
 - d. Click the **Error Type** drop-down and select **Error**.
 - e. In the **Message** field, enter "PO quantity does not match the value on the proposal line item."
18. Click **Next** to display the Review screen.
19. Click **Done** to save the ruleset and return to the PO Admin page.
20. Click **Activate** to activate the ruleset.

Result: A Data Enrichment/Validation Ruleset is created and activated. Any custom purchase orders created in the system from this point forward will use the enrichment and validation rules defined by the ruleset.

Next Step: Create a Customer Purchase Order and Purchase Order items for an accepted quote.

Mitigating Data Validation Issues

Conga Order Management displays the data validation messages based on the data validation rules that you set while configuring a ruleset.

For example,

If you have set proposal item quantity cannot be greater than PO Item quantity, then after the data validation rule is executed, Conga Order Management maps the target and source object fields for quantity.

Depending on the error type that you configured in the data validation rule, a warning or an error is shown that you need to act upon in order to complete the operation.

Customer PO Issue Logs

For the In-Amendment orders, when you click **Validate Customer PO** Conga Order Management generates all the PO related issue details in the **Customer PO Issue Logs**.

Error or Warning fields


Based on the below fields you can identify errors with your Customer PO and its items.

- Validation Errors
- Number Of Errors
- Number Of Warnings

To fix the errors, you can go to each ruleset using the issue log and amend the CPO that are in Draft and In-Amendment states.

Creating Customer Purchase Orders


Before you can create a sales order from a purchase order, you must capture the details of the customer purchase order by creating it manually using the Order Management UI.

 Customer Purchase Orders can be created automatically through integration, as long as the Purchase Order data comes to Conga in a machine-readable format (for example, XML or JSON). For assistance with this procedure, contact Conga Professional Services.

To create a customer purchase order

1. Navigate to the **Customer Purchase Orders** tab. The Customer Purchase Order page is displayed.

2. Click **New Purchase Order**. The New Purchase Order page is displayed.
3. Enter the following fields in the Order Basic section of the page: **Previous Version, Account, Account Name, PO Date, Price List, and Price List Name**. Enter the **Quote Number/Proposal number** associated with the Price Agreement. Enter any additional fields as required.
4. Enter the following fields in the Order Customer section of the page: **Bill To, Ship To, and Contact**. Enter their corresponding reference fields to store the IDs. The **Account Reference** field is the record Id received from the external system.
5. Add the **Requested Delivery Date**.
6. Click **Save & Add Products**. The **PO Number** is autogenerated. The **Status** of the Customer Purchase Order is set to *Draft*.
7. Once the purchase order is created, click **Add/Edit Line Items** to modify existing purchase order line items or add new line items to the customer purchase order. Click **Save** to save line items.


 You can **Add/Edit Line Items** only when the purchase order is in "Draft" or "In Amendment" status.


Accepting Customer Purchase Orders User Guide

After you create a Customer Purchase Order, the status of PO and its line items changes to "Draft."

To accept a Customer PO

1. Navigate to the **Customer Purchase Orders** tab. The Customer Purchase Order page is displayed.
2. Use the type ahead to filter the list of purchase orders. Click the **PO Number** for the purchase order you want to view. The Customer Purchase Order detail page is displayed.
3. Click **Accept**. The status of the CPO and its PO Items changes to **Accepted**.

 The Accept button is only enabled when the PO is in "Draft," "In Amendment," or "Pending Cancellation" status **and** at least one line item has been added.

4. Click the info () icon to view a list of warnings and errors. Click **Correct Problem** to take mitigating action for any errors.
5. After you have corrected all errors, you can [create a sales order](#).

Amending Customer Purchase Orders User Guide

Amend a Customer Purchase Order to create a new version after accepting a purchase order and correcting any errors. You can only amend a CPO when the status is in "Accepted" or "Partially Processed."

To amend a Customer PO

1. Navigate to the **Customer Purchase Orders** tab. The Customer Purchase Orders page is displayed.
2. Use the type-ahead to filter the list of purchase orders. Click the **PO Number** for the purchase order you want to view. The Customer Purchase Order detail page is displayed.
3. Click **Amend**. A confirmation dialog is displayed.
4. Click **Yes** to confirm the amend action:
 - A new version of the CPO is created and assigned the "In Amendment" status.
 - The **Accept Button** and **Undo Amend** buttons are enabled.
 - The status of the previous version is updated to "Being Amended."
5. Add, Edit, or Delete a PO Item as required. For details, refer to [Creating Customer Purchase Orders](#). You can add a new PO item or edit the quantity of an existing PO item as appropriate.
6. Click **Accept**. The status of the PO and its line items changes to "Accepted." The status of the previous version changes to "Superseded."

Reverting Changes to an Amended Customer Purchase Order

You can revert the changes that you made to the CPO by using **Undo Amend Order** action. You can only revert Customer POs in the "In Amendment" status.

To undo an amendment to a Customer PO

1. From the Customer Purchase details, click **Undo Amend Order**:
 - Any changes made to Customer PO and its line items are rolled back to the previous version.
 - The status of the previous Customer PO and its line items changes to "Accepted."

Canceling Customer Purchase Orders User Guide

You can cancel a Customer PO when it is in the "Accepted" status.

To cancel a Customer PO

1. Navigate to the **Customer Purchase Orders** tab. The Customer Purchase Order page is displayed.
2. Use the type ahead to filter the list of purchase orders. Click the **PO Number** for the purchase order you want to view. The Customer Purchase Order detail page is displayed.
3. Click **Cancel Order**
 - The **Status** of PO and its line items changes to **Pending Cancellation**.
 - The **Accept** button is enabled.
 - A new version of the PO is created.
 - The status of previous version changes to **Being Cancelled** and the status of PO items remains **Accepted**.
4. Click **Accept**. The status of the Customer PO and its line items changes to "Cancelled." The status of previous version changes to "Superseded."

Reverting Cancelled Customer Purchase Orders

You can revert cancelling a Customer PO and its items by using the **Undo Cancel Order** action.

To undo cancel a customer PO


1. From the Customer Purchase Order details page, click **Undo Cancel Order**.
 - The changes made to customer PO and its line items are rolled back to the old version.
 - The status of the previous Customer PO and its line items changes to "Accepted."

Creating Sales Orders from a Customer Purchase Order

After a Customer Purchase Order is accepted and validated, you can create one or more sales orders from the purchase order.



You can create a sales order from the purchase order in one of three ways as follows:

- **Create Single Order:** Create a single order for the whole customer PO.
- **Split Multi Orders:** Select one or more "split criteria" to create multiple orders organized into groups based on the criteria (for example, group PO Line Items by Contract Number).
- **Create Partial Order:** Create a partial order from one or more PO Items.

 For more information about creating orders from Customer Purchase Orders APIs, refer to the *Order Management SOAP API Guide*.

Creating a Single Order

After an order is accepted and any errors or warnings are resolved, you can create a single order from the entire Customer PO.



 Do not click **Cancel** to go back to the previous page as this will cancel the entire Customer Purchase Order. Click the Home icon () icon to return to the list of purchase orders.

To create a single order from the CPO

1. Navigate to the **Customer Purchase Order** tab.
2. Use the type-ahead to filter the list of purchase orders and click a **PO Number** to view the purchase order details.
3. From the header in the upper right-hand corner of the page, click **Create Sales Order** and select **Create Single Order**:
 - The Order is generated under the **Sales Orders** tab.
 - The status of the sales order generated from the customer purchase order is updated to "Pending."
 - The status of the purchase order is updated to "Processed."

Creating Multiple Sales Orders

After an order is accepted and any errors or warnings are resolved, you can split the Customer PO into multiple orders based on split criteria. For example, PO Line Number, Quantity, Net Price, End Date, and so on.



 Do not click **Cancel** to go back to the previous page as this will cancel the entire Customer Purchase Order. Click the Home icon () icon to return to the list of purchase orders.

To create a split order from the Customer PO

1. Navigate to the **Customer Purchase Order** tab.
2. Use the type-ahead to filter the list of purchase orders and click a **PO Number** to view the purchase order details.
3. From the header in the upper right-hand corner of the page, click **Create Sales Order** and select **Split Multi Orders**. The Create Multiple Sales Order screen is displayed.
4. Click the drop-down to select one or more **Split Criteria** to group purchase order items into multiple orders.
5. Scroll to the bottom of the page to view a message indicating how many orders will be created.
6. Click **Create Sales Order**. Multiple sales orders are created based on Split Criteria:
 - Each order is generated under the Sales Orders tab.
 - The status of each sales order generated from the customer purchase order is updated to "Pending."
 - The status of the purchase order is updated to "Processed."


Creating Partial Orders

After an order is accepted and any errors or warnings are resolved, you can create a Partial Order from the Customer Purchase Order. Create a partial order when you only need to order a partial quantity of the PO Items listed on the Customer PO.

 Do not click **Cancel** to go back to the previous page as this will cancel the entire Customer Purchase Order. Click the Home icon () icon to return to the list of purchase orders.

To create a partial order from the Customer PO

1. Navigate to the **Customer Purchase Order** tab.
2. Use the type ahead to filter the list of purchase orders and click a **PO Number** to view the purchase order details.
3. From the header in the upper right-hand corner of the page, click **Create Sales Order** and select **Create Partial Order**. The Create Partial Order screen is displayed.
4. Use the type ahead to filter the list of PO Items.
5. For each PO Item, enter a **Process Qty** for the partial order. Process Qty must be less than or equal to the difference between Remaining Qty and Processed Qty (also displayed in the list).
6. Click **Create Sales Order**. The Partial Order is created:
 - Each order is generated under the Sales Orders tab.
 - The status of each sales order generated from the customer purchase order is updated to "Pending."
 - The status of the purchase order is updated to "Partially Processed."

 You cannot create a partial order from a Customer PO if the status of the order is "Partially Processed."

Intelligent Order Assistant (Max for OM)

Intelligent Order Assistant is a conversational interface developed using Conga Max bot technology. It is available embedded within web applications as well as on mobile devices through channels Slack, Skype etc. This can help in various order management functions including view orders, view order details, create order, amend order, cancel order among others. There are standard conversational flows available which can be modified at the project level.

 Max functionality is not currently supported in this release.

Order Management (Out-Of-the-Box) Max Flow

When User types, Hello or Hi, Max respond with the following options:

- View Orders
- View Order Details
- Create New Order
- Amend Order

Order Management

- Cancel Order
- Clone Order

A user can choose an option to start the relevant conversation.

Order management for SOAP API

Developers

The *Conga Order Management SOAP API Guide* provides a collection of SOAP (Simple Object Access Protocol) service APIs to allow users to extend the feature functionality of Conga Order Management. Use the SOAP APIs in this reference guide as a tool for designing solutions to complex business needs. All APIs in the reference are defined using the SOAP, WSDL, and WS-I standards.

| Topic | Description |
|------------------|--|
| What's Covered | This guide defines and provides examples of Order Web Service APIs that can be used to create and manage orders, including partial orders from price agreements and customer purchase orders, as well as in-flight order changes and cancellation. |
| Primary Audience | <ul style="list-style-type: none"> • Order Management Administrators • Conga Professional Services |
| IT Environment | Refer to the latest <i>Order Management on Salesforce Release Notes</i> for information on System Requirements and Supported Platforms. |
| Updates | For a comprehensive list of updates to this guide for each release, see the What's New in Order Management Documentation topic. |
| Other Resources | <ul style="list-style-type: none"> • Salesforce SOAP API Developer Guide • Order Management User Guide: Refer to this guide for installing and setting up Order Management in your organization. • Order Management Release Notes: Refer to this guide for the new feature, enhancements, resolved, and known issues. • Order Management SOAP API Guide: Refer to this guide for documentation of public-facing SOAP APIs for Order Management. • Order Management Administrator Guide: Refer to this guide for setting up Order Management in your organization. • CPQ on Salesforce Administrator Guide: Refer to this guide for setting up products, price lists, and constraint rules. |

This guide describes the following tasks:

- Implementing Order Management Web Service
- Implementing Customer Purchase Order Web Service

Before using Order Management APIs, you must be familiar with the following:

- Basic Salesforce administration knowledge
- Conga CPQ and Conga CLM administration
- Salesforce and Conga terms and definitions
- Conga Order Management basics
- Experience with SOAP/WSDL API protocols

Select one of the following topics for more information:

- [Getting Started](#)
- [API Reference](#)
- [Scenarios](#)

Getting Started

Execution of Conga Order Management requires a working knowledge of API standards and development platform, as well as the subset of supported data and field types in Salesforce. Refer to the *Conga Order Management Release Notes* for the supported Order Management package for the latest release.

Refer to the following topics:

- [API Standards and Development Platforms](#)
- [Field Types](#)
- [Integrating Conga with External Systems](#)

API Standards and Development Platforms

Conga APIs are based on Salesforce APIs and use the same standards and platforms.

Standards

| Name | Reference |
|--|---|
| Simple Object Access Protocol (SOAP) 1.1 | http://www.w3.org/TR/2000/NOTE-SOAP-20000508 |

| Name | Reference |
|---|---|
| Web Service Description Language (WSDL) 1.1 | http://www.w3.org/TR/2001/NOTE-wsdl-20010315 |
| WS-I Basic Profile 1.1 | http://www.ws-i.org/Profiles/BasicProfile-1.1-2004-08-24.html |

Development Platforms

Conga SOAP APIs work with standard SOAP development environments. For a list of compatible development platforms, see [Salesforce Developer Force API](#) details.

Note

A later version of this content will contain sample code.

Field Types

Conga APIs use a subset of the supported data and field types on Salesforce.

The following table lists the APIs that Conga provides. For a comprehensive list of all field types supported by Salesforce, see [Salesforce Data Types](#).

| Type | Description |
|-------------|---|
| Boolean | The Boolean field has a true (or 1) or false (or 0) value. |
| Data object | The Data Object field is an ID type and is represented by <code>CPQ.nno0</code> in this document. |
| Date | The Date field contains date values only and does not contain relevant time values. Time in a date field is always set to midnight in the UTC time zone. If you want a timestamp you must use a DateTime field. |

| Type | Description |
|---------|--|
| Decimal | The Decimal field provides an exact numeric value and you can arbitrarily size the precision and scale of the value. |
| ID | <p>The ID field is an alphanumeric field that acts as the primary key for a specific record associated with an object. The ID value includes a three-character code that identifies which object the record is associated with. The ID for a specific record does not change.</p> <p>For some objects, this field may also be a reference type value, which contains the ID value for a related record. They are identified by field names ending in 'Id', such as <code>priceListId</code>. The ID field acts like foreign keys and their values can be changed using an <code>update()</code> call.</p> |
| Integer | The Integer field contains whole numbers only. There are no digits after the decimal. |
| List | The List field includes a fixed set of values from which you must select a single value. Picklists are available as drop-down lists. If a picklist is unrestricted, the API does not limit entries to only currently active values. |
| String | The String field contains text and may have different length restrictions based on the data you store in the specific field. For instance, <code>City</code> may be limited to 50 characters, while <code>AddressLine1</code> is limited to 255 characters. |

Integrating Conga with External Systems

Additional steps are required when you choose to integrate Order Management on Salesforce with external applications, customer portals, or other critical business systems. Because Order Management Web Services are hosted on Salesforce, you should familiarize yourself with the Salesforce SOAP API and processes surrounding integration and best practices detailed here: https://developer.salesforce.com/docs/atlas.en-us.api.meta/api/sforce_api_quickstart_intro.htm

Best Practices

Conga recommends that you work with Conga Professional Services to design and implement your integration. Use this documentation for basic integration steps and to reference Order Management Web Service calls.


The following basic steps are required to get started using the Conga Order Management Web Services API.

1. **Generate the Enterprise or Partner WSDL** – Integration with data stored in Salesforce requires you to first point your browser to the Salesforce Enterprise or Partner WSDL. This WSDL is generally provided by Conga Professional Services. Refer to [Salesforce Documentation](#) for complete instructions on generating the Web Service WSDL.
2. **Generate the Conga Web Services WSDL** – After you have connected to Conga Web Service, go to your organization and download the WSDL for the appropriate Web Service (Order Management Web Service, Batch Update, Proposal Web Service, and so on).
3. **Import the WSDL Files Into Your Development Platform** – After you have generated the WSDL files, you can import them into your development platform. Conga does not provide instructions for the import process. Refer to Salesforce documentation or documentation related to your development platform.
4. **Connect to Conga** – Before you can begin using Order Management Web Services, you must first authenticate to Conga using the login() API.
 - [Generating the Conga Web Services WSDL](#)
 - [To generate the Conga Web Services WSDL](#)
 - [Connecting to Conga](#)
 - [To connect to Conga Web Service using SoapUI](#)

Generating the Conga Web Services WSDL

Before you can import Conga SOAP Web Service into your development or testing platform, you must generate and download the Conga WSDL for the appropriate Web Service.

The example provided here uses SoapUI.


-  There a known bug in the WSDL Generator on Salesforce that does not include several field types, so it is recommended to update the WSDL file after you have generated it but *before* importing it into your development platform. You can find the details for any workaround tasks here:
- https://success.salesforce.com/issues_view?id=a1p3A000000eatxQAA&title=generated-wsdl-for-apex-webservices-is-malformed

- https://success.salesforce.com/issues_view?id=a1p300000008XKUAA2


When updating generated WSDL, make sure that the target namespace for any schema you add points to the correct Web Service (for example, schemas/class/Conga_QPConfig/QPConfigWebService). If you are still having trouble, please ask Conga Professional Services for a modified WSDL for the Web Services you are using.

To generate the Conga Web Services WSDL

1. Log in to the Salesforce organization that contains your Conga records and data (sandbox or production).
2. Go to **Setup > Develop > Apex Classes** (on Lightning, go to **Setup > Custom Code > Apex Classes**).
3. Find the Web Service you want to generate the WSDL for (for example, **OrderWebService**).
4. Click the **WSDL** link to generate the WSDL. The WSDL XML is generated and displayed in a new tab.

| | | |
|-------------------------------|---|----------------|
| Edit |  CPQSystemOverviewController | Apttus_Config2 |
| Edit |  CPQSystemOverviewControllerTest | Apttus_Config2 |
| Edit Security |  CPQWebService | Apttus_CPQApi |
| Edit WSDL Security |  CPQWebService | Apttus_Config2 |
| Edit |  CPQWebServiceSupport | Apttus_CPQApi |
| Edit |  CPQWebServiceTest | Apttus_CPQApi |
| Edit |  CPQWebServiceTest | Apttus_Config2 |

5. Right-click on the page and select **View Page Source**. Copy the XML content to any text editor.
6. Save the file with the extension **.wsdl**.
7. Open SoapUI (or wherever is required on your development platform).
8. Create a new SOAP project and import the Conga Web Services WSDL. All methods under that Web Service are now available to call.

 Refer to the **Request/Response XML** section for any API in this reference to get the structure of the request and any prerequisite calls required for any API.

Connecting to Conga


After you have downloaded the Enterprise or Partner WSDL, call the **login()** method to obtain a session ID from your org that you can use when calling Order Management Web Services. After authenticating, you can use the same session ID until it either expires or your logout or login again.

The example provided here uses SoapUI, an API testing tool which can be downloaded for free here: <https://www.soapui.org/>.

Prerequisite: To authenticate with Conga, please make sure to have your production or test org credentials on hand (username and password).

To connect to Conga Web Service using SoapUI

1. Open SoapUI. Go to **File > New SOAP Project**.
2. Enter a name for the project.
3. Click **Browse**. Navigate to the saved Enterprise or Partner WSDL file that you downloaded and click **Open**.
4. Click **OK** to close the project window.
5. From the Navigation panel to the left, highlight the project folder and click to expand. Click to expand the **SoapBinding**. The list of methods that comprise the Enterprise or Partner services are displayed.
6. Scroll down and right click on **login**. Double-click on an existing **Request**. The request window opens in the SoapUI interface.

 If you are doing this for the first time, you need to right-click on the **login** method and select **New Request**.

7. Select and delete all content following the `<soapenv:Header>` tag and the `</soapenv:Header>` tag.
8. Enter the username for your org (must have appropriate privileges) between the `<urn:username>` and `</urn:username>` tags.
9. Enter the password for your org (must have appropriate privileges) between the `<urn:password>` and `</urn:password>` tags.

The request should look like the following:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:urn="urn:partner.soap.sforce.com">
  <soapenv:Header>
  </soapenv:Header>
  <soapenv:Body>
    <urn:login>
      <urn:username>username@example.com</urn:username>
      <urn:password>password</urn:password>
    </urn:login>
  </soapenv:Body>
</soapenv:Envelope>
```

10. From the upper-left corner of the window, click the **Run** (▶) icon. The response is generated in the right-hand window.
11. Make note of the **serverURL** and the **sessionId** returned by the server. You will use the information to make Conga Web Service calls.

API Reference

The following set of Conga Order Management APIs are divided into two main web service categories:

- [Order Web Service](#): Use Order Web Service APIs to create and manage orders and execute order workflow automation rules.
- [Customer Purchase Order \(CPO\) Web Service](#): Use the CPO Web Service APIs to manage the customer purchase order lifecycle and create sales orders from purchase orders.
- [DataValidationService](#): Use Data validation web service API to execute the data validation and enrichment rules.

Order Web Service

Use the Order Web Service APIs to manage the order lifecycle, create direct orders, and create partial orders from Price Agreements.

You can invoke APIs in Order Web Service (Conga_Config2) from the following command.

```
Apttus_Config2.CPQStruct.<Name of the Function>
where the name of the function is API Name and it parameters.
```

Use the Order Web Service APIs to complete the following tasks:

- [Creating an Order](#)
- [Creating a Cart for an Order](#)
- [Synchronizing a Cart to an Order](#)
- [Accepting an Order](#)
- [In-Flight Order Changes and Cancellation](#)
- [Cloning an Order using SOAP API](#)
- [Fetching Price Agreements](#)
- [Fetching Price Agreement Items](#)
- [Creating a Partial Order from a Quote or Agreement](#)
- [Creating a Direct Order](#)
- [Executing Order Workflow](#)
- [Splitting an Order using API](#)

Creating an Order

This API creates an order for a specific Account and Price List.

API Details

| API | Signature |
|--------------------|--|
| createOrder | <i>webService static</i> <i>Apttus_Config2.CPQStruct.CreateOrderResponseDO</i> <i>createOrder(Apttus_Config2.CPQStruct.CreateOrderRequestDO)</i> |

| Parameters | | |
|------------|--------------------------------|--------------------------|
| Name | Type | Description |
| request | CPQStruct.CreateOrderRequestDO | The request data object. |

| Request Data Object - CPQStruct.CreateOrderRequestDO | | | |
|--|----------|------------|---|
| Field | Type | Required ? | Description |
| AccountId | ID | Yes | The Id of the Account to be associated with the order. |
| PriceListId | ID | Yes | The Id of the PriceList to be associated with the order. Only the products associated with the Price List appear on the cart. |
| PricingDate | Datetime | No | Date and time associated with the order and represent the date when the order will be priced. |

API Response

| Response Data Object - CPQStruct.CreateOrderResponseDO | | |
|--|------|--|
| Field | Type | Description |
| orderId | ID | The Id of the newly created order object |

Code Sample

The following code sample enables you to create an order for a valid account with an Account Id and a Price List with a valid Price List Id. Using the sample, you can search for a valid account using an account number. If an account exists with the account number entered, you can create an order using the createOrder() API. You can invoke this API in use cases when you want to create an order based on the account and price list.

```

1 public void createOrder()
2 {
3     // Create the request
4     Apttus_Config2.CPQStruct.CreateOrderRequestDO request = new
    Apttus_Config2.CPQStruct.CreateOrderRequestDO();

```

```

5
6     // Add request parameters
7     request.AccountId = accountId;
8     request.PricelistId = priceListId;
9     request.PricingDate = Datetime.now();
10
11    // Create a new order
12    Apttus_Config2.CPQStruct.CreateOrderResponseDO result =
Apttus_Config2.OrderWebService.createOrder(request);
13
14    // Assign orderId to local variable
15    orderId = result.OrderSO.Id;
16 }

```

Integration Details

Use the following information in your integrations with Conga CPQ Web Services API. Refer to [Integrating Conga with External Systems](#) for information on how to get started.

API Prerequisites

None.

Request and Response XML

Example Request

```

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:ord="
http://soap.sforce.com/schemas/class/Apttus_Config2/OrderWebService" xmlns:cpq="http:
//soap.sforce.com/schemas/class/Apttus_Config2/CPQStruct">
  <soapenv:Header>
    <ord:SessionHeader>
      <ord:sessionId>00D7A0000009QII!
ARUAQFmpeB0msYS2JcADxYc4PxWo0KAHJP1JiNo5vtvnDBCEvfnWWm07pqbEXY7WadiCmm_Vib6MjhmGUbR8
321tI01NQRg</ord:sessionId>
    </ord:SessionHeader>
  </soapenv:Header>
  <soapenv:Body>
    <ord:createOrder>
      <ord:request>
        <cpq:AccountId>0017A00000WW0pZQAT</cpq:AccountId>

```

```

        </ord:request>
    </ord:createOrder>
</soapenv:Body>
</soapenv:Envelope>

```

Example Response

```

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns="http://soap.sforce.com/schemas/class/Apttus_Config2/OrderWebService"
xmlns:CreateOrderResponseD0="http://soap.sforce.com/schemas/class/Apttus_Config2/CPQStruct" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <createOrderResponse>
      <result>
        <CreateOrderResponseD0:OrderSO xsi:type="Apttus_Config2__Order__c">
          <Id>a2n7A0000002tviQAA</Id>
          <Apttus_Config2__AutoActivateOrder__c>>false</
Apttus_Config2__AutoActivateOrder__c>
          <Apttus_Config2__BillToAccountId__c>0017A00000WW0pZQAT</
Apttus_Config2__BillToAccountId__c>
          <Apttus_Config2__BillingPreferenceId__c>a2d2f0000000h0YAAQ</
Apttus_Config2__BillingPreferenceId__c>
          <Apttus_Config2__OrderDate__c>2020-07-06T06:33:53.000Z</
Apttus_Config2__OrderDate__c>
          <Apttus_Config2__PricingDate__c>2020-07-06T06:33:53.000Z</
Apttus_Config2__PricingDate__c>
          <Apttus_Config2__ShipToAccountId__c>0017A00000WW0pZQAT</
Apttus_Config2__ShipToAccountId__c>
          <Apttus_Config2__SoldToAccountId__c>0017A00000WW0pZQAT</
Apttus_Config2__SoldToAccountId__c>
          <Apttus_Config2__SoldToAccountId__r xsi:type="Account">
            <Id>0017A00000WW0pZQAT</Id>
            <Name>OM_Account</Name>
          </Apttus_Config2__SoldToAccountId__r>
          <Apttus_Config2__Source__c>Account</Apttus_Config2__Source__c>
          <Apttus_Config2__Status__c>Draft</Apttus_Config2__Status__c>
          <Apttus_Config2__Type__c>New</Apttus_Config2__Type__c>
          <Name>0-00005958</Name>
          <OwnerId>0052f000000j6YUAA</OwnerId>
        </CreateOrderResponseD0:OrderSO>
      </result>

```

```

    </createOrderResponse>
  </soapenv:Body>
</soapenv:Envelope>

```

Creating a Cart for an Order

This API creates a cart for the order referenced by OrderId.

API Details

| API | Signature |
|------------|--|
| createCart | <pre> webService static Conga_Config2.CPQStruct.CreateCartResponseDO createCart(Apttus_Config2.CPQStruct.CreateCartRequestDO) </pre> |

| Parameters | | |
|------------|-------------------------------|--------------------------|
| Name | Type | Description |
| request | CPQStruct.CreateCartRequestDO | The request data object. |

| Request Data Object - CPQStruct.CreateCartRequestDO | | | |
|---|------------------------|-----------|---|
| Field | Type | Required? | Description |
| OrderId | ID | Yes | The Id of the order to be associated with the cart. |
| Properties | List<Config2.Property> | No | The list of properties applicable to the cart. |

| Data Object - Config2.Property | | | |
|--------------------------------|--------|------------|--|
| Field | Type | Required ? | Description |
| Name | String | Yes | Specify the features applicable to the cart. Applicable values are: <ul style="list-style-type: none"> • useAdvancedApproval: Enables Advanced Approval for a cart. • useDealOptimizer: Enables Deal Optimizer for a cart. |
| Value | String | No | The applicable values are true or false. Specifying the value as true enables the feature for a cart. |

API Response

| Response Data Object - CPQStruct.CreateCartResponseDO | | |
|---|------|--|
| Field | Type | Description |
| CartId | ID | The Id of the newly created cart object. |

Code Sample

The following code sample enables you to create a cart for a valid order with an Order Id. Using the sample, you can search for a valid order using an order number. If an order exists with the order number entered, you can create a cart using `createCart()`. You can invoke this API in use cases when you want to show a cart page based on the order.

```

1  public void createCart()
2  {
3      if(String.isNotBlank(orderId))
4      {
5          // Create config properties
6          List<Apttus_Config2.Property> configProps = new
List<Apttus_Config2.Property>();

```



```

7
8     Apttus_Config2.Property prop = new Apttus_Config2.Property();
9
10    prop.Name = 'useAdvancedApproval';
11    prop.Value = 'false';
12    configProps.add(prop);
13
14    prop = new Apttus_Config2.Property();
15    prop.Name = 'useDealOptimizer';
16    prop.Value = 'false';
17    configProps.add(prop);
18
19    // Create the request
20    Apttus_Config2.CPQStruct.CreateCartRequestDO request = new
Apttus_Config2.CPQStruct.CreateCartRequestDO();
21
22    // Add request parameters
23    request.OrderId = orderId;
24    request.Properties.addAll(configProps);
25
26    // Create a new cart for Order
27    Apttus_Config2.CPQStruct.CreateCartResponseDO result =
Apttus_Config2.OrderWebService.createCart(request);
28
29    // Get the cart Id
30    cartId = result.CartId;
31    } else {
32        ApexPages.addMessage(new
ApexPages.Message(ApexPages.severity.info, 'Invalid or Blank Order Id.))
;
33    }
34 }

```

Integration Details

Use the following information in your integrations with Conga CPQ Web Services API. Refer to [Integrating Conga with External Systems](#) for information on how to get started.

API Prerequisites

None.

Request and Response XML

Example Request

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:ord="
http://soap.sforce.com/schemas/class/Apttus_Config2/OrderWebService" xmlns:cpq="http:
//soap.sforce.com/schemas/class/Apttus_Config2/CPQStruct">
  <soapenv:Header>
    <ord:SessionHeader>
      <ord:sessionId>00D7A0000009QII!
ARUAQFmpeB0msYS2JcADxYc4PxWo0kAHJP1JiNo5vtvnDBCEvfnWWm07pqbEXY7WadiCmm_Vib6MjhmguBR8
321tI01NQRg</ord:sessionId>
    </ord:SessionHeader>
  </soapenv:Header>
  <soapenv:Body>
    <ord:createCart>
      <ord:request>
        <!--Optional:-->
        <cpq:OrderId>a2n7A0000002tviQAA</cpq:OrderId>
      </ord:request>
    </ord:createCart>
  </soapenv:Body>
</soapenv:Envelope>
```

Example Response

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns="ht
tp://soap.sforce.com/schemas/class/Apttus_Config2/OrderWebService"
xmlns:CreateCartResponseD0="http://soap.sforce.com/schemas/class/Apttus_Config2/
CPQStruct">
  <soapenv:Body>
    <createCartResponse>
      <result>
        <CreateCartResponseD0:CartId>a1I7A000001IqcfUAC</CreateCartResponseD0:Car
tId>
      </result>
    </createCartResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

Synchronizing a Cart to an Order

This API synchronizes shopping cart line items with order line items. Before you synchronize your cart, add products using `addMultiProducts()`, add bundles using `addBundle()`. The API will generally be invoked when the user clicks the Place Order action from the cart.

API Details

| API | Signature |
|------------------------------|---|
| <code>synchronizeCart</code> | <pre>webService static Conga_Config2.CPQStruct.SynchronizeCartResponseDO synchronizeCart(Apttus_Config2.CPQStruct.SynchronizeCartRequestDO request)</pre> |

| Parameters | | |
|----------------------|---|--------------------------|
| Name | Type | Description |
| <code>request</code> | <code>CPQStruct.SynchronizeCartRequestDO</code> | The request data object. |

| Request Data Object - <code>CPQStruct.SynchronizeCartRequestDO</code> | | | |
|---|------|-----------|---|
| Name | Type | Required? | Description |
| <code>CartId</code> | ID | Yes | The Id of the cart (Product Configuration) you want to synchronize. |

API Response

| Response Data Object - CPQStruct.SynchronizeCartResponseDO | | |
|--|---------|--|
| Name | Type | Description |
| IsSuccess | Boolean | Indicates whether synchronizing the cart items was successful. |

Code Sample

The following code sample enables you to synchronize cart items with an order. When the customer has finalized the cart, you can synchronize the products in the cart with the order used to generate the cart. Invoke this API after the user has finalized the cart.

```

1  public void synchronizeCart()
2  {
3      if(String.isNotBlank(cartId))
4      {
5          // Create the request
6          Apttus_Config2.CPQStruct.SynchronizeCartRequestDO request2 = new
Apttus_Config2.CPQStruct.SynchronizeCartRequestDO();
7
8          // Add request parameters
9          request2.CartId = cartId;
10
11         // Synchronize cart
12         Apttus_Config2.CPQStruct.SynchronizeCartResponseDO result2 =
Apttus_Config2.OrderWebService.synchronizeCart(request2);
13
14     } else {
15         ApexPages.addMessage(new
ApexPages.Message(ApexPages.severity.info, 'Invalid or blank cart Id.));
16     }
17 }

```

Request and Response XML

Example Request

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:ord="
http://soap.sforce.com/schemas/class/Apttus_Config2/OrderWebService" xmlns:cpq="http:
//soap.sforce.com/schemas/class/Apttus_Config2/CPQStruct">
  <soapenv:Header>
    <ord:SessionHeader>
      <ord:sessionId>00D7A0000009QII!
ARUAQFmpeB0msYS2JcADxYc4PxWo0kAHJP1JiNo5vtvnDBCEvfnWWm07pqbEXY7WadiCmm_Vib6MjhmGUbR8
321tI01NQRg</ord:sessionId>
    </ord:SessionHeader>
  </soapenv:Header>
  <soapenv:Body>
    <ord:synchronizeCart>
      <ord:request>
        <cpq:CartId>a1I7A000001IqcFUAC</cpq:CartId>
      </ord:request>
    </ord:synchronizeCart>
  </soapenv:Body>
</soapenv:Envelope>
```

Example Response

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns="ht
tp://soap.sforce.com/schemas/class/Apttus_Config2/OrderWebService"
xmlns:SynchronizeCartResponseD0="http://soap.sforce.com/schemas/class/Apttus_Config2/
CPQStruct">
  <soapenv:Body>
    <synchronizeCartResponse>
      <result>
        <SynchronizeCartResponseD0:IsSuccess>>true</SynchronizeCartResponseD0:IsSu
ccess>
      </result>
    </synchronizeCartResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

Accepting an Order

This API finalizes a draft order on customer confirmation. Accepting an order performs the following functions:

- IF Auto Activate Order = False
 - Order and Order Line Item status is set to "Pending".
 - Asset Line Items are created in "Pending" status.
 - If "Ready for Billing Date" is set, the billing schedule generation is initiated.
- IF Auto Activate Order = True
 - Order and Order Line Item status is set to "Activated".
 - Asset Line Items are created in "Activated" status.
 - If "Ready for Billing Date" is set, the billing schedule generation is initiated.

API Details

| API | Signature |
|-------------|--|
| acceptOrder | <pre> webService static Apttus_Config2.CPQStruct.AcceptOrderResponseDO acceptOrder(Apttus_Config2.CPQStruct.AcceptOrderRequestDO) </pre> |

| Parameters | | |
|------------|---|--------------------------|
| Name | Type | Description |
| Request | Apttus_Config2.CPQStruct.AcceptOrderRequestDO | The request data object. |

| Request Data Object - Apttus_Config2.CPQStruct.AcceptOrderRequestDO | | | |
|--|------|-----------|--------------------------------------|
| Field | Type | Required? | Description |
| OrderId | ID | Yes | The Id of the Order to be finalized. |

API Response

| Response Data Object - CPQStruct.AcceptOrderResponseDO | | |
|--|--|--|
| Field | Type | Description |
| IsSuccess | boolean | Specifies whether the order is finalized successfully. |
| AssetLineItems | List<Apttus_Config2__AssetLineItem__c> | List of Asset Line Items |

Code Sample

The following code sample enables you to finalize an order for a valid order with an Order Id.

```

1 public void acceptOrder() {
2
3     Apttus_Config2.CPQStruct.AcceptOrderRequestDO request = new
4     Apttus_Config2.CPQStruct.AcceptOrderRequestDO();
5
6     request.OrderId = orderId;

```

```

6
7     Apttus_Config2.CPQStruct.AcceptOrderResponseDO response =
Apttus_Config2.OrderWebService.acceptOrder(request);
8
9     ApexPages.addMessage(new
    ApexPages.Message(ApexPages.severity.info, 'acceptOrder :' +
response.IsSuccess));
10    ApexPages.addMessage(new
    ApexPages.Message(ApexPages.severity.info, 'List of Assets :' +
response.AssetItems));
11
12    }

```

Integration Details

Use the following information in your integrations with Conga CPQ Web Services API. Refer to [Integrating Conga with External Systems](#) for information on how to get started.

Request and Response XML

Example Request

```

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:ord="
http://soap.sforce.com/schemas/class/Apttus_Config2/OrderWebService" xmlns:cpq="http:
//soap.sforce.com/schemas/class/Apttus_Config2/CPQStruct">
  <soapenv:Header>
    <ord:SessionHeader>
      <ord:sessionId>00D7A0000009QII!
ARUAQFmpeB0msYS2JcADxYc4PxWo0kAHJP1JiNo5vtvnDBCEvfnWWm07pqbEXY7WadiCmm_Vib6MjhmGUbR8
321tI01NQRg</ord:sessionId>
    </ord:SessionHeader>
  </soapenv:Header>
  <soapenv:Body>
    <ord:acceptOrder>
      <ord:request>
        <cpq:OrderId>a2n7A0000002tviQAA</cpq:OrderId>
      </ord:request>
    </ord:acceptOrder>
  </soapenv:Body>
</soapenv:Envelope>

```


Example Response

```

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns="ht
tp://soap.sforce.com/schemas/class/Apttus_Config2/OrderWebService"
xmlns:AcceptOrderResponseD0="http://soap.sforce.com/schemas/class/Apttus_Config2/
CPQStruct" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <acceptOrderResponse>
      <result>
        <AcceptOrderResponseD0:AssetItems xsi:nil="true"/>
        <AcceptOrderResponseD0:IsSuccess>true</AcceptOrderResponseD0:IsSuccess>
      </result>
    </acceptOrderResponse>
  </soapenv:Body>
</soapenv:Envelope>

```

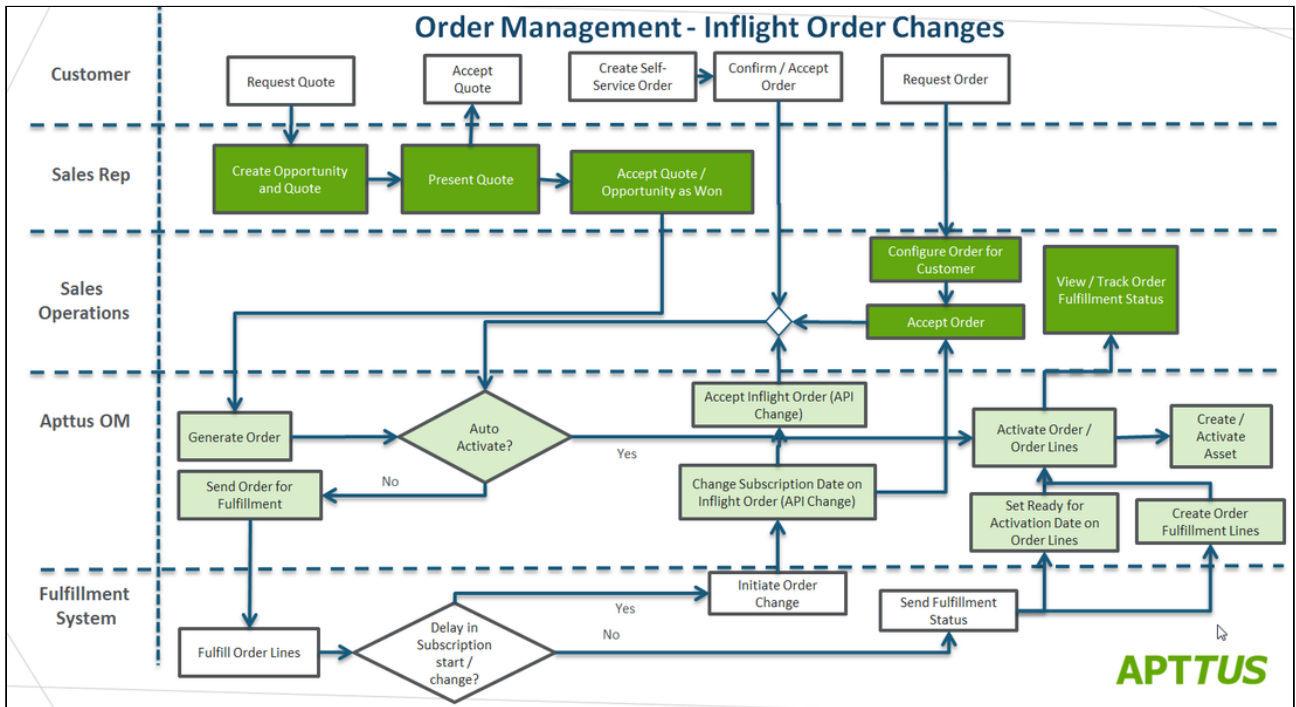
In-Flight Order Changes and Cancellation

In-Flight Order Changes and Cancellation allows you to change or cancel orders after order confirmation and before fulfillment. Changes to orders come in the form of versioned amendments, to existing order line items that include modification of quantity, subscription dates, discounts, and other pricing parameters. You can achieve the following using the APIs:

- Add a new order line item
- Change an existing order line item
- Revert the changes if required
- Cancel an order and its order line items
- Revert cancellation of an order

Using the AmendOrder() API, you can make subscription date changes on order line items of an "In Flight" order (an order with the status Pending, In Fulfillment, or Partially Fulfilled). For example, you may want to amend an order in a scenario when there is a delay in provisioning or fulfillment of the subscription after the order has been confirmed by the customer and before it is fulfilled.

The following diagram visually represents the workflow for In-Flight Order Changes



Refer to the *Order Management on Salesforce Administrator Guide* for complete steps to configure In-Flight Order Changes and Cancellation. At a minimum, the "Enable In-Flight Changes and Cancellation" custom setting must be enabled for any of the following APIs to function. You can also create workflow automation rules for in-flight order changes and cancellation using the `execWorkflow()` API.

In-Flight Changes and Cancellation APIs

Invoke the following APIs to make In-Flight Order Changes to orders (amendments) or to cancel an order.

- [Amending an Order using API](#)
- [Reverting an Amendment to an Order](#)
- [Cancelling an Order using API](#)
- [Reverting a Cancelled Order](#)

Order and Order Line Item Behavior on Amendment

| Action | Behavior |
|---------------------------------|--|
| In-flight Order changes allowed | <ul style="list-style-type: none"> • Change to Start Date and/or End Date • Change the term as needed based on new start date/end date • Recalculate Net Price as needed • Add a new Product (order line) for the existing order: For instance, after placing an order for salesforce licenses, you can add more products for the SFDC Admin Training. • Cancel an order line: Cancel a particular order line from an existing order and keep rest of the order lines unchanged. • Change an existing order line for the existing order: Change the quantity or start/end date for a particular order line on an existing order. Price and Term are recalculated after changing quantity and dates (start/End) respectively. |
| Order Versioning | <ul style="list-style-type: none"> • New order and order line version created to make the in-flight change • The previous version superseded once the new change is confirmed |
| Delayed Asset Creation | <ul style="list-style-type: none"> • Option added to "Create Assets on Order Activation" in Order System Properties. This option needs to be enabled for In-flight change capability. • With this option, assets are created and activated only when the order/order line is activated • For bundles / multi-charge types, assets created/updated when last related option or charge type is fulfilled |

| Action | Behavior |
|--|---|
| Billing Changes | <ul style="list-style-type: none"> • Billing can only be initiated after order/order line activation • "Initiate Billing on Order Activation" is not relevant for In-flight Order changes and is always treated as if the property is enabled. |
| Revenue Recognition | <ul style="list-style-type: none"> • No impact on Revenue Recognition as scheduled, after activation |
| <p>DRAFT or IN AMENDMENT Order</p> <p>Setting Ready for Activation Date or (Ready for Billing Date) on Order Line Item on orders in Draft / In Amendment Order</p> | <p>When the orders are in "Draft" or "In Amendment" status, users can set "Ready for Activation Date" or "Ready for Billing Date" on the order or order line item level. This will not initiate order/order line item activation or billing while order stays in "Draft" or "In Amendment" status.</p> <p>If "Ready for Activation Date" is set on the order or order line item then this will become a trigger to activate order or order line item respectively when the order is accepted.</p> |

| Action | Behavior |
|--|--|
| <p>ACCEPT ORDER / QUOTE</p> <p>Order line item activation on Accept Order or Quote</p> | <p>When the order or quote is accepted, the system uses the following behavior on the order line items irrespective of delayed asset creation flag.</p> <p>READY FOR ACTIVATION DATE ON THE ORDER HEADER</p> <p>Mark all the line items as Fulfilled</p> <p>Initiate logic for order line item activation, which will activate all the line items as well as order header. It will also create and activate assets.</p> <p>INDEPENDENT LINE ITEM:</p> <p>If ready for activation date is set on the order line item and status of the order line is in "In Amendment" or "Draft", then system will mark the line item as "Fulfilled" and initiate "order line item activation logic</p> <p>BUNDLES OR MULTI-CHARGE TYPES WHEN Create Asset on Order Activation = TRUE</p> <p>If ready for activation date is set and status of the order line is in "In Amendment" or "Draft", then system will mark the line item as "Fulfilled"</p> <p>Initiate activation logic, if the current fulfilled item is the last item in the bundle or multi-charge items</p> <p>BUNDLES OR MULTI-CHARGE TYPES WHEN Create Asset on Order Activation = FALSE</p> <p>If ready for activation date is set and status of the order line is in "In Amendment" or "Draft", the system will mark the line item as "Fulfilled"</p> <p>Initiate activation even if there are other items in the bundle or multi-charge types that are still NOT in "Fulfilled" status</p> |

| Action | Behavior |
|--|--|
| <p>PENDING / IN FULFILLMENT / PARTIALLY FULFILLED ORDERS</p> <ul style="list-style-type: none"> Activating Order Line Items | <p>READY FOR ACTIVATION DATE ON THE ORDER HEADER</p> <ul style="list-style-type: none"> Mark all the line items as Fulfilled Initiate logic for order line item activation <p>INDEPENDENT LINE ITEM:</p> <ul style="list-style-type: none"> If ready for activation date is set on the order line item and status of the order line is in "Pending" or "In Fulfillment", then the system will mark the line item as "Fulfilled" and initiate "order line item activation logic" If Fulfilled Quantity >= Delta Quantity and status of the order line is in "Pending" or "In Fulfillment" or "Partially Fulfilled", then mark the line item as "Fulfilled" and initiate "order line item activation logic" <p>BUNDLES OR MULTI-CHARGE TYPES WHEN Create Asset on Order Activation = TRUE</p> <ul style="list-style-type: none"> If ready for activation date is set and status of the order line is in "Pending" or "In Fulfillment", then mark the line item as "Fulfilled" If Fulfilled Quantity >= Delta Quantity and status of the order line is in "Pending" or "In Fulfillment" or "Partially Fulfilled", then mark the line item as "Fulfilled" Initiate activation logic, if the current fulfilled item is the last item in the bundle or multi-charge items <p>BUNDLES OR MULTI-CHARGE TYPES WHEN Create Asset on Order Activation = FALSE</p> <ul style="list-style-type: none"> If ready for activation date is set and status of the order line is in "Pending" or "In Fulfillment", then mark the line item as "Fulfilled" |

| Action | Behavior |
|--------|--|
| | <ul style="list-style-type: none"> • If Fulfilled Quantity \geq Delta Quantity and status of the order line is in "Pending" or "In Fulfillment" or "Partially Fulfilled", then mark the line item as "Fulfilled" • Initiate activation logic, even if there are other items in the bundle or multi-charge types that are still NOT in "Fulfilled" status. |

Use Cases for Changing In-Flight orders

This section highlights the use cases when you would make changes to an In-Flight order, For example, delay in the provisioning of Subscription from a confirmed order, impacts the subscription effective dates, duration, and Price. Accordingly on provisioning, change In-Flight order, order line item, asset, and other related data.

| | |
|--|--|
| <p>New Subscription</p> <p>Change in subscription dates without changing Term/Duration or Quantity</p> | <p>New subscription start is delayed by a month due to provisioning delay.</p> <p>Ordered subscription: 01-Jan-2017 to 31-Dec-2017</p> <p>Provisioned subscription: 01-Feb-2017 to 31-Jan-2018</p> |
| <p>Amend Subscription</p> <p>Change amendment start date without changing the End Date and Quantity</p> | <p>Provisioning of an add-on order on an existing subscription is delayed by a month.</p> <p>Ordered subscription: 01-Aug-2017 to 31-Jan-2018</p> <p>Provisioned subscription: 01-Sep-2017 to 31-Jan-2018</p> |
| <p>Terminate Subscription</p> <p>Change termination date of subscription</p> | <p>Termination of an existing subscription is delayed by a month due to provisioning delay</p> <p>Ordered subscription Termination: End Date: 31-Oct-2017</p> <p>Provisioned subscription Termination: End Date: 30-Nov-2017</p> |

| | |
|---|--|
| <p>Swap Subscription</p> <p>Change start date of upgrade and thus change termination date of original subscription</p> | <p>Start date of the upgrade of an existing subscription is delayed by a month due to provisioning delay. Accordingly, termination of existing subscription also need to be delayed</p> <p>Ordered subscription Upgrade: Gold Warranty: End Date: 31-July-2017 Platinum Warranty: 01-Aug-2017 to 31-Jan-2018</p> <p>Provisioned subscription Upgrade: Gold Warranty: End Date: 31-Aug-2017 Platinum Warranty: 01-Sep-2017 to 31-Jan-2018</p> |
| <p>Add a new order line</p> <p>Addition of n number of products to the existing order</p> | <p>Addition of Dimond Warranty to the existing order</p> <p>Add new order line to the existing order by adding Dimond warranty</p> |
| <p>Cancel an existing Order Line</p> | <p>Canceling Gold Warranty for the existing order</p> <p>Gold Warranty Status is <i>Cancelled</i>.</p> |
| <p>Change an existing order line</p> | <p>Provisioning of an add-on order on an existing subscription is shifted by a month and quantity needs to be increased from 1 to 2</p> <p>Ordered subscription: 01-Aug-2017 to 31-Jan-2018</p> <p>Provisioned subscription: 01-Sep-2017 to 31- Feb -2018 (Quantity: 2)</p> |

| | |
|---|--|
| <p>Calculate the end date by providing a start date, selling term and selling frequency</p> | <p>Provide your input for start date, selling term, selling frequency to the order line item and ensure that the end date on order line item must be null while passing the API.</p> <p>Code Sample:</p> <pre>for(Apttus_Config2_OrderLineItem_c oli: orderLineList) { oli.Apttus_Config2__EndDate__c = null; oli.Apttus_Config2__StartDate__c = oli.Apttus_Config2__StartDate__c.addMonths(4); oli.Apttus_Config2__SellingTerm__c = 24; oli.Apttus_Config2__SellingFrequency__c = 'Monthly'; oli.Apttus_Config2__Quantity__c = oli.Apttus_Config2__Quantity__c; }</pre> |
| <p>Calculate the selling term by providing a start date, end date, and selling frequency</p> | <p>Provide your input for start date, end date, selling frequency to the order line item and ensure that the selling term on order line item must be "0" while passing the API.</p> <p>Code Sample:</p> <pre>for(Apttus_Config2_OrderLineItem_c oli: orderLineList) { oli.Apttus_Config2__SellingTerm__c = 0; oli.Apttus_Config2__EndDate__c = oli.Apttus_Config2__EndDate__c.addMonths(4); oli.Apttus_Config2__StartDate__c = oli.Apttus_Config2__StartDate__c; oli.Apttus_Config2__SellingFrequency__c = 'Monthly'; oli.Apttus_Config2__Quantity__c = oli.Apttus_Config2__Quantity__c; }</pre> |

Amending an Order using API

Make In-Flight order changes. Use this API to make the following changes to Order Line Items (OLI):

- Change the start date or end date of subscription on one or more order line items that are in "Pending", "In Fulfillment", or "Partially Fulfilled" status.
- Make subscription date changes on the standalone item, bundles, and multiple charge line items.
- Cancel order line items in "Pending," "In Fulfillment," or "Partially Fulfilled" status. This changes their status to "Pending Cancellation."

- Override the "Base Price" field.
- Change the Quantity.
- Change the Selling Term, Selling Frequency, and/or Selling Unit-of-Measure (Selling Uom).
- Modify Contract Numbers
- Change the Adjustment Type/Adjustment Amount.

The following changes to an order occur after it has been amended:

- The change in subscription start and/or end date recalculates the price for the line item or bundle after the changes are applied.
- The changes to the order line items are also reflected in the corresponding assets.
- Assets are only created on activation of the order line item.
- Creates a new version of the order to make and track changes and has reference to the previous version. The previous version of the order and order line items are "Superseded", once the work-in-progress (In Amendment) changes are confirmed on "Accept" of the "In Amendment" order.
- Cancelled order line items are assigned the "Pending Cancellation" status. The order must be accepted to confirm cancellation.



- Order line items that are added from catalog (line status = "New") or created through ABO actions (line status In "Amended", "Cancelled", "Renewed", "Upgraded") can be amended.
- Any order configured directly or generated from quote or eCommerce can be amended.

For more information on amending an order, refer to the topic "Managing In-Flight Changes and Cancellation" in the *Order Management on Salesforce User Guide*.

API Details

| API | Signature |
|------------|--|
| amendOrder | <pre>webService static Conga_Config2.CPQStruct.AmendOrderResponseDO amendOrder(Apttus_Config2.CPQStruct.AmendOrderRequestDO)</pre> |

| Parameters | | | |
|--|--|--------------------------|---|
| Name | Type | Description | |
| request | Apttus_Config2.CPQStruct.AmendOrderRequestDO | The request data object. | |
| Request Data Object - Apttus_Config2.CPQStruct.AmendOrderRequestDO | | | |
| Field | Type | Required? | Description |
| OrderId | ID | Yes | <p>Reference to the order. Order Id can be any one of the following:</p> <ul style="list-style-type: none"> • Order in "Pending, In Fulfillment, Partially Fulfilled" status that requires amendment. • Order in "In Amendment" status for which additional line item changes are required. |

| Request Data Object - Apttus_Config2.CPQStruct.AmendOrderRequestDO | | | |
|--|------|-----------|---|
| Field | Type | Required? | Description |
| OrderItems | List | No | <p>List of order line items and changes to make:</p> <ul style="list-style-type: none"> • Order line item id: Reference to the order line item • Start Date: New Start date for the order line item. Not needed for the order line item if the termination date for existing subscription needs to be changed. • End Date: New end date for the order line item. If the termination date needs to be changed on existing subscription then just specify the new end date. • Contract Numbers: If the termination date needs to be changed on existing subscription then set this value to null or same as End Date. • Base Price: New base price for the order line item. • Quantity: New quantity for the order line item (see sample below). • Selling Term / Selling Frequency: The new Selling Term or Selling Frequency for an order line item. • Selling Unit-of-Measure: The new Selling Uom for an order line item. • Adjustment Type / Adjustment Amount: The new price Adjustment Type and/or Adjustment amount for the order line item. |

API Response

| Response Data Object - CPQStruct.AmendOrderResponseDO | | |
|---|----------------------|--------------------------------------|
| Field | Type | Description |
| AmendedOrderItems | List | List of amended order items |
| AmendOrderSO | Standard Data Object | The fields of the AmendOrder Object. |

Code Sample

The following code sample enables you to amend an order for a valid order with an Order Id.

```

1  Id orderid = 'a2n4C0000007NFpQAM';
2  List<Apttus_Config2__OrderLineItem__c> lineitemlist =
3
4  [ Select Id, Name, Apttus_Config2__StartDate__c,Apttus_Config2__EndDate__c
5  from Apttus_Config2__OrderLineItem__c
6  where
7  Id IN ('a2k4C000000D7yt', 'a2k4C000000D7yu', 'a2k4C000000D7yv', 'a2k4C00000
8  0D7yw')];
9
10         lineitemlist[0].Apttus_Config2__StartDate__c=date.newInstance
11         e(2017, 10, 01);
12         lineitemlist[0].Apttus_Config2__EndDate__c=date.newInstance(2
13         018, 8, 31);
14         /*****For line item, where cancellation date needs to be
15         changed, provide the end date and set Contract Numbers field to
16         null*****/
17         lineitemlist[1].Apttus_Config2__EndDate__c=date.newInstance
18         (2018, 9, 30);
19         lineitemlist[1].Apttus_Config2__ContractNumbers__c=null;
20
21         /****to change the quantity of line item ****/
22         lineitemlist[2].Apttus_Config2__Quantity__c=3;
23
24         /***Canceling the line item by changing the status of line
25         item to Pending cancellation *****/
26         lineitemlist[3].Apttus_Config2__Status__c='Pending
27         Cancellation';
28
29         Apttus_Config2.CPQStruct.AmendOrderRequestDO request = new
30         Apttus_Config2.CPQStruct.AmendOrderRequestDO();
31         request.OrderId = orderid; request.OrderItems =
32         lineitemlist;
33         Apttus_Config2.CPQStruct.AmendOrderResponseDO
34         Amend_response = Apttus_Config2.OrderWebService.amendOrder (request);
35         system.debug('OrderID_Of_New_Version -'+Amend_response.Amend
36         OrderSO.id); /**This ID should be passed While Processing Undo Amend
37         request **/
38
39         /***** Accepting new order *****/

```


```

27         Apttus_Config2.CPQStruct.AcceptOrderRequestDO request1 =
new Apttus_Config2.CPQStruct.AcceptOrderRequestDO();
28
29
30         /*Apttus_Config2__Order__c newOrder = [ Select Id, Name,
Apttus_Config2__ActivatedDate__c from Apttus_Config2__Order__c
31         where Apttus_Config2__PreviousVersion__c = :orderid];
32
33         request1.OrderId = newOrder.Id; /** you can either use id
for above query **/
34         OR */
35         request1.OrderId=Amend_response.AmendOrderS0.id // you can
use the id of new order version from above response
36
37
38         /* ***** updating ready for activation date on the
order or order line items for auto activation of the amended lines *****/
39
40         //newOrder.Apttus_Config2__ActivatedDate__c = system.now();
41
42         List<Apttus_Config2__OrderLineItem__c> lineitemlist1 =
43
44         [Select Id, Name, Apttus_Config2__ActivatedDate__c from
Apttus_Config2__OrderLineItem__c where Apttus_Config2__OrderId__c = :
45         .Id and Apttus_Config2__Status__c = 'In Amendment'];
46         lineitemlist1[0].Apttus_Config2__ActivatedDate__c =
system.now();
47         lineitemlist1[1].Apttus_Config2__ActivatedDate__c =
system.now();
48
49         update lineitemlist1;
50
51         Apttus_Config2.CPQStruct.AcceptOrderResponseDO response =
Apttus_Config2.OrderWebService.acceptOrder(request1);

```

Reverting an Amendment to an Order

This API enables you to revert an amendment to an order. For example, you can roll back the changed order date to the previous date. UndoAmendOrder is a composite API.

 For more information on how an order and its line items are affected when you revert an amendment to order, refer to the *Order Management on Salesforce User Guide*.

API Details

| API | | Signature | |
|--|---|---|---|
| undoAmendOrder | | <pre>webservice static Apttus_Config2.CPQStruct.UndoAmendOrderResponseDO undoAmendOrder(Apttus_Config2.CPQStruct.UndoAmendOrderRequestDO)</pre> | |
| Parameters | | | |
| Name | Type | Description | |
| request | Apttus_Config2.CPQStruct.UndoAmendOrderResponseDO | The request data object. | |
| Request Data Object - Apttus_Config2.CPQStruct.UndoAmendOrderRequestDO | | | |
| Field | Type | Required? | Description |
| orderId | ID | Yes | Unique identifier of the amended order. |

API Response

| Response Data Object - CPQStruct.UndoAmendOrderResponseDO | | |
|---|----------------------|--|
| Field | Type | Description |
| PreviousOrderSO | Standard Data Object | The fields of the Previous Order Object. |

Code Sample

Use the following code sample to revert an amendment to an order.

```


1  Apttus_Config2.CPQStruct.UndoAmendOrderRequestDO Undo_Amend_request = new
    Apttus_Config2.CPQStruct.UndoAmendOrderRequestDO();
2  Undo_Amend_request.orderid = 'a2n4C0000007NGJ'; /** This is the Id of the
    order whose status is 'In Amendment' **/
3  Apttus_Config2.CPQStruct.UndoAmendOrderResponseDO UndoAmend_response =
    Apttus_Config2.OrderWebService.undoAmendOrder(Undo_Amend_request);
4  system.debug('Order_OLD_Version - ' +
    UndoAmend_response.PreviousOrderSO.id); /** returns the OrderId of the
    previous version **/

```

Canceling an Order using API

This API cancels an order. The status of the order and its line items changes to *Pending Cancellation* and a new version is created. The status of the previous order is set to *Being Cancelled*. Please note the following:

- If you decide to revert (undo) the cancelled order, the new order version is deleted and the previous order version shows *Pending* status.
- If you decide to cancel the order, you must accept the cancellation of the order. After accepting the cancellation, a new version is created showing with *Cancelled* status for the order as well as its line items. In this case, the previous version shows the *Superseded* status for the order and its line items.

 For more information on how an order and its line items are affected when you cancel an order, refer to the *Order Management on Salesforce User Guide*.

API Details

| API | Signature |
|-------------|--|
| cancelOrder | <pre> webService static Apttus_Config2.CPQStruct.CancelOrderResponseDO cancelOrder(Apttus_Config2.CPQStruct.CancelOrderRequestDO) </pre> |

| Parameters | | |
|------------|---|--------------------------|
| Name | Type | Description |
| request | Apttus_Config2.CPQStruct.CancelOrderRequestDO | The request data object. |

| Request Data Object - Apttus_Config2.CPQStruct.CancelOrderRequestDO | | | |
|---|------|-----------|---|
| Field | Type | Required? | Description |
| orderId | ID | Yes | Unique identifier of the order to be cancelled. |

API Response

| Response Data Object - CPQStruct.CancelOrderResponseDO | | |
|--|----------------------|--|
| Field | Type | Description |
| cancelOrderSO | Standard Data Object | The fields of the Cancel Order Object. |

Code Sample

The following code sample allows you to cancel an order.

```

ID Orderid = 'a2n4C0000007NFpQAM';
Apttus_Config2.CPQStruct.CancelOrderRequestDO CancelRequest = new
Apttus_Config2.CPQStruct.CancelOrderRequestDO();
CancelRequest.orderid=Orderid;
Apttus_Config2.CPQStruct.CancelOrderResponseDO Cancel_Response =
Apttus_Config2.OrderWebService.cancelOrder(CancelRequest);
system.debug('OrderID_New_Version - '+Cancel_Response.CancelOrderSO.id); /**This ID
should be passed While Processing Undo Cancel request **/

```

Integration Details

Use the following information in your integrations with Conga CPQ Web Services API. Refer to [Integrating Conga with External Systems](#) for information on how to get started.

API Prerequisites

None.

Request and Response XML

Example Request

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:ord="
http://soap.sforce.com/schemas/class/Apttus_Config2/OrderWebService" xmlns:cpq="http:
//soap.sforce.com/schemas/class/Apttus_Config2/CPQStruct">
  <soapenv:Header>
    <ord:SessionHeader>
      <ord:sessionId>00D7A0000009QII!
ARUAQFmpeB0msYS2JcADxYc4PxWo0kAHJP1JiNo5vtvnDBCEvfnWWm07pqbEXY7WadiCmm_Vib6MjhmgUbR8
321tI01NQRg</ord:sessionId>
    </ord:SessionHeader>
  </soapenv:Header>
  <soapenv:Body>
    <ord:cancelOrder>
      <ord:request>
        <cpq:OrderId>a2n7A0000002tviQAA</cpq:OrderId>
      </ord:request>
    </ord:cancelOrder>
  </soapenv:Body>
</soapenv:Envelope>
```

Example Response

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns="ht
tp://soap.sforce.com/schemas/class/Apttus_Config2/OrderWebService"
xmlns:CancelOrderResponseD0="http://soap.sforce.com/schemas/class/Apttus_Config2/
CPQStruct" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
```

```

<cancelOrderResponse>
  <result>
    <CancelOrderResponseD0:CancelOrderS0 xsi:type="Apttus_Config2__Order__c">
      <Id>a2n7A0000002twXQAQ</Id>
      <Apttus_Config2__AutoActivateOrder__c>>false</
Apttus_Config2__AutoActivateOrder__c>
      <Apttus_Config2__BillToAccountId__c>0017A00000WW0pZQAT</
Apttus_Config2__BillToAccountId__c>
      <Apttus_Config2__BillingPreferenceId__c>a2d2f0000000h0YAAQ</
Apttus_Config2__BillingPreferenceId__c>
      <Apttus_Config2__ConfigurationSyncDate__c>2020-07-06T08:43:16.000Z</
Apttus_Config2__ConfigurationSyncDate__c>
      <Apttus_Config2__IsTaskPending__c>>false</
Apttus_Config2__IsTaskPending__c>
      <Apttus_Config2__OrderAmount__c>300.00000</
Apttus_Config2__OrderAmount__c>
      <Apttus_Config2__OrderDate__c>2020-07-06T06:33:53.000Z</
Apttus_Config2__OrderDate__c>
      <Apttus_Config2__OriginalOrderNumber__c>0-00005958</
Apttus_Config2__OriginalOrderNumber__c>
      <Apttus_Config2__PreviousVersion__c>a2n7A0000002tviQAA</
Apttus_Config2__PreviousVersion__c>
      <Apttus_Config2__PriceListId__c>a173C000000fHFQQA2</
Apttus_Config2__PriceListId__c>
      <Apttus_Config2__PricingDate__c>2020-07-06T06:33:53.000Z</
Apttus_Config2__PricingDate__c>
      <Apttus_Config2__ReadyForWorkflow__c>>false</
Apttus_Config2__ReadyForWorkflow__c>
      <Apttus_Config2__ShipToAccountId__c>0017A00000WW0pZQAT</
Apttus_Config2__ShipToAccountId__c>
      <Apttus_Config2__SoldToAccountId__c>0017A00000WW0pZQAT</
Apttus_Config2__SoldToAccountId__c>
      <Apttus_Config2__SourceChannel__c>Direct</
Apttus_Config2__SourceChannel__c>
      <Apttus_Config2__Source__c>Account</Apttus_Config2__Source__c>
      <Apttus_Config2__Status__c>Pending Cancellation</
Apttus_Config2__Status__c>
      <Apttus_Config2__Type__c>New</Apttus_Config2__Type__c>
      <Apttus_Config2__VersionNumber__c>1</Apttus_Config2__VersionNumber__c>
      <CurrencyIsoCode>USD</CurrencyIsoCode>
    </CancelOrderResponseD0:CancelOrderS0>
  </result>
</cancelOrderResponse>
</soapenv:Body>

```

```
</soapenv:Envelope>
```

Reverting a Cancelled Order

This API reverts the cancellation of an order.

i For more information on how the order and its line items are affected when you revert a cancelled order, refer to the *Order Management on Salesforce User Guide*.

API Details

| API | Signature |
|-----------------|--|
| undoCancelOrder | <i>webservice static</i> <i>Apttus_Config2.CPQStruct.UndoCancelOrderResponseDO</i> <i>undoCancelOrder(Apttus_Config2.CPQStruct.UndoCancelOrderRequestDO)</i> |

| Parameters | | | |
|---|---|--------------------------|---|
| Name | Type | Description | |
| Request | Apttus_Config2.CPQStruct.UndoCancelOrderRequestDO | The request data object. | |
| Request Data Object - Apttus_Config2.CPQStruct.UndoCancelOrderRequestDO | | | |
| Field | Type | Required? | Description |
| orderId | ID | Yes | Unique identifier of the cancelled order. |

API Response

| Response Data Object - CPQStruct.UndoCancelOrderResponseDO | | |
|--|----------------------|--|
| Field | Type | Description |
| PreviousOrderSO | Standard Data Object | The fields of the Previous Order Object. |

Code Sample

The following API lets you undo or revert the cancelled order.

```
Apttus_Config2.CPQStruct.UndoCancelOrderRequestDO UndoCancelrequest = new
  Apttus_Config2.CPQStruct. UndoCancelOrderRequestDO();
UndoCancelrequest.orderid=Cancel_Response.CancelOrderSO.id;
Apttus_Config2.CPQStruct.UndoCancelOrderResponseDO UndoCancel_Response=
Apttus_Config2.OrderWebService.undoCancelOrder (UndoCancelrequest);
system.debug('OrderID_OLD_Version - '+UndoCancel_Response.PreviousOrderSO.id);/**
returns the OrderId of the previous version **/
```

Integration Details

Use the following information in your integrations with Conga CPQ Web Services API. Refer to [Integrating Conga with External Systems](#) for information on how to get started.

API Prerequisites

None.

Request and Response XML

Example Request

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:ord="
http://soap.sforce.com/schemas/class/Apttus_Config2/OrderWebService" xmlns:cpq="http:
//soap.sforce.com/schemas/class/Apttus_Config2/CPQStruct">
  <soapenv:Header>
    <ord:SessionHeader>
      <ord:sessionId>00D7A0000009QII!
ARUAQFmpeB0msYS2JcADxYc4PxWo0KAHJP1JiNo5vtvnDBCEvfnWWm07pqbEXY7WadiCmm_Vib6MjhmgUbR8
321tI01NQRg</ord:sessionId>
    </ord:SessionHeader>
```

```

</soapenv:Header>
<soapenv:Body>
  <ord:undoCancelOrder>
    <ord:request>
      <cpq:OrderId>a2n7A0000002twXQAQ</cpq:OrderId>
    </ord:request>
  </ord:undoCancelOrder>
</soapenv:Body>
</soapenv:Envelope>

```

Example Response

```

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns="http://soap.sforce.com/schemas/class/Apttus_Config2/OrderWebService"
  xmlns:UndoCancelOrderResponseD0="http://soap.sforce.com/schemas/class/Apttus_Config2/CPQStruct" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <undoCancelOrderResponse>
      <result>
        <UndoCancelOrderResponseD0:PreviousOrderSO xsi:type="Apttus_Config2__Order__c">
          <Id>a2n7A0000002tviQAA</Id>
          <Apttus_Config2__AutoActivateOrder__c>>false</
Apttus_Config2__AutoActivateOrder__c>
          <Apttus_Config2__BillToAccountId__c>0017A00000WW0pZQAT</
Apttus_Config2__BillToAccountId__c>
          <Apttus_Config2__BillingPreferenceId__c>a2d2f0000000h0YAAQ</
Apttus_Config2__BillingPreferenceId__c>
          <Apttus_Config2__ConfigurationSyncDate__c>2020-07-06T08:43:16.000Z</
Apttus_Config2__ConfigurationSyncDate__c>
          <Apttus_Config2__IsTaskPending__c>>false</
Apttus_Config2__IsTaskPending__c>
          <Apttus_Config2__OrderAmount__c>300.00000</
Apttus_Config2__OrderAmount__c>
          <Apttus_Config2__OrderDate__c>2020-07-06T06:33:53.000Z</
Apttus_Config2__OrderDate__c>
          <Apttus_Config2__OriginalOrderNumber__c>0-00005958</
Apttus_Config2__OriginalOrderNumber__c>
          <Apttus_Config2__PriceListId__c>a173C000000fHFQA2</
Apttus_Config2__PriceListId__c>

```

```

        <Apttus_Config2__PricingDate__c>2020-07-06T06:33:53.000Z</
Apttus_Config2__PricingDate__c>
        <Apttus_Config2__ReadyForWorkflow__c>>false</
Apttus_Config2__ReadyForWorkflow__c>
        <Apttus_Config2__ShipToAccountId__c>0017A00000WW0pZQAT</
Apttus_Config2__ShipToAccountId__c>
        <Apttus_Config2__SoldToAccountId__c>0017A00000WW0pZQAT</
Apttus_Config2__SoldToAccountId__c>
        <Apttus_Config2__SourceChannel__c>Direct</
Apttus_Config2__SourceChannel__c>
        <Apttus_Config2__Source__c>Account</Apttus_Config2__Source__c>
        <Apttus_Config2__Status__c>Draft</Apttus_Config2__Status__c>
        <Apttus_Config2__Type__c>New</Apttus_Config2__Type__c>
        <CurrencyIsoCode>USD</CurrencyIsoCode>
        <Name>0-00005958</Name>
    </UndoCancelOrderResponseD0:PreviousOrderS0>
</result>
</undoCancelOrderResponse>
</soapenv:Body>
</soapenv:Envelope>

```

Cloning an Order using SOAP API

This API performs a deep clone of an order. Creates a new copy of the order along with its related objects rather than creating a new version. Deep clone can only be performed for orders where all order line items have the status "New." Any order with ABO (Asset-Based Ordering) line items cannot be cloned in this way.

The following applies when the CloneOrder() API is invoked:

- Cloning an existing order sets the status of the order header and order line items to "Draft".
- Cloning an existing order resets the following fields: Asset Line Item, Ready for Activation Date, and Ready for Billing Date.
- Clones the corresponding cart and cart line items. Cart line items can then be configured and change similar to new draft order.

API Details

| API | Signature |
|------------|---|
| cloneOrder | <i>WebService static</i> <i>Apttus_Config2.CPQStruct.CloneOrderResponseDO</i> <i>cloneOrder(Apttus_Config2.CPQStruct.CloneOrderRequestDO)</i> |

| Parameters | | |
|------------|--|--------------------------|
| Name | Type | Description |
| request | Apttus_Config2.CPQStruct.CloneOrderRequestDO | The request data object. |

| Request Data Object - Apttus_Config2.CPQStruct.CloneOrderRequestDO | | | |
|---|------|-----------|--|
| Field | Type | Required? | Description |
| OrderId | ID | Yes | The order Id to clone. This is the order number that remains constant from version to version and is referred by the customer. |

API Response

| Response Data Object - CPQStruct.CloneOrderResponseDO | | |
|---|--------|---------------------------------------|
| Field | Type | Description |
| OrderItems | List | List of cloned order items. |
| CloneOrderSO | Object | The fields of the Clone Order object. |

Code Sample

The following code sample enables you to clone an order for a valid order with an Order Id.

```

1  /***** Clone an order *****/
2  Id orderid = 'a2n3C000000CfAC';
3  Apttus_Config2.CPQStruct.CloneOrderRequestDO request = new
   Apttus_Config2.CPQStruct.CloneOrderRequestDO();
4  request.OrderId = orderid;
5
6
7  Apttus_Config2.CPQStruct.CloneOrderResponseDO response
   =Apttus_Config2.OrderWebService.cloneOrder (request);

```

Integration Details

Use the following information in your integrations with Conga CPQ Web Services API. Refer to [Integrating Conga with External Systems](#) for information on how to get started.

API Prerequisites

None.

Request and Response XML

Example Request

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:ord="
http://soap.sforce.com/schemas/class/Apttus_Config2/OrderWebService" xmlns:cpq="http:
//soap.sforce.com/schemas/class/Apttus_Config2/CPQStruct">
  <soapenv:Header>
    <ord:SessionHeader>
      <ord:sessionId>00D7A0000009QII!
ARUAQFmpeB0msYS2JcADxYc4PxWo0kAHJP1JiNo5vtvnDBCEvfnWWm07pqbEXY7WadiCmm_Vib6MjhmguBR8
321tI01NQRg</ord:sessionId>
    </ord:SessionHeader>
  </soapenv:Header>
  <soapenv:Body>
    <ord:cloneOrder>
      <ord:request>
        <cpq:OrderId>a2n7A0000002tviQAA</cpq:OrderId>
      </ord:request>
    </ord:cloneOrder>
  </soapenv:Body>
</soapenv:Envelope>
```

Example Response

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns="ht
tp://soap.sforce.com/schemas/class/Apttus_Config2/OrderWebService"
xmlns:CloneOrderResponseD0="http://soap.sforce.com/schemas/class/Apttus_Config2/
CPQStruct" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <cloneOrderResponse>
      <result>
        <CloneOrderResponseD0:CloneOrderS0 xsi:type="Apttus_Config2__Order__c">
          <Id>a2n7A0000002tvsQAA</Id>
          <Apttus_Config2__AutoActivateOrder__c>>false</
Apttus_Config2__AutoActivateOrder__c>
          <Apttus_Config2__BillToAccountId__c>0017A00000WW0pZQAT</
Apttus_Config2__BillToAccountId__c>
```

```

    <Apttus_Config2__BillingPreferenceId__c>a2d2f0000000h0YAAQ</
Apttus_Config2__BillingPreferenceId__c>
    <Apttus_Config2__ConfigurationSyncDate__c>2020-07-06T07:29:47.000Z</
Apttus_Config2__ConfigurationSyncDate__c>
    <Apttus_Config2__IsTaskPending__c>>false</
Apttus_Config2__IsTaskPending__c>
    <Apttus_Config2__OrderDate__c>2020-07-06T06:33:53.000Z</
Apttus_Config2__OrderDate__c>
    <Apttus_Config2__PriceListId__c>a173C000000fHFQQA2</
Apttus_Config2__PriceListId__c>
    <Apttus_Config2__PricingDate__c>2020-07-06T06:33:53.000Z</
Apttus_Config2__PricingDate__c>
    <Apttus_Config2__ReadyForWorkflow__c>>false</
Apttus_Config2__ReadyForWorkflow__c>
    <Apttus_Config2__ShipToAccountId__c>0017A00000WW0pZQAT</
Apttus_Config2__ShipToAccountId__c>
    <Apttus_Config2__SoldToAccountId__c>0017A00000WW0pZQAT</
Apttus_Config2__SoldToAccountId__c>
    <Apttus_Config2__SourceChannel__c>Direct</
Apttus_Config2__SourceChannel__c>
    <Apttus_Config2__Source__c>Account</Apttus_Config2__Source__c>
    <Apttus_Config2__Status__c>Draft</Apttus_Config2__Status__c>
    <Apttus_Config2__Type__c>New</Apttus_Config2__Type__c>
    <CurrencyIsoCode>USD</CurrencyIsoCode>
  </CloneOrderResponseD0:CloneOrderS0>
</result>
</cloneOrderResponse>
</soapenv:Body>
</soapenv:Envelope>

```

Fetching Price Agreements

This API fetches price agreements based on proposals or agreements.

API Details

| API | | Signature |
|--------------------|--|--|
| GetPriceAgreements | | <i>webService</i> <i>static</i> <i>CPQApiStruct.GetPriceAgreementsResponseDOGetPriceAgreements(CPQApiStruct.GetPriceAgreementsRequestDO)</i> |
| Parameters | | |
| Name | Type | Description |
| request | CPQApiStruct.GetPriceAgreementsRequestDO | The request data object. |

| Request Data Object - CPQApiStruct.GetPriceAgreementsRequestDO | | | |
|--|-----------------------|-----------|--|
| Field | Type | Required? | Description |
| PriceAgreementType | String | Yes | The type of price agreement. Valid Values: Proposal, Agreement. |
| Criteria | CPQApiStruct.FilterDO | No | The criteria details like API agreement name and proposal. This is based on FilterDo object. |
| Fields | List<String> | No | The list of fields to be displayed in the response data object based on the price agreement type. |
| AccountIds | List<Id> | No | The account IDs passed to fetch the price agreements. This field narrows down the price agreements based on the filter of account IDs. |

CPQApiStruct.FilterDO

This is an API object that feeds to criteria variable of below request objects:

- CPQApiStruct.GetPriceAgreementsRequestDO
- GetPriceAgreements CPQApiStruct.GetPriceAgreementItemsRequestDO

| Request Data Object - CPQApiStruct.FilterDO | | | |
|---|--------------------------------|-----------|---|
| Field | Type | Required? | Description |
| SObjectName | String | Y | API name of the subject valid values are: Apttus_Proposal__Proposal__c, Apttus__APTS_Agreement__c |
| Predicates | List<CPQApiStruct.PredicateDO> | Y | List of conditions in where clause |
| Condition Expr | String | N | To define the complex logic among multiple conditions. Example for 2 conditions (1 OR 2) |

CPQApiStruct.PredicateDO

This API defines a filter based on which criteria specifications are set. List of this needs to feed Predicates variable of FilterDO object type variable.

| Parameters | | |
|------------|--------------------------|--------------------------|
| Name | Type | Description |
| request | CPQApiStruct.PredicateDO | The request data object. |

| Request Data Object - CPQApiStruct.PredicateDO | | | |
|--|---------|-----------|----------------------------|
| Field | Type | Required? | Description |
| RowNum | Integer | Y | The row numbers. |
| FieldName | String | Y | The API name of the field. |

| Request Data Object - CPQApiStruct.PredicateDO | | | |
|--|--------|-----------|---|
| Field | Type | Required? | Description |
| CompOper | String | Y | The type of query comparator. Valid values: equal to, not equal to, less than, less than or equal to, greater than, greater than or equal to, in, not in, like, not like, includes, excludes, starts with, ends with, contains, does not contain. |
| FieldValue | String | Y | The value based on which data should be fetched. |

API Response

| Response Data Object - CPQApiStruct.GetPriceAgreementsResponseDO | | |
|--|---------------|---|
| Field | Type | Description |
| PriceAgreements | List<sObject> | The list of price agreements fetched based on the specified input fields. |

Code Sample

The following sample enables you to create an order for a valid account with an Account Id and a Price List with a valid Price List Id. Using the sample below, you can search for a valid account using an account number. If an account exists with the account number entered, you can create an order using the createorder() API. You can invoke this API in use cases when you want to create an order page based on the account and price list.

```

1  Apttus_Config2.CPQApiStruct.GetPriceAgreementsRequestDO request = new
    Apttus_Config2.CPQApiStruct.GetPriceAgreementsRequestDO();
2
3  // possible values Proposal,Agreement
4  request.PriceAgreementType = 'Proposal';
5
6  Apttus_Config2.CPQApiStruct.FilterDO criteriaSpec = new
    Apttus_Config2.CPQApiStruct.FilterDO();
7
8  // Proposal,Agreement subjectApi name
9  criteriaSpec.SObjectName = 'Apttus_Proposal__Proposal__c';
10
11 // setup predicates
12 Apttus_Config2.CPQApiStruct.PredicateDO predicate = new
    Apttus_Config2.CPQApiStruct.PredicateDO();
13 predicate.RowNum = 1;
14 predicate.FieldName = 'Name';
15 predicate.CompOper = '=';
16 predicate.FieldValue = 'Q-11122';
17 criteriaSpec.Predicates.add(predicate);
18 request.Criteria = criteriaSpec;
19
20 List<String> fieldNames = new List<String>{'Id','Name'};
21 request.Fields.addAll(fieldNames);
22
23
24 List<Id> accIds = new List<Id>();
25 accIds.add('0015000000Fn6m0');
26
27 request.AccountIds.addAll(accIds);
28
29 //select Id, Name From Apttus_Proposal__Proposal__c Where
    Apttus_Proposal__Approval_Stage__c =:Accepted
30

```

```

31 Apttus_CPQApi.PriceAgreementApiCallback apiCall = new
    Apttus_CPQApi.PriceAgreementApiCallback();
32 Apttus_Config2.CPQApiStruct.GetPriceAgreementsResponseDO result =
    apiCall.getPriceAgreements(request);
33 List<SObject> PriceAgreements = new List<SObject>();
34
35 if (!result.PriceAgreements.isEmpty())
36 PriceAgreements = result.PriceAgreements;
37 system.debug(PriceAgreements);

```

Fetching Price Agreement Items

GetPriceAgreementItems()

This API fetches price agreement items for a specified price agreement type.

API Details

| API | Signature | | |
|---|--|--------------------------|--|
| createOrder | <i>webservice static</i> <i>CPQApiStruct.GetPriceAgreementItemsResponseDOGetPriceAgreementItems(CPQApiStruct.GetPriceAgreementItemsRequestDO)</i> | | |
| Parameters | | | |
| Name | Type | Description | |
| request | CPQApiStruct.GetPriceAgreementItemsRequestDO | The request data object. | |
| Request Data Object - CPQApiStruct.GetPriceAgreementItemsRequestDO | | | |
| Field | Type | Required? | Description |
| PriceAgreement Type | String | Yes | The type of price agreement. Valid Values: Proposal, Agreement. |
| Criteria | CPQApiStruct.FilterDo | No | The criteria details like API agreement name and proposal. This is based on FilterDo object. |

| Request Data Object - CPQApiStruct.GetPriceAgreementItemsRequestDO | | | |
|--|--------------|-----------|---|
| Field | Type | Required? | Description |
| fieldNames | List<String> | No | The list of fields to be displayed in the response data object based on the price agreement type. |
| priceAgreement Numbers | List<String> | No | The list of price agreements numbers. |

CPQApiStruct.FilterDO

This is an API object that feeds to criteria variable of below request objects:

- CPQApiStruct.GetPriceAgreementsRequestDO
- GetPriceAgreements CPQApiStruct.GetPriceAgreementItemsRequestDO

| Request Data Object - CPQApiStruct.FilterDO | | | |
|---|--------------------------------|-----------|---|
| Field | Type | Required? | Description |
| SObjectName | String | Y | API name of the subject valid values are: Apttus_Proposal__Proposal__c, Apttus__APTS_Agreement__c |
| Predicates | List<CPQApiStruct.PredicateDO> | Y | List of conditions in where clause |
| ConditionExpr | String | N | To define the complex logic among multiple conditions. Example for 2 conditions (1 OR 2) |

CPQApiStruct.PredicateDO

This API defines a filter based on which criteria specifications are set. List of this needs to feed Predicates variable of FilterDO object type variable.

| Parameters | | | |
|--|--------------------------|--------------------------|---|
| Name | Type | Description | |
| request | CPQApiStruct.PredicateDO | The request data object. | |
| Request Data Object - CPQApiStruct.PredicateDO | | | |
| Field | Type | Required? | Description |
| RowNum | Integer | Y | The row numbers. |
| FieldName | String | Y | The API name of the field. |
| Comparator | String | Y | The type of query comparator. Valid values: equal to, not equal to, less than, less than or equal to, greater than, greater than or equal to, in, not in, like, not like, includes, excludes, starts with, ends with, contains, does not contain. |

| Request Data Object - CPQApiStruct.PredicateDO | | | |
|--|--------|-----------|--|
| Field | Type | Required? | Description |
| FieldValue | String | Y | The value based on which data should be fetched. |

API Response

| Response Data Object - CPQApiStruct.GetPriceAgreementItemsResponseDO | | |
|--|---------------|--|
| Field | Type | Description |
| PriceAgreementItems | List<sObject> | The list of price agreement items fetched based on the specified input fields. |

Code Sample

```

1  Apttus_Config2.CPQApiStruct.GetPriceAgreementItemsRequestDO request = new
    Apttus_Config2.CPQApiStruct.GetPriceAgreementItemsRequestDO ();
2  // possible values Proposal,Agreement
3  request.PriceAgreementType = 'Proposal';
4
5  Apttus_Config2.CPQApiStruct.FilterDO criteriaSpec = new
    Apttus_Config2.CPQApiStruct.FilterDO();
6  // subject name
7
8  criteriaSpec.SObjectName = 'Apttus_Proposal__Proposal_Line_Item__c';
9  // setup predicates
10 Apttus_Config2.CPQApiStruct.PredicateDO predicate = new
    Apttus_Config2.CPQApiStruct.PredicateDO();
11 predicate.RowNum = 1;
12 predicate.FieldName =
    'Apttus_Proposal__Proposal__r.Apttus_Proposal__Approval_Stage__c';
13 predicate.CompOper = 'equal to';
14 predicate.FieldValue = 'Accepted';
15
16 Apttus_Config2.CPQApiStruct.PredicateDO predicate2 = new
    Apttus_Config2.CPQApiStruct.PredicateDO();
17 predicate2.RowNum = 1;

```


```

18 predicate2.FieldName =
   'Apttus_Proposal__Proposal__r.Apttus_Proposal__ExpectedStartDate__c';
19 predicate2.CompOper = '=';
20 predicate2.FieldValue = ':Apttus_QPConfig__StartDate__c';
21
22 criteriaSpec.Predicates.add(predicate2);
23
24 criteriaSpec.ConditionExpr = '1 AND 2';
25
26 List<String> fieldNames = new List<String>{'Id','Name'};
27 request.Fields.addAll(fieldNames);
28
29 List<String> priceAgreementNumbers = new List<String>();
30 priceAgreementNumbers.add('Q-00002175');
31
32 // contract numbers
33 request.PriceAgreementNumbers.addAll(priceAgreementNumbers);
34
35
36
37 Apttus_CPQApi.PriceAgreementApiCallback apiCall = new
   Apttus_CPQApi.PriceAgreementApiCallback();
38 Apttus_Config2.CPQApiStruct.GetPriceAgreementItemsResponseDO result =
   apiCall.getPriceAgreementItems(request);
39 system.debug(result.PriceAgreementItems);

```

Creating a Partial Order from a Quote or Agreement

This API creates a partial order from a quote or agreement.

 For detailed information on partial orders, refer to the *Order Management on Salesforce User Guide*.

API Details

| API | Signature |
|--------------------|--|
| CreateOrderRequest | <i>webservice static</i> <i>CPQApi.CPQ.CreateOrderResponseDO CreateOrderRequest(</i> <i>CPQApi.CPQ.CreateOrderRequestDO)</i> |

| Parameters | | | |
|---|--|--------------------------|---|
| Name | Type | Description | |
| request | CPQApi.CPQ.CreateOrderRequestDO | The request data object. | |
| Request Data Object - CPQApi.CPQ.CreateOrderRequestDO | | | |
| Field | Type | Required? | Description |
| CustomFields | List<String> | No | List of custom fields on orderSO |
| OrderInput | Apttus_Config2__Order__c | Yes | OrderSO |
| OrderItems | List<Apttus_CPQApi.CPQ.SelectedProductDO> | No | List of order items |
| OrderItemSource | String | Yes | Source of order items |
| Properties | List<Apttus_Config2.Property> | No | List of properties for cart created for order |
| Request Data Object - CPQApi.CPQ.SelectedProductDO | | | |
| Field | Type | Required? | Description |
| AttributeValues | List<Apttus_Config2__ProductAttributeValue__c> | No | The product attribute value for the product |
| Comments | String | No | Comments |
| CopyBundleConfigurationFromSource | Boolean | No | For bundles, to copy bundle configuration |
| CustomData | Apttus_Config2__LineItem__c | No | Custom data line item |
| CustomFields | List<String> | No | List of custom fields |
| EndDate | Date | No | End date |

| Request Data Object - CPQApi.CPQ.SelectedProductDO | | | |
|--|--|-----------|---|
| Field | Type | Required? | Description |
| ProductId | Id | No | Product Id |
| Quantity | Decimal | No | Quantity |
| RelatedLineItems | List<Apttus_Config2__RelatedLineItem__c> | No | List of related line items |
| SellingTerm | Decimal | No | Selling term |
| SourceFields | List<String> | No | List of source custom fields to be copied |
| SourceId | Id | No | Id of a source item |
| StartDate | Date | No | Start Date |

Code Sample

The following sample uses an explicit product definition for order items while creating partial order from quote/agreement.

```

1  Apttus_CPQApi.CPQ.CreateOrderRequestDO request
2  = new Apttus_CPQApi.CPQ.CreateOrderRequestDO();
3
4  // order item source
5
6  request.OrderItemSource = Apttus_CPQApi.CPQ.ITEMSOURCE_QUOTE_LINEITEM;
7
8
9  // create order input
10
11 Apttus_Config2__Order__c inputSO = new Apttus_Config2__Order__c();
12
13
14 inputSO.Apttus_Config2__SoldToAccountId__c = '0014C00000F9Zz3';
15
16 inputSO.Apttus_Config2__PriceListId__c = 'a174C0000001MQ1';

```

```
17
18 inputSO.Apttus_Config2__PricingDate__c = Datetime.now();
19
20 inputSO.Apttus_Config2__OrderDate__c = Datetime.now();
21
22 inputSO.Apttus_Config2__OrderStartDate__c = Date.today();
23
24 inputSO.Apttus_Config2__OrderEndDate__c = Date.today().addMonths(12);
25
26 // add custom fields to orderSO (optional)
27
28
29 inputSO.put('Apttus_QPConfig__ProposalId__c','a0Y4C0000021Cm5');
30
31
32 request.OrderInput = inputSO;
33
34 request.CustomFields.add('Apttus_QPConfig__ProposalId__c');
35
36 // add order items (optional)
37
38
39 // selected item
40
41
42 Apttus_CPQApi.CPQ.SelectedProductDO productDO
43 = new Apttus_CPQApi.CPQ.SelectedProductDO();
44
45
46 productDO.Comments = 'Test Create Order API';
47
48
49 productDO.CopyBundleConfigurationFromSource = false;
50
51 productDO.StartDate = Date.today();
52
53 productDO.EndDate = Date.today().addMonths(12);
54
55 productDO.SellingTerm = 12;
56
57 productDO.Quantity = 1;
58
59 productDO.ProductId = '01t4C000001pB1J';
60
61 // add attributes to the selected product (optional)
```

```
62
63
64 Apttus_Config2__ProductAttributeValue__c
65 attVal = new Apttus_Config2__ProductAttributeValue__c();
66
67
68 attVal.APTS_TestAttColor__c = 'Black';
69
70
71 productDO.AttributeValues.add(attVal);
72
73 // add related line items to selected product (optional)
74
75
76 Apttus_Config2__RelatedLineItem__c
77 lineItem = new Apttus_Config2__RelatedLineItem__c();
78
79
80 lineItem.Apttus_Config2__RelationType__c = 'Wallet';
81
82
83 productDO.RelatedLineItems.add(lineItem);
84
85
86 // source id and the explicit product definition are mutually exclusive
87
88
89 // productDO.SourceId = 'a0W4C000001MRRF';
90
91
92 request.OrderItems.add(productDO);
93
94 // order properties (optional)
95
96
97 // request.Properties.add(new
98 Apttus_Config2.Property('useAdvancedApproval' 'false'));
99
100
101 // request.Properties.add(new Apttus_Config2.Property('useDealOptimizer'
102 'false'));
103
104
```



```

105 // request.Properties.add(new Apttus_Config2.Property('retId',
106 'a0Y4C0000021Cm5'));
107
108
109 // request.Properties.add(new Apttus_Config2.Property('cntrNbr_1',
110 'Q-00004442'));
111
112
113 // create order api call
114
115
116 Apttus_CPQApi.CPQ.CreateOrderResponseDO result
117 = Apttus_CPQApi.CPQWebService.createOrder(request);

```

Code Sample

This sample uses an explicit product definition for order items while creating partial order from quote/agreement.

```

1 Apttus_CPQApi.CPQ.CreateOrderRequestDO request
2 = new Apttus_CPQApi.CPQ.CreateOrderRequestDO();
3
4 // order item source
5
6 request.OrderItemSource = Apttus_CPQApi.CPQ.ITEMSOURCE_QUOTE_LINEITEM;
7
8 // create order input
9
10 Apttus_Config2__Order__c inputSO = new Apttus_Config2__Order__c();
11
12 inputSO.Apttus_Config2__SoldToAccountId__c = '0014C00000F9Zz3';
13
14 inputSO.Apttus_Config2__PriceListId__c = 'a174C0000001MQ1';
15
16 inputSO.Apttus_Config2__PricingDate__c = Datetime.now();
17
18 inputSO.Apttus_Config2__OrderDate__c = Datetime.now();
19
20 inputSO.Apttus_Config2__OrderStartDate__c = Date.today();
21
22 inputSO.Apttus_Config2__OrderEndDate__c = Date.today().addMonths(12);
23
24

```

```
25 // add custom fields to orderSO (optional)
26
27 inputSO.put('Apttus_QPConfig__ProposalId__c','a0Y4C0000021Cm5');
28
29 request.OrderInput = inputSO;
30
31 request.CustomFields.add('Apttus_QPConfig__ProposalId__c');
32
33 // add order items (optional)
34
35
36 // selected item
37
38
39 Apttus_CPQApi.CPQ.SelectedProductDO productDO
40 = new Apttus_CPQApi.CPQ.SelectedProductDO();
41
42 productDO.Comments = 'Test Create Order API';
43
44 productDO.CopyBundleConfigurationFromSource = false;
45
46 productDO.StartDate = Date.today();
47
48 productDO.EndDate = Date.today().addMonths(12);
49
50 productDO.SellingTerm = 12;
51
52 productDO.Quantity = 1;
53
54 //productDO.ProductId = '01t4C000001pB1J';
55
56
57 // add attributes to the selected product (optional)
58
59
60 Apttus_Config2__ProductAttributeValue__c
61 attVal = new Apttus_Config2__ProductAttributeValue__c();
62
63 attVal.APTS_TestAttColor__c = 'Black';
64
65 productDO.AttributeValues.add(attVal);
66
67
```

```
68 // add related line items to selected product (optional)
69
70
71 Apttus_Config2__RelatedLineItem__c
72 lineItem = new Apttus_Config2__RelatedLineItem__c();
73
74
75 lineItem.Apttus_Config2__RelationType__c = 'Wallet';
76
77
78 productD0.RelatedLineItems.add(lineItem);
79
80
81 // source id and the explicit product definition are mutually exclusive
82
83
84 productD0.SourceId = 'a0W4C000001MRRF';
85
86
87 request.OrderItems.add(productD0);
88
89
90 // add order properties to use contract pricing (optional)
91
92
93 // order properties (optional)
94
95
96 // request.Properties.add(new
97 Apttus_Config2.Property('useAdvancedApproval' 'false'));
98
99
100 // request.Properties.add(new Apttus_Config2.Property('useDealOptimizer'
101 'false'));
102
103
104 // request.Properties.add(new Apttus_Config2.Property('retId',
105 'a0Y4C0000021Cm5'));
106
107
108 request.Properties.add(new Apttus_Config2.Property('cntrNbr_1',
109 'Q-00004442'));
110
111 // Q-00004442 is the contract number in this example (Contract Number
```

```

112     field on PriceList object)
113
114
115     // create order api call
116
117
118     Apttus_CPQApi.CPQ.CreateOrderResponseDO result
119     = Apttus_CPQApi.CPQWebService.createOrder(request);
    
```

Creating a Direct Order

This API creates a direct order without source Id.

API Details

| API | | Signature | |
|---|---|--|----------------------------------|
| CreateOrderRequest | | <i>WebService static CPQApi.CPQ.CreateOrderResponseDO CreateOrderRequest(CPQApi.CPQ.CreateOrderRequestDO)</i> | |
| Parameters | | | |
| Name | Type | Description | |
| request | CPQApi.CPQ.CreateOrderRequestDO | The request data object. | |
| Request Data Object - CPQApi.CPQ.CreateOrderRequestDO | | | |
| Field | Type | Required? | Description |
| CustomFields | List<String> | No | List of custom fields on orderSO |
| OrderInput | Conga_Config2__Order__c | Yes | OrderSO |
| OrderItems | List<Apttus_CPQApi.CPQ.SelectedProductDO> | No | List of order items |

| Request Data Object - CPQApi.CPQ.CreateOrderRequestDO | | | |
|---|--|-----------|---|
| Field | Type | Required? | Description |
| OrderItemSource | String | Yes | Source of order items |
| Properties | List<Apttus_Config2.Property> | No | List of properties for cart created for order |
| Request Data Object - CPQApi.CPQ.SelectedProductDO | | | |
| Field | Type | Required? | Description |
| AttributeValues | List<Apttus_Config2__ProductAttributeValue__c> | No | Product attribute value for the product |
| Comments | String | No | Comments |
| CopyBundleConfigurationFromSource | Boolean | No | For bundles, to copy bundle configuration |
| CustomData | Apttus_Config2__LineItem__c | No | Custom data line item |
| CustomFields | List<String> | No | List of custom fields |
| EndDate | Date | No | End date |
| ProductId | Id | No | Product Id |
| Quantity | Decimal | No | Quantity |
| RelatedLineItems | List<Apttus_Config2__RelatedLineItem__c> | No | List of related line items |
| SellingTerm | Decimal | No | Selling term |
| SourceFields | List<String> | No | List of source custom fields to be copied |
| SourceId | Id | No | Id of a source item |

| Request Data Object - CPQApi.CPQ.SelectedProductDO | | | |
|--|------|-----------|-------------|
| Field | Type | Required? | Description |
| StartDate | Date | No | Start Date |

Code Sample

```

1  Apttus_CPQApi.CPQ.CreateOrderRequestDO
2  request = new Apttus_CPQApi.CPQ.CreateOrderRequestDO();
3
4
5  // order item source
6
7  request.OrderItemSource = Apttus_CPQApi.CPQ.ITEMSOURCE_QUOTE_LINEITEM;
8
9
10 // create order input
11
12 Apttus_Config2_Orderc
13 inputS0 = new Apttus_Config2Order_c();
14
15
16 inputS0.Apttus_Config2_SoldToAccountId_c
17 = '0014C00000F9Zz3';
18
19 inputS0.Apttus_Config2_PriceListId_c
20 = 'a174C0000001MQ1';
21
22 inputS0.Apttus_Config2_PricingDate_c
23 = DateTime.now();
24
25 inputS0.Apttus_Config2_OrderDate_c
26 = DateTime.now();
27
28 inputS0.Apttus_Config2_OrderStartDate_c
29 = Date.today();
30
31 inputS0.Apttus_Config2_OrderEndDate_c
32 = Date.today().addMonths(12);
33

```

```
34
35 inputSO.put('Apttus_QPConfig_ProposalId_c','a0Y4C0000021Cm5');
36
37 request.OrderInput = inputSO;
38
39 request.CustomFields.add('Apttus_QPConfig_ProposalId_c');
40
41
42 // add order items (optional)
43
44 // selected item
45
46 Apttus_CPQApi.CPQ.SelectedProductDO productDO = new
47 Apttus_CPQApi.CPQ.SelectedProductDO();
48
49 productDO.Comments = 'Test Create Order API';
50
51 productDO.CopyBundleConfigurationFromSource = false;
52
53 productDO.StartDate = Date.today();
54
55 productDO.EndDate = Date.today().addMonths(12);
56
57 productDO.SellingTerm = 12;
58
59 productDO.Quantity = 1;
60
61 productDO.ProductId = '01t4C000001pB1J';
62
63
64 Apttus_Config2_ProductAttributeValuec
65 attVal = new Apttus_Config2ProductAttributeValue_c();
66
67 attVal.RY_Color__c = 'Black';
68
69
70 productDO.AttributeValues.add(attVal);
71
72 // productDO.SourceId = 'a0W4C000001MRRF';
73
74 request.OrderItems.add(productDO);
75
76
77 // create order
78
```

```

79 Apttus_CPQApi.CPQ.CreateOrderResponseDO result =
80 Apttus_CPQApi.CPQWebService.createOrder(request);

```

Executing Order Workflow

This API executes an Order Workflow Ruleset (Composite API).

API Details

| API | | Signature | |
|---|---|---|--|
| execOrderWorkflow | | <i>static</i> Apttus_Config2.CPQStruct.ExecOrderWorkflowResponseDO execOrderWorkflow(Apttus_Config2.CPQStruct.ExecOrderWorkflowRequestDO) | |
| Parameters | | | |
| Name | Type | Description | |
| request | Apttus_Config2.CPQStruct.ExecOrderWorkflowRequestDO | The request data object. | |
| Request Data Object - Apttus_Config2.CPQStruct.ExecOrderWorkflowRequestDO | | | |
| Field | Type | Required? | Description |
| OrderId | ID | Yes | Reference to the order. Order ID can be any one of the following: <ul style="list-style-type: none"> Order in "Pending, In Fulfillment, Partially Fulfilled" status that requires amendment Order in "In Amendment" status for which additional line item changes are required |

API Response

| Response Data Object - Apttus_Config2.CPQStruct.ExecOrderWorkflowResponseDO | | |
|---|---------|--|
| Field | Type | Description |
| IsSuccess | Boolean | Workflow is a success true/false |
| JobId | ID | Id of asynchronous apex thread. The status of this job can be tracked as a Salesforce Apex Job |

Code Sample

The following sample enables you to execute an order workflow.

```

1  // id of an order record where status is Pending or In Fulfillment or
2  // Partially Fulfilled
3  Id orderId = 'a2n2f0000008Swz';
4
5  // request
6  Apttus_Config2.CPQStruct.ExecOrderWorkflowRequestDO request = new
7  Apttus_Config2.CPQStruct.ExecOrderWorkflowRequestDO();
8  request.OrderId = orderId;
9
10 // response
11
12 response =
13 Apttus_Config2.OrderWebService.execOrderWorkflow(request);
14 system.debug('Success = '+response.IsSuccess);
15 system.debug('Async Job Id = '+response.JobId);

```

Splitting an Order using API

This API divides an existing order (parent order) into multiple orders (child orders) during the provisioning process. You can split any large order that was created from a Quote, Agreement, or Direct order that has not been amended.

API Details

| API | Signature |
|------------|---|
| SplitOrder | <i>WebService static</i> <i>Apttus_Config2.CPQStruct.SplitOrderResponseDO</i> <i>splitOrder(Apttus_Config2.CPQStruct.SplitOrderRequestDO)</i> |

| Parameters | | |
|------------|-------------------------------|--------------------------|
| Name | Type | Description |
| request | CPQStruct.SplitOrderRequestDO | The request data object. |

| Request Data Object - CPQStruct.SplitOrderRequestDO | | | |
|---|------|-----------|---|
| Field | Type | Required? | Description |
| OrderId | ID | Yes | The parent order ID to create a split order. |
| OrderLineNumbers | ID | Yes | order line numbers to be part of new split order. |

API Response

| Response Data Object - CPQStruct.SplitOrderResponseDO | | |
|---|------|--|
| Field | Type | Description |
| SplitOrderSO | ID | The Id of the newly created split order. |

Code Sample

The following code sample enables you to split an existing valid order into multiple orders using OrderId and OrderLineNumbers.

Sample snippet:

```

1  // split the order
2      Apttus_Config2.CPQStruct.SplitOrderRequestDO request = new
      Apttus_Config2.CPQStruct.SplitOrderRequestDO();
3      request.OrderId = 'a2n7A0000002jnwQAQ'; // order id to create split
      order from
4
5      request.OrderLineNumbers = new List<Integer>{1}; //add order line
      numbers to be part of new split order
6
7      Apttus_Config2.CPQStruct.SplitOrderResponseDO result =
      Apttus_Config2.OrderWebService.splitOrder(request);
8
9      Apttus_Config2__Order__c splitOrderSO = result.SplitOrderSO;

```

Integration Details

Use the following information in your integrations with Conga CPQ Web Services API. Refer to [Integrating conga with External Systems](#) for information on how to get started.

Request and Response XML

Example Request

```

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:ord="
http://soap.sforce.com/schemas/class/Apttus_Config2/OrderWebService" xmlns:cpq="http:
//soap.sforce.com/schemas/class/Apttus_Config2/CPQStruct">
  <soapenv:Header>
    <ord:SessionHeader>
      <ord:sessionId>00D7A0000009QII!
ARUAQFmpeB0msYS2JcADxYc4PxWo0kAHJP1JiNo5vtvnDBCEvfnWWm07pqbEXY7WadiCmm_Vib6MjhmGUbR8
321tI01NQRg</ord:sessionId>
    </ord:SessionHeader>
  </soapenv:Header>

```

```

<soapenv:Body>
  <ord:splitOrder>
    <ord:request>
      <cpq:OrderId>a2n7A0000002tviQAA</cpq:OrderId>
      <cpq:OrderLineNumbers>2</cpq:OrderLineNumbers>
    </ord:request>
  </ord:splitOrder>
</soapenv:Body>
</soapenv:Envelope>

```

Example Response

```

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns="http://soap.sforce.com/schemas/class/Apttus_Config2/OrderWebService"
  xmlns:SplitOrderResponseD0="http://soap.sforce.com/schemas/class/Apttus_Config2/CPQStruct" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <splitOrderResponse>
      <result>
        <SplitOrderResponseD0:SplitOrderS0 xsi:type="Apttus_Config2__Order__c">
          <Id>a2n7A0000002tvxQAA</Id>
          <Apttus_Config2__AutoActivateOrder__c>>false</
Apttus_Config2__AutoActivateOrder__c>
          <Apttus_Config2__BillToAccountId__c>0017A00000WW0pZQAT</
Apttus_Config2__BillToAccountId__c>
          <Apttus_Config2__BillingPreferenceId__c>a2d2f0000000h0YAAQ</
Apttus_Config2__BillingPreferenceId__c>
          <Apttus_Config2__ConfigurationSyncDate__c>2020-07-06T08:39:40.000Z</
Apttus_Config2__ConfigurationSyncDate__c>
          <Apttus_Config2__IsTaskPending__c>>false</
Apttus_Config2__IsTaskPending__c>
          <Apttus_Config2__OrderAmount__c>500.00000</
Apttus_Config2__OrderAmount__c>
          <Apttus_Config2__OrderDate__c>2020-07-06T06:33:53.000Z</
Apttus_Config2__OrderDate__c>
          <Apttus_Config2__OriginalOrderNumber__c>0-00005958</
Apttus_Config2__OriginalOrderNumber__c>
          <Apttus_Config2__ParentOrderId__c>a2n7A0000002tviQAA</
Apttus_Config2__ParentOrderId__c>
          <Apttus_Config2__PriceListId__c>a173C000000fHFQQA2</
Apttus_Config2__PriceListId__c>

```

```

        <Apttus_Config2__PricingDate__c>2020-07-06T06:33:53.000Z</
Apttus_Config2__PricingDate__c>
        <Apttus_Config2__ReadyForWorkflow__c>>false</
Apttus_Config2__ReadyForWorkflow__c>
        <Apttus_Config2__ShipToAccountId__c>0017A00000WW0pZQAT</
Apttus_Config2__ShipToAccountId__c>
        <Apttus_Config2__SoldToAccountId__c>0017A00000WW0pZQAT</
Apttus_Config2__SoldToAccountId__c>
        <Apttus_Config2__SourceChannel__c>Direct</
Apttus_Config2__SourceChannel__c>
        <Apttus_Config2__Source__c>Account</Apttus_Config2__Source__c>
        <Apttus_Config2__Status__c>Draft</Apttus_Config2__Status__c>
        <Apttus_Config2__Type__c>New</Apttus_Config2__Type__c>
        <CurrencyIsoCode>USD</CurrencyIsoCode>
    </SplitOrderResponseD0:SplitOrderS0>
</result>
</splitOrderResponse>
</soapenv:Body>
</soapenv:Envelope>

```

Splitting an Order by criteria

This API divides an existing order (parent order) into multiple orders (child orders) based on the defined criteria. Criteria should be added while calling the API. You can split any large order that was created from a Quote, Agreement, or Direct order that has not been amended.

API Details

| API | | Signature |
|------------|-------------------------------|--|
| splitOrder | | <pre> webService static Conga_Config2.CPQStruct.SplitOrderResponseDO splitOrder(Apttus_Config2.CPQStruct.SplitOrderRequestDO) </pre> |
| Parameters | | |
| Name | Type | Description |
| request | CPQStruct.SplitOrderRequestDO | The request data object. |

| Request Data Object - CPQStruct.SplitOrderRequestDO | | | |
|---|------|-----------|--|
| Field | Type | Required? | Description |
| OrderId | ID | Yes | The parent order ID to create a split order. |
| ProductID | ID | Yes | The product ID added in the parent order. |

API Response

| Response Data Object - CPQStruct.SplitOrderResponseDO | | |
|---|------|--|
| Field | Type | Description |
| SplitOrderSO | ID | The Id of the newly created split order. |

Code Sample

The following code sample enables you to split an existing valid order into multiple orders using OrderId and OrderLineNumbers.

Sample snippet:

```

1      Apttus_Config2.CPQStruct.SplitOrderRequestDO request = new
      Apttus_Config2.CPQStruct.SplitOrderRequestDO();
2      request.OrderId = 'a2n7A0000002mSo';
3      //split based on product
4      request.SplitCriteriaFields.add('Apttus_Config2__ProductId__c');
5      Apttus_Config2.CPQStruct.SplitOrderResponseDO result =
      Apttus_Config2.OrderWebService.splitOrder(request);

```

Integration Details

Use the following information in your integrations with Conga CPQ Web Services API. Refer to [Integrating Conga with External Systems](#) for information on how to get started.

Request and Response XML

Example Request

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:ord="
http://soap.sforce.com/schemas/class/Apttus_Config2/OrderWebService" xmlns:cpq="http:
//soap.sforce.com/schemas/class/Apttus_Config2/CPQStruct">
  <soapenv:Header>
    <ord:SessionHeader>
      <ord:sessionId>00D7A0000009QII!
ARUAQHAmHUdKQgPNeSQPZdo719RxuQc_XBiAYgTnBaswMACJu1z.5fLLZYp0P6kNf1jKKe4tVrdBb7Hn09Nqe
PMYF.s.rdq8</ord:sessionId>
    </ord:SessionHeader>
  </soapenv:Header>
  <soapenv:Body>
    <ord:splitOrder>
      <ord:request>
        <cpq:OrderId>a2n7A0000002tvs</cpq:OrderId>
        <cpq:SplitCriteriaFields>product</cpq:SplitCriteriaFields>
      </ord:request>
    </ord:splitOrder>
  </soapenv:Body>
</soapenv:Envelope>
```

Example Response

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns="ht
tp://soap.sforce.com/schemas/class/Apttus_Config2/OrderWebService"
xmlns:SplitOrderResponseD0="http://soap.sforce.com/schemas/class/Apttus_Config2/
CPQStruct" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <splitOrderResponse>
      <result>
        <SplitOrderResponseD0:SplitOrderS0 xsi:nil="true"/>
      </result>
    </splitOrderResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

```

        </splitOrderResponse>
    </soapenv:Body>
</soapenv:Envelope>
    
```

Splitting an Order by threshold

This API divides an existing order (parent order) into multiple orders (child orders) based on the threshold defined in the *Order System Properties*.

Important

The split threshold can not be defined independently. Criteria should be added while calling the API. Apttus Order Management recommends you select the common fields like a price list id.

API Details

| API | Signature |
|------------|---|
| splitOrder | <pre> webService static Apttus_Config2.CPQStruct.SplitOrderResponseDO splitOrder(Apttus_Config2.CPQStruct.SplitOrderRequestDO) </pre> |

| Parameters | | |
|------------|-------------------------------|--------------------------|
| Name | Type | Description |
| request | CPQStruct.SplitOrderRequestDO | The request data object. |

| Request Data Object - CPQStruct.SplitOrderRequestDO | | | |
|---|------|-----------|---|
| Field | Type | Required? | Description |
| OrderId | ID | Yes | The parent order ID to create a split order. |
| OrderLineNumbers | ID | Yes | order line numbers to be part of a new split order. |

API Response

| Response Data Object - CPQStruct.SplitOrderResponseDO | | |
|---|------|--|
| Field | Type | Description |
| SplitOrderSO | ID | The Id of the newly created split order. |

Code Sample

The following code sample enables you to split an existing valid order into multiple orders using OrderId and OrderLineNumbers.

Sample snippet:

```

1  Apttus_Config2.CPQStruct.SplitOrderRequestDO request = new
    Apttus_Config2.CPQStruct.SplitOrderRequestDO();
2      request.OrderId = 'a2n7A0000002tRi';
3  //Split based on threshold which is defined in the order system
    properties.Split Order Threshold
4  //Must add a common criterion like price list id while splitting with a
    threshold
5      request.SplitCriteriaFields.add('Apttus_Config2__PriceListId__c');
6      Apttus_Config2.CPQStruct.SplitOrderResponseDO result =
    Apttus_Config2.OrderWebService.splitOrder(request);

```

Integration Details

Use the following information in your integrations with Conga CPQ Web Services API. Refer to [Integrating Conga with External Systems](#) for information on how to get started.

Request and Response XML

Example Request

```

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:ord="
http://soap.sforce.com/schemas/class/Apttus_Config2/OrderWebService" xmlns:cpq="http:
//soap.sforce.com/schemas/class/Apttus_Config2/CPQStruct">
  <soapenv:Header>
    <ord:SessionHeader>
      <ord:sessionId>00D7A0000009QII!
ARUAQHAmHuDkQgPNeSQPZdo719RxuQc_XBiAYgTnBaswMACJu1z.5fLLZYp0P6kNf1jKKe4tVrdBb7Hn09Nqe
PMYF.s.rdq8</ord:sessionId>
    </ord:SessionHeader>
  </soapenv:Header>
  <soapenv:Body>
    <ord:splitOrder>
      <ord:request>
        <cpq:OrderId>a2n7A0000002tvs</cpq:OrderId>
        <!--Adding price list ID-->
        <cpq:SplitCriteriaFields>a173C000000fHFQ</cpq:SplitCriteriaFields>
      </ord:request>
    </ord:splitOrder>
  </soapenv:Body>
</soapenv:Envelope>

```

Example Response

```

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns="ht
tp://soap.sforce.com/schemas/class/Apttus_Config2/OrderWebService"
xmlns:SplitOrderResponseD0="http://soap.sforce.com/schemas/class/Apttus_Config2/
CPQStruct" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <splitOrderResponse>
      <result>
        <SplitOrderResponseD0:SplitOrderS0 xsi:nil="true"/>
      </result>
    </splitOrderResponse>
  </soapenv:Body>
</soapenv:Envelope>

```

Use Case

Use Case - A Manufacturing company quotes 450 hardware items on quote Q1. An order O1 has been generated with 450 line items. The back-office ERP system leveraged for the fulfillment of hardware items can only process 100 lines in a given order. Hence, the order needs a split.

The **splitOrder** API allows the users to split an order based on the threshold defined in the Order System Properties.

The Split result- The split operation creates five new orders where four orders with 100 lines and one with 50 lines. The original order will be empty. However, there is no impact on the configuration defined in the quote. on the quote is not impacted.

Customer Purchase Order (CPO) Web Service

You can invoke APIs in Order Web Service (Apttus_Config2) from the following command.

```
Apttus_Config2.CPQStruct.<Name of the Function>
where the name of the function is API Name and its parameters.
```

Use CPO Web Service APIs to complete the following tasks:

- [Accepting Customer Purchase Orders](#)
- [Amending Customer Purchase Orders](#)
- [Reverting Amended Customer Purchase Orders](#)
- [Cancelling Customer Purchase Orders](#)
- [Reverting Cancelled Customer Purchase Orders](#)
- [Creating Orders from PO](#)

i While capturing a Customer Purchase Order for Purchase Order and Purchase Order Items, refer to standard Salesforce SOQL APIs.

Accepting Customer Purchase Orders

This API accepts a Customer Purchase Order in **Draft** status. Changes the status of the Customer Purchase Order and Purchase Order Items to **Accepted**.

API Details

| API | | Signature |
|-------------|---|---|
| acceptOrder | | <i>webservice static</i> <i>Apttus_Config2.CPQStruct.AcceptCustomerPOResponseDO</i> <i>acceptOrder(Apttus_Config2.CPQStruct.AcceptCustomerPORestDO)</i> |
| Parameters | | |
| Name | Type | Description |
| request | Apttus_Config2.CPQStruct.AcceptCustomerPORestDO | The request data object. |

| Request Data Object - Apttus_Config2.CPQStruct.AcceptCustomerPORestDO | | | |
|--|------|-----------|--------------------------------------|
| Field | Type | Required? | Description |
| CustomerPOId | ID | Yes | The Id of the Order to be finalized. |

API Response

| Response Data Object - CPQStruct.AcceptCustomerPOResponseDO | | |
|---|-----------------|---------------------------|
| Field | Type | Description |
| CustomerPurchaseOrderSO | Standard Object | The response data object. |

| Response Data Object - CustomerPurchaseOrderSO | | |
|--|---------|--|
| Field | Type | Description |
| IsSuccess | boolean | Specifies whether the order is finalized successfully. |

Code Sample

The following enables you to accept an order for a valid customer purchase order with an Order ID.

```

1  // accept the order
2  CPQStruct.AcceptCustomerPORequestDO request = new
    CPQStruct.AcceptCustomerPORequestDO();
3  request.CustomerPOId = customerPO.Id;
4
5  CPQStruct.AcceptCustomerPOResponseDO result =
    CustomerPOWebService.acceptOrder(request);
6  system.debug(result.CustomerPurchaseOrderSO)

```

Amending Customer Purchase Orders

This API modify or amend a Customer Purchase Orders without changing the value of the order line items (Composite API).

API Details

| API | | Signature | |
|---|---|---|---|
| amendCustomerPO | | <i>webService</i> <i>static</i> <i>Apttus_Config2.CPQStruct.AmendCustomerPOResponseDOamendCustomerPO(Apttus_Config2.CPQStruct.AmendCustomerPORequestDO)</i> | |
| Parameters | | | |
| Name | Type | Description | |
| request | Apttus_Config2.CPQStruct.AmendCustomerPORequestDO | The request data object. | |
| Request Data Object - Apttus_Config2.CPQStruct.AmendCustomerPORequestDO | | | |
| Field | Type | Required? | Description |
| CustomerPOId | ID | Yes | Reference to the order. Order Id can be any one of the following: <ul style="list-style-type: none"> Order in "Accept, Partially Processed" status that requires amendment Order in "In Amendment" status for which additional line item changes are required |
| CustomerPOItems | List<CustomerPOItem_C> | No | CPO Items to be amended |

API Response

| Response Data Object - CPQStruct.AmendCustomerPOResponseDO | | |
|--|----------------------|-------------------------------------|
| Field | Type | Description |
| AmendCustomerPO | Standard Data Object | The fields of the AmendOrder Object |

| Response Data Object - CPQStruct.AmendCustomerPOResponseDO | | |
|--|------|-----------------------------|
| Field | Type | Description |
| AmendedCustomerPOItems | List | List of amended order items |

Code Sample

The following sample enables you to amend an order for a valid order with an Order Id.

```

1 // create the amend request CPQStruct.AmendCustomerPORequestDO request =
2   new CPQStruct.AmendCustomerPORequestDO();
3 request.customerPOId = customerPO.ID;
4 request.CustomerPOItems = CustomerPOItems;
5 CPQStruct.AmendCustomerPOResponseDO result =
6   CustomerPOWebService.amendCustomerPO(request);
7 system.debug(result.AmendCustomerPOSO);
8 system.debug(result.AmendedCustomerPOItems);

```

Reverting Amended Customer Purchase Orders

This API undo or revert an amended Customer Purchase Order (Composite API). For example, You can roll back the changed order date to the previous date using this API.

API Details

| API | Signature |
|---------------------|--|
| undoAmendCustomerPO | <pre> <i>webService</i> <i>static</i> <i>Apttus_Config2.CPQStruct.UndoAmendCustomerPOResponseDO</i>undoAmendCustomerPO(<i>Apttus_Config2.CPQStruct.UndoAmendCustomerPORequestDO</i>) </pre> |

| Parameters | | |
|------------|---|--------------------------|
| Name | Type | Description |
| request | Apttus_Config2.CPQStruct.UndoAmendCustomerPORequestDO | The request data object. |

| Request Data Object - Apttus_Config2.CPQStruct.UndoAmendCustomerPORequestDO | | | |
|---|------|-----------|---|
| Field | Type | Required? | Description |
| CustomerPOId | ID | Yes | Unique identifier of the amended order. |

API Response

| Response Data Object - CPQStruct.UndoAmendCustomerPOResponseDO | | |
|--|----------------------|--|
| Field | Type | Description |
| PreviousCustomerPurchaseOrderSO | Standard Data Object | The fields of the Previous Order Object. |

Code Sample

The following sample allows you to roll back an amendment to an order.

| | |
|---|---|
| 1 | <code>//undoamend customerPurchaseOrder</code> |
| 2 | <code>CPQStruct.UndoAmendCustomerPORequestDO request = new</code> |
| | <code>CPQStruct.UndoAmendCustomerPORequestDO();</code> |
| 3 | <code>request.CustomerPOId = CustomerPOId;</code> |
| 4 | |
| 5 | <code>CPQStruct.UndoAmendCustomerPOResponseDO result =</code> |
| | <code>CustomerPOWebService.undoAmendCustomerPO(request);</code> |


```
6 system.debug(result.PreviousCustomerPurchaseOrderSO)
```

Cancelling Customer Purchase Orders

This API cancels a Customer Purchase Order.

API Details

| API | | Signature | |
|---|---|---|---|
| cancelCustomerPO | | <i>WebService static</i> <i>Apttus_Config2.CPQStruct.CancelCustomerPOResponseDOcancelCustomerPO(Apttus_Config2.CPQStruct.CancelCustomerPOResponseDO)</i> | |
| Parameters | | | |
| Name | Type | Description | |
| request | Apttus_Config2.CPQStruct.CancelCustomerPOResponseDO | The request data object. | |
| Request Data Object - Apttus_Config2.CPQStruct.CancelCustomerPOResponseDO | | | |
| Field | Type | Required? | Description |
| CustomerPOId | ID | Yes | Unique identifier of the latest version of the customer purchase order to be cancelled. |

API Response

| Response Data Object - CPQStruct.CancelCustomerPOResponseDO | | |
|---|----------------------|--|
| Field | Type | Description |
| CancelCustomerPOSO | Standard Data Object | The fields of the Cancel Order Object. |

Code Sample

The following sample allows you to cancel a customer purchase order.

```

1 // Create the Cancel Request Data Object
2 CPQStruct.CancelCustomerPORequestDO request = new
   CPQStruct.CancelCustomerPORequestDO();
3 request.CustomerPOId = customerPO.Id;
4
5 CPQStruct.CancelCustomerPOResponseDO result = new
   CPQStruct.CancelCustomerPOResponseDO();
6 result = CustomerPOWebservice.cancelCustomerPO(request);
7 // returns next version of customerpurchaseOrder
8 system.debug(result.CancelCustomerPOS0);

```

Reverting Cancelled Customer Purchase Orders

This API reverts the cancellation of a customer purchase order.

API Details

| API | Signature |
|----------------------|--|
| undoCancelCustomerPO | <i>webservice static</i> <i>Apttus_Config2.CPQStruct.UndoCancelCustomerPOResponseDO</i> <i>undoCancelCustomerPO(Apttus_Config2.CPQStruct.UndoCancelCustomerPORequestDO)</i> |

| Parameters | | |
|------------|--|--------------------------|
| Name | Type | Description |
| request | Apttus_Config2.CPQStruct.UndoCancelCustomerPORequestDO | The request data object. |

| Request Data Object - Apttus_Config2.UndoCancelCustomerPORequestDO | | | |
|--|------|-----------|---|
| Field | Type | Required? | Description |
| CustomerPOId | ID | Yes | Unique identifier of the latest version of cancelled customer purchase order. |

API Response

| Response Data Object - CPQStruct.UndoCancelOrderResponseDO | | |
|--|----------------------|--|
| Field | Type | Description |
| PreviousCustomerPurchaseOrderSO | Standard Data Object | The fields of the previous customer purchase order object. |

Code Sample

The following sample allows you to roll back cancellation of a customer purchase order.

```
//undocancel customerPurchaseOrder
CPQStruct.UndoCancelCustomerPORequestDO request = new
    CPQStruct.UndoCancelCustomerPORequestDO();
request.CustomerPOId = CustomerPOId;

CPQStruct.UndoCancelCustomerPOResponseDO result =
    CustomerPOWebService.undoCancelCustomerPO(request);
system.debug(result.PreviousCustomerPurchaseOrderSO);
```

Creating Orders from PO

This API creates a sales order from a Customer Purchase Order: single order, multiple order using split criteria, or partial order.

i You must enable the Split Group Properties field on PO item in order to consume this functionality. Refer to the *Order Management on Salesforce Administrator Guide* for steps.

API Details

| API | | Signature |
|--|--|---|
| createOrderForPO | | <pre>webService static Apttus_CPQApi.CPQ.CreateOrderForPOResponseDO createOrderForPO(Apttus_CPQApi.CPQ.CreateOrderForPORequestDO)</pre> |
| Parameter 1: Create Order for Customer Purchase Order Data Object | | |
| Name | Type | Description |
| request | Apttus_CPQApi.CPQ.CreateOrderForPOResponseDO | The request data object. |
| Request Data Object - CreateOrderForPOResponseDO | | |
| Field | Type | Description |
| orderInputSO | Standard Object | To create the order inputs. |
| PurchaseOrderId | ID | To create a single order. |
| SplitGroupFields | List<String> | To group by orders |

| Request Standard Object - orderInputSO | | | |
|---|------|-----------|---|
| Field | Type | Required? | Description |
| SoldToAccountId | ID | Yes | The id of the Account to which the order is placed. |

| Request Standard Object - orderInputSO | | | |
|--|--|--------------------------|---|
| Field | Type | Required? | Description |
| PriceListId | ID | Yes | The id of the PriceList to be associated with the order. |
| PricingDate | Datetime | No | Date and time associated with the order and represent the date when the order will be priced. |
| OrderDate | Datetime | No | Date and time when the order is created. |
| OrderStartDate | Datetime | No | Date and time when the order fulfillment starts. |
| OrderEndDate | Datetime | No | Date and time when the order fulfillment ends. |
| useAdvancedApproval | Boolean | No | Whether to use advanced approval. The default value is False. |
| useDealOptimizer | Boolean | No | Whether to use the Deal Optimizer. The default value is False. |
| Parameter 2: Create Selected Purchase Order Item Data Object | | | |
| Name | Type | Description | |
| request | Apttus_CPQApi.CPQ.SelectedPOItemDO poltemDO | The request data object. | |
| Request Data Object - poltemDO (Required for partial order only) | | | |
| Field | Type | Required? | Description |
| POItemId | ID | Yes | Customer PO item number. |
| ProductId | ID | Yes | Customer Product item number. |
| Quantity | Integer | Yes | Quantity of the order. |
| StartDate | Datetime | No | Date and time when the PO fulfillment starts. |

| Request Data Object - poltemDO (Required for partial order only) | | | |
|--|----------|-----------|--|
| Field | Type | Required? | Description |
| EndDate | Datetime | No | Date and time when the PO fulfillment ends. |
| Selling Term | Integer | No | Duration of the customer PO. |
| Comments | Text | No | Any notes. |
| Sourceld | ID | Yes | Source ID of the order. For example, Quote, Contract, or Agreement ID. |
| Comments | Text | No | Comments about the agreement. |
| ContractNumbers | Text | No | Contract number from the agreement. |

API Response

| Response Data Object - Apttus_CPQApi.CPQ.CreateOrderForPOResponseDO | | |
|---|------|--|
| Name | Type | Description |
| OrderId | ID | Order number which is created for customer PO. |

Code Sample

```

1  String ACCOUNT_NAME = 'RY_Account_002';
2  String PRICELIST_NAME = 'RY_PriceList';
3  Integer SELLING_TERM = 12;
4  Id CUSTOMER_PO_ID = 'a440R0000002Tif';
5  String SPLIT_ORDER_FIELD = 'RY_GroupingTest__c';
6  Id PO_ITEM_ID = 'a430R00000002tC';
7  Id SOURCE_ITEM_ID = 'a0W0R000000JT9F';
8
9  Account myAccount = [SELECT Id FROM Account WHERE Name =: ACCOUNT_NAME];
10 Apttus_Config2__PriceList__c myPriceList = [SELECT Id FROM
    Apttus_Config2__PriceList__c WHERE Name =: PRICELIST_NAME];

```

```

11  Apttus_Config2__CustomerPOItem__c poItem = [SELECT Id, RY_GroupingTest__c,
Apttus_Config2__ProductId__c, Apttus_Config2__Quantity__c,
Apttus_Config2__StartDate__c, Apttus_Config2__EndDate__c FROM
Apttus_Config2__CustomerPOItem__c WHERE Apttus_Config2__PurchaseOrderId__c
=: CUSTOMER_PO_ID AND Id =: PO_ITEM_ID LIMIT 1];

12
13
14  /*
15
16  Create Order for Customer Purchase Order Data Object */
17  Apttus_CPQApi.CPQ.CreateOrderForPORequestDO request = new
    Apttus_CPQApi.CPQ.CreateOrderForPORequestDO();
18
19  // Create the Order Input
20  Apttus_Config2__Order__c orderInputSO = new Apttus_Config2__Order__c();
21
22  orderInputSO.Apttus_Config2__SoldToAccountId__c = myAccount.Id;
23  orderInputSO.Apttus_Config2__PriceListId__c = myPriceList.Id;
24  orderInputSO.Apttus_Config2__PricingDate__c = Datetime.now();
25  orderInputSO.Apttus_Config2__OrderDate__c = Datetime.now();
26  orderInputSO.Apttus_Config2__OrderStartDate__c = Date.today();
27  orderInputSO.Apttus_Config2__OrderEndDate__c =
    Date.today().addMonths(SELLING_TERM);
28  request.OrderInput = orderInputSO;
29
30
31
32  // Add Order Properties (Optional)
33  request.Properties.add(new Apttus_Config2.Property('useAdvancedApproval',
    'false'));
34  request.Properties.add(new Apttus_Config2.Property('useDealOptimizer',
    'false'));
35
36
37
38  // Assign the Customer Purchase Order (For single order)
39  request.PurchaseOrderId = CUSTOMER_PO_ID;
40
41
42
43  // Add Order Items (Optional)
44
45  // Selecting Items
46  // Create Selected Purchase Order Item Data Object
47

```

```

48  Apttus_CPQApi.CPQ.SelectedPOItemDO poItemDO = new
    Apttus_CPQApi.CPQ.SelectedPOItemDO(); (For partial order)
49
50  //WebService Apttus_Config2__LineItem__c CustomData
51  //WebService List CustomFields
52
53  poItemDO.POItemId = poItem.Id;
54  poItemDO.ProductId = poItem.Apttus_Config2__ProductId__c;
55  poItemDO.Quantity = poItem.Apttus_Config2__Quantity__c;
56  poItemDO.StartDate = poItem.Apttus_Config2__StartDate__c;
57  poItemDO.EndDate = poItem.Apttus_Config2__EndDate__c;
58  poItemDO.SellingTerm =
    Apttus_Config2.CPQWebService.computeTerm(poItemDO.StartDate,poItemDO.EndDate, 'Monthly');
59  poItemDO.Comments = 'Purchase order item test';
60
61
62
63  // Set Additional Parameters From Source Line Item
64  poItemDO.SourceId = SOURCE_ITEM_ID;
65  poItemDO.SourceFields.add('Apttus_QPConfig__Comments__c');
66  poItemDO.SourceFields.add('Apttus_QPConfig__ContractNumbers__c');
67  request.OrderItems.add(poItemDO);
68
69
70
71  // Split Order Criteria
72  //request.SplitGroupFields.add(SPLIT_ORDER_FIELD); (For group by orders)
73
74
75
76  // Create the Order(s)
77  Apttus_CPQApi.CPQ.CreateOrderForPOResponseDO result =
    Apttus_CPQApi.CPQWebService.createOrderForPO(request);
78  system.debug('Order Id = ' + result.OrderId);

```

Data Validation Service

You can invoke APIs in Data Validation Web Service (Apttus_Config2) from the following command.


```
Apttus_Config2.CPQStruct.<Name of the Function>
where the name of the function is API Name and it parameters.
```

Use Data Validation Service API to complete the following task:

- [validateAndEnrich](#)

validateAndEnrich

This API validates and enriches data for a Customer Purchase Order.

API Details

| API | | Signature | |
|---|--|--|---|
| acceptOrder | | <i>webService static Apttus_Config2.CPQStruct.DataValidateEnrichResult validateAndEnrichSO(Id bObjectId)</i> | |
| Parameters | | | |
| Name | Type | Description | |
| request | <i>Apttus_Config2.CPQStruct.DataValidateEnrichResult validateAndEnrichSO</i> | The request data object. | |
| Request Data Object - Apttus_Config2.CPQStruct.DataValidateEnrichResult validateAndEnrichSO | | | |
| Field | Type | Required? | Description |
| bObjectId | ID | Yes | Customer Purchase Order ID, Order Item ID to enrich and validate. |

API Response

| Response Data Object - DataValidateEnrichResult | | |
|---|---|--|
| Field | Type | Description |
| IsSuccess | boolean | Validation and Enrichment is a success. True- If there are no issue logs created for validation error as part of the execution. False- If there are issue logs create for validation error as part of the execution. |
| IssueLogs | List<Apttus_Config2__CustomerPOIssueLog__c> | List of Issue logs generated for a given PO Item / PO while running the validation rule and enrichment rule against it. |
| ModifiedSOObjects | Map<ID, SObject> | Objects Modified as Part of Enrichment rule. Key-Value Pair where Key is ID of modified record and Value is record SObject with modified Value. |
| NumErrorsByObject | Map<ID, Integer> | The number of Validation/Enrichment Error for each record. Key is record Id and value is the number of Error Issue logs created for that record. |
| NumWarningsByObject | Map<ID, Integer> | The number of Validation/Enrichment Warning for each record. Key is record Id and value is the number of Warning Issue logs created for that record. |

Code Sample

The following enables you to validate and enrich the data in the customer purchase order.

```

1  Id customerPOId = 'a6PW00000008wGY';
2  Apttus_Config2.CPQStruct.DataValidateEnrichResult response =
   Apttus_Config2.DataValidationService.validateAndEnrichSO(customerPOId);
3  system.debug('IsSuccess=>'+response.IsSuccess);
4  system.debug('IssueLogs=>'+response.IssueLogs);
5  system.debug('NumErrorsByObject=>'+response.NumErrorsByObject);
6  system.debug('NumWarningsByObject=>'+response.NumWarningsByObject);

```

Scenarios

This section covers the following scenarios.

- [Working with orders in Order Management.](#)

Working with orders in Order Management

Once the purchase is confirmed, you (sales and customer support representatives) can create an order as a confirmation before delivering goods and services.

To create a successful order:

1. In an Order page, create an order using [Creating an Order](#).
2. In an order page, create a cart for an order using [Creating a Cart for an Order](#).
3. Synchronize a new cart to order using [Synchronizing a Cart to an Order](#).
4. If you want to make the changes to the order, amend the order using [Amending an Order](#).
5. After all, reviewers have reviewed the order, accept the order using [Accepting an Order](#).
6. If the order is large and you want to divide the order based on different criteria, split the order using [Splitting an Order](#).

Order Management Feature by Release

Review the latest Order Management Features by Release document.

- [Features by Release](#)

Features by Release

This document contains an overview of features introduced in each major release of Conga Order Management. For more information, see [Order Management Features by Release](#).

Community & Learning Center Resources

The Conga Customer Community is your one-stop shop for success!

After registering as a new member you'll gain access to a wide variety of resources, including: further details regarding the onboarding process, access to the knowledge base, and product-specific discussion groups.

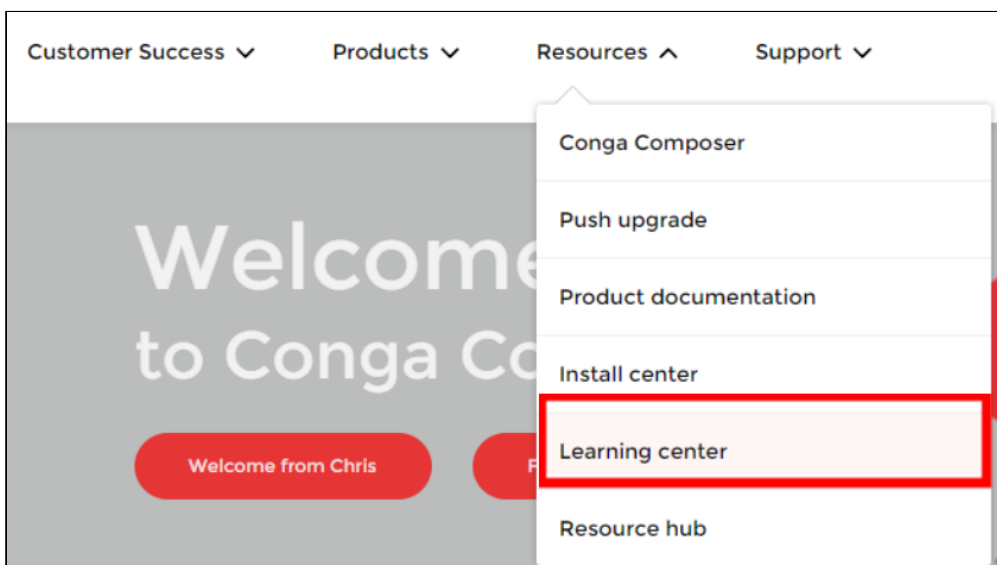
In addition, just a click away from Community is the Conga Learning Center: our comprehensive training portal well stocked with FREE self-paced courses complimented by virtual instructor-led sessions for more advanced and intensive training!

Ready to get started?

First download the [Salesforce Authenticator app](#); you'll need it as part of the login procedure.

Then create a Community account, or log in [here](#)!

Once you're logged in, navigate to the Resources dropdown menu and click Learning Center to reach the [Conga Learning Center](#)!



Conga Copyright Disclaimer

Copyright © 2022 Apttus Corporation (“Conga”) and/or its affiliates. All rights reserved.

No part of this document, or any information linked to or referenced herein, may be reproduced, distributed, or transmitted in any form or by any means, including photocopying, recording, or other electronic or mechanical methods, without the prior written consent of Conga. All information contained herein is subject to change without notice and is not warranted to be error free.

This document may describe certain features and functionality of software that Conga makes available for use under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not, in any form, or by any means, use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part of the software. Reverse engineering, disassembly, decompilation of, or the creation of derivative work(s) from, the software is strictly prohibited. Additionally, this document may contain descriptions of software modules that are optional and for which you may not have purchased a license. As a result, your specific software solution and/or implementation may differ from those described in this document.

U.S. GOVERNMENT END USERS: Conga software, including any operating system(s), integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are “commercial computer software” pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

Neither the software nor the documentation were developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Conga and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Apttus, AI Analyze, Conga, Conga AI, Conga AI Discover, Conga Batch, Conga Collaborate, Conga Composer, Conga Conductor, Conga Connect, Conga Courier, Conga Grid, Conga Mail Merge, Conga Merge, Conga Orchestrate, Conga Sign, Conga Trigger, Digital Document Transformation, True-Up, and X-Author are registered trademarks of Conga and/or its affiliates.

The documentation and/or software may provide links to web sites and access to content, products, and services from third parties. Conga is not responsible for the availability of, or any content provided by third parties. You bear all risks associated with the use of such content. If you choose to purchase any products or services from a third party, the relationship is directly between you and the third party. Conga is not responsible for: (a) the quality of third-party products or services; or (b) fulfilling any of the terms of the agreement with the third party, including delivery of products or services and warranty obligations related to purchased products or services. Conga is not responsible for any loss or damage of any sort that you may incur from dealing with any third party.

For additional resources and support, please visit <https://community.conga.com>.