



# Service and Asset Manager

**Ops Console Cloud Guide**

2020.2

## Copyright Notice

This document contains the confidential information and/or proprietary property of Ivanti, Inc. and its affiliates (referred to collectively as "Ivanti"), and may not be disclosed or copied without prior written consent of Ivanti.

Ivanti retains the right to make changes to this document or related product specifications and descriptions, at any time, without notice. Ivanti makes no warranty for the use of this document and assumes no responsibility for any errors that can appear in the document nor does it make a commitment to update the information contained herein. For the most current product information, please visit [www.ivanti.com](http://www.ivanti.com).

Copyright © 2020, Ivanti. All rights reserved.

Ivanti and its logos are registered trademarks or trademarks of Ivanti, Inc. and its affiliates in the United States and/or other countries. Other brands and names may be claimed as the property of others.

Protected by patents, see <https://www.ivanti.com/patents>.

Last updated: 7/16/2020

# Contents

---

<b>About This Guide</b>	<b>4</b>
Intended Audience	4
List of Topics	5
Related Documentation	5
How to Contact Us	6
<b>Operations Console Overview</b>	<b>7</b>
Terminology	7
<b>Before You Begin Migrating Data</b>	<b>17</b>
The Four Ways to Migrate Data	17
About Data Migration	18
Conceptual Information: Migration Process Overview	18
Items Excluded from Migration	27
<b>Troubleshooting and Known Issues</b>	<b>29</b>
Debugging the Migration	29
Known Issues in the Operations Console	31
<b>Appendix C: Using the ISM Development Project to Migrate Data</b>	<b>34</b>
Creating a Project	35
Viewing a Project	35
Creating a Transaction Set	36
Viewing Transaction Sets	36
Creating a Package	38
Viewing a Package	38
Assigning Transaction Sets to a Package	39
Adding an Aspect to a Package	40
Adding Packages to a Master Package (Mapping)	41
Closing a Package	42
About Exporting a Package	42
Importing a Package	43
Troubleshooting Package Import Errors	46
Saving a Package as a File	47
Deleting a Package	47
<b>Appendix D: Working with Packages from the App Store</b>	<b>48</b>
About the App Store	48
Accessing the App Store	48
Downloading and Importing an App Store Package	49
Troubleshooting Problems with App Store Packages	50

# About This Guide

Use the information in this guide in conjunction with the *Installation and Deployment Guide for Service and Asset Manager*. That guide contains step-by-step instructions for installing and configuring your deployment, while this guide only describes procedures related to using the Operations Console.



For a complete overview of all types of deployments and step-by-step instructions for installing and configuring each deployment, refer to the "*Installation and Deployment Guide for Service and Asset Manager Version 2020.2*". For information about using Ivanti Service and Asset Manager, including using packages and projects, see the Ivanti Service and Asset Manager online help. (See "Related Documentation" on the next page for more information about accessing the online help.)

---

This topic contains the following sections:

- "Intended Audience" below
- "List of Topics" on the next page
- "How to Contact Us" on page 6
- "Related Documentation" on the next page
- "About This Guide" above

## Intended Audience

This document is intended for system administrators, including both Ivanti personnel and members of other organizations, who will be installing and using the Operations Console to manage tenants and tenant instances and to migrate data.

This guide is aimed at the following:

- The MSP (managed service provider) system administrator for an organization that manages the Ivanti Service Manager systems of several organizations. An MSP system administrator manages the Operations Consoles and migrates data for several organizations. This customer is also called a Software partner.
- Ivanti Software employees called customer service managers who manage tenants and data migration for customers using Ivanti Service Manager Cloud.



This guide uses the term "customer." For MSP system administrators and Ivanti Software employees, the term customer refers to the actual customer (users from a different organization), but for on-premise system administrators, the term customer refers to users within the same organization.

## List of Topics

This document contains the following sections:

- "Operations Console Overview" on page 7: Describes the Operations Console. Includes a terminology list and instructions for logging in to the Operations Console.
- "Before You Begin Migrating Data" on page 17: Lists the four different ways to migrate data and contains conceptual information.
- "Troubleshooting and Known Issues" on page 29: Lists troubleshooting tips and known issues with the Operations Console.
- "Appendix C: Using the ISM Development Project to Migrate Data" on page 34: Contains the information from the Ivanti Service Manager online help that describes how to use the Configuration Console to migrate data using the Ivanti Service and Asset Manager development project.
- "Appendix D: Working with Packages from the App Store" on page 48: Describes how to migrate data that came from the App Store.

## Related Documentation

Ivanti Service and Asset Manager has online help available within the application. To access it, click the **Help** button on the top right of the application.

Additional documentation is available through

- The [Ivanti community](#) website. You may need to request user access if you cannot log in.

Or through

- The [Ivanti Product Documentation](#) website. Click the Service Manager tile to see a list of the documents available.

The online help is different depending on the role you used to log in. There are three types of online help:

- **Online help for the Self-Service Portal:** For anyone who logs in to the Self-Service Portal.
- **Online help for administrators:** For anyone who logs in as an administrator to the Configuration Console.
- **Online help for users:** For anyone who logs in to the Service Desk Console.

## How to Contact Us

To contact us about the documentation, or if you have any other questions or issues about Ivanti Service Manager, contact Ivanti Software Global Support services by logging an incident via Self Service at: <https://www.ivanti.com/support/ivanti-support>.

# Operations Console Overview

This section contains the following:

- Terminology used in this guide, including definitions of data types. See "Terminology" below.

## Terminology

This section defines key terms pertaining to the Operations Console and the tenant creation and data migration process.

- "General Terms" below
- "Types of Data" on page 9
- "Landscapes" on page 12
- "Tenant and Tenant Instance Types" on page 15

## General Terms

- "Operations Console" below
- "Customer Success Manager (CSM)" below
- "Service Level Agreement (SLA)" on the next page
- "Backup Types" on the next page

## Operations Console

The tool used to create tenants and tenant instances, move data from one tenant instance to another (for example, from the staging instance of a tenant to the production instance), and perform various administrative tasks.

## Customer Success Manager (CSM)

An Ivanti Software staff member who is the primary contact for a customer, providing support in managing the customer's Ivanti Service Manager system. A CSM establishes a long-term relationship with a customer while also supporting the Ivanti Software Professional Services Organization and other Ivanti Software teams throughout the customer's engagement with Ivanti Software.

A CSM works with a customer to do the following:

- Understand the customer's business requirements.
- Help the customer understand the roles of other Ivanti Service Manager implementation team members.

- Work with the customer to understand the customer's predefined metrics of success and ensure that those metrics are tracked and met during the customer's relationship with Ivanti Software.
- Ensure that the customer's questions and issues are acted upon quickly by the appropriate implementation team members.
- Provide the customer with information and access to training resources to ensure that the Ivanti Service Manager system is successful. Information and training resources are provided over the duration of the customer's relationship with Ivanti Software.

## Service Level Agreement (SLA)

A customer's service level agreement defines the response and resolution target times for services provided by Ivanti Software. In the context of Ivanti Service Manager, a service level agreement determines the details that govern the process to provision, configure, and upgrade the tenant and its instances.

A support service is offered through the CSM organization. Incidents and service requests are covered under the service delivery agreements, meaning that they are handled and completed within agreed time to the satisfaction of the customer within the boundaries of the contractual agreements.

## Backup Types

There are three steps to the process of backing up a database:

1. Back up the database on the source system.
2. Copy the database from the source system to the target system.
3. Restore the database on the target system.

You can either use Microsoft SQL or LiteSpeed to perform the backup. There are several backup types you can choose from when using the Operations Console:

- **From last LiteSpeed backup:** Copies the source database from the source database backup location to the target database backup location and restores the database from there.
- **From live LiteSpeed backup:** Performs a live LiteSpeed backup of the source database and places the backup in the source database backup location. Copies the source database backup to the target database backup location and restores the database from there.
- **From copied LiteSpeed backup:** The source database has already been manually copied to the target database backup location. If you choose this option, the system adds a field called **Backup Filename** where you enter the name of the LiteSpeed backup.
- **From last MSSQL backup:** Copies the source database from the source database backup location to the target database backup location and restores the database from there.



- **From live MSSQL backup:** Performs a live Microsoft SQL backup of the source database and places the backup in the source database backup location. Copies the source database backup to the target database backup location and restores the database from there.
- **From copied MSSQL backup:** The source database has already been manually copied to the target database backup location. If you choose this option, the system adds a field called **Backup Filename** where you enter the name of the Microsoft SQL backup.
- **Database already restored:** The database was already restored manually outside of the Operations Console (for example, through Microsoft SQL Server Management Studio). Choosing this option means the database is not backed up or restored.

## Types of Data

- "Configuration Data (Settings)" below
- "Master Data" on the next page
- "Metadata" on the next page
- "Transactional Data" on page 12
- "Validation Data" on page 12

## Configuration Data (Settings)

Configuration data, also called settings, is defined as core configuration attributes that define Ivanti Service Manager.

The only configuration data (settings) that the system migrates when you migrate configuration data (settings), are the following:

- Allowed attachments
- Authentication providers such as LDAP, SAML, OpenId, and Windows Integrated Security
- Bulk upload profiles
- Dynamic styles
- Global constants
- Integration settings
- LDAP configurations
- Object matching lists
- Password policies
- Voice integrations

- Watch list items
- Email configurations (including approval status keywords, email status update mapping settings, ignore message with subject settings, object mappings for email, task assignment status keywords, truncate body settings, and trusted sender domains)

You cannot migrate other types of configuration data (settings) using the Operations Console. To migrate this data, do so manually outside of the Operations Console by using Microsoft SQL commands.

## Master Data

Master data is defined as data sets that are used across an organization and are typically unique to that organization. User and role definitions, role assignments, and categories are examples of master data. Customers typically make their own changes to their master data, although the implementation team can also change master data for a customer.

Master data is migrated together with metadata so that it is consistent with the business object definitions. There are four types of master data and you can migrate any or all of them:

- User data, except for user email addresses because they may have been scrubbed.
- User-related data, such as contact group, standard user team, organizational unit, and so on.
- Service request data, such as request offerings, request parameters, and so on.
- Escalation schedule.

You can now track these types of master data:

- Analytic metrics dashboards and configuration
- Schedule entries
- Escalation settings and schedules
- Escalation clock control
- Service level packages
- Service level targets

To see a list of master data for a tenant, if any has been defined, place your mouse over the destination tenant.

## Metadata

Configured business objects, business object attributes, forms, grids, quick actions, searches, and so on. When you reconfigure your Ivanti Service Manager system, most of the changes are implemented by editing metadata.

The types of metadata are as follows:

- Business objects
- Counters
- Forms
- Grids
- Layouts
- Quick actions
- Rules
- Rule overrides
- Branding
- Roles
- Validation (pick list) definitions
- Business rules
- Searches (namely saved searches)
- Dashboards
- Chart-part metadata (one type of dashboard)
- Grid-part metadata (one type of dashboard)
- Special-part metadata (one type of dashboard)
- View-part metadata (one type of dashboard)
- Workflows
- Workflow types
- Reports (subscription, schedule, and distribution are not migrated from source to target)
- Validation data (of pure validation business objects and custom data)
- Images in the frs\_def\_images table

You can lock all of the metadata, except for quick actions, dashboards, searches, and reports. Those four types of metadata cannot be locked and are always open to be edited in the production instance of the tenant. To lock the other types of metadata, set the metadata to read-only. Each tenant instance has a **Metadata Locked** field that lets you know if the metadata is locked for an instance of a tenant.

## Transactional Data

A customer's data records, including incidents, changes, service requests, and so on.

## Validation Data

Validation data objects defined by customers. You must migrate all validation data together at the same time and you must migrate metadata and validation data together in the same migration. This guarantees that when metadata is migrated, it includes all pick list definitions that are associated with validation objects.

Validation data is any data that is configured as part of a *validation* business object. A *validation* business object and its associated data (for example, a pick list and the data in the list) are both considered validation data. But a *standard* business object and its associated data (for example, a pick list and the data in the list) are not both validation data. If the business object is a *standard* business object, then that is validation data, but the associated data is not considered validation data.

Business Object	Is it Validation Data?
Standard business object	Yes
Validation business object	Yes
Data associated with a standard business object	No
Data associated with a validation business object	Yes

## Landscapes

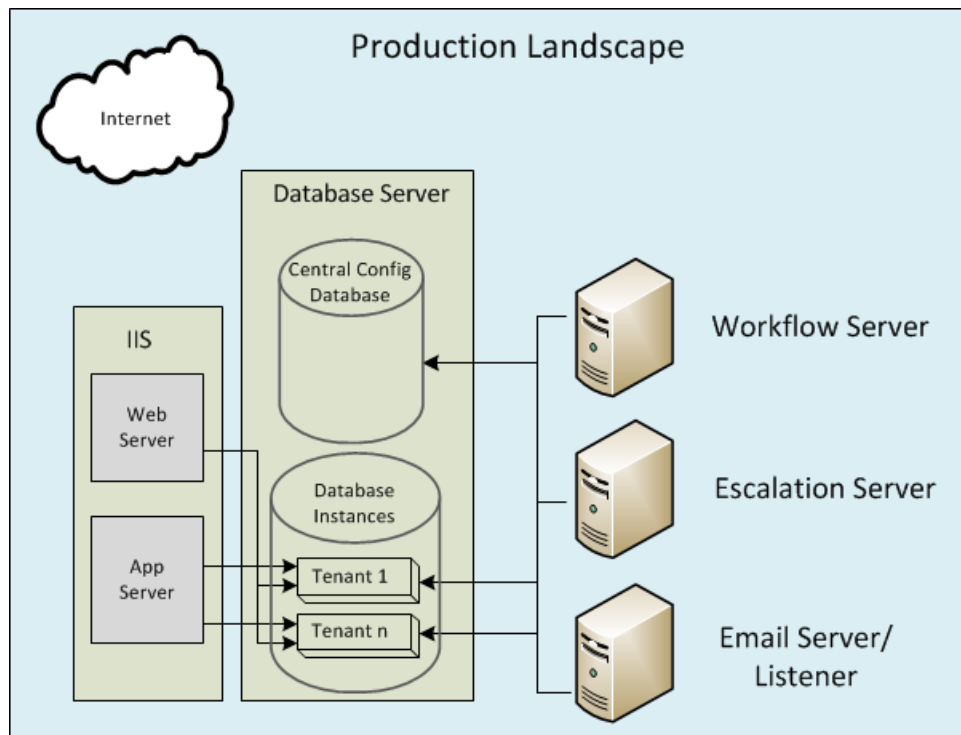
- "Landscape Definition" below
- "Landscape Components" on the next page
- "Types of Landscapes" on page 14
- "About the Pilot Landscape Lifecycle" on page 14

## Landscape Definition

A landscape is a set of IT infrastructure components that support tenant instances of the same type (where type refers to production, staging, and UAT). The production landscape only contains the production instances of tenants, the staging landscape only contains the staging instances of tenants, and the UAT landscape only contains the UAT instances of tenants.

"Typical Production Landscape" below shows a typical configuration for a production landscape. Components and their locations vary from customer to customer.

*Typical Production Landscape*



## Landscape Components

At a minimum, a landscape contains these components:

- An individual, isolated Microsoft SQL Ivanti Service Manager application database (called IvantiSM) in each tenant (each tenant instance has its own database, but not its own database instance).
- A central Ivanti Service Manager configuration database (called ConfigDB) containing information about all tenant instances in the landscape.
- A shared pool of Ivanti Service Manager application servers for all Ivanti Service Manager instances in the landscape.
- A shared database cluster for the Ivanti Service Manager configuration and Ivanti Service Manager application databases.
- A shared pool of workflow servers used by all Ivanti Service Manager instances in the landscape.
- A shared pool of escalation engines used by all Ivanti Service Manager instances in the landscape. This is sometimes combined with the workflow server.

- A shared pool of email listeners used by all Ivanti Service Manager instances in the landscape.
- A shared pool of integration servers per function, such as survey, Discovery, Desktop Server and Management (DSM), Report Server, License Server, and so on.

## Types of Landscapes

There are six types of landscapes:

- **Production landscape:** Hosts the production instances of tenants. Used during the data migration lifecycle. This is the environment where the product is deployed.
- **Staging landscape:** Hosts the staging instances of tenants. Used during the data migration lifecycle. This is the environment where development work is done.
- **UAT landscape:** Hosts the UAT instances of tenants. Used during the data migration lifecycle. This is the environment where testing is done.
- **Pilot landscape:** Optional. A standalone landscape that hosts the pilot tenant to provide customers with early access to a new Ivanti Service Manager point release. See "About the Pilot Landscape Lifecycle" below.
- **Development landscape:** Optional. A standalone landscape that hosts tenant templates for demos, test drives, training, and so on.
- **Other landscape:** Optional. A standalone landscape type that can be used for any purpose.

## About the Pilot Landscape Lifecycle

The pilot landscape hosts the pilot tenant which provides customers with early access to a new Ivanti Service Manager point release. Participating in the pilot release is optional. If you want to participate, request information from your Solutions/Implementation Manager. Ivanti Software accepts pilot customers on a first-come, first-served basis.

Approximately three weeks before a release, Ivanti Software makes the pilot landscape available to customers who have requested access. This is a temporary landscape that is only available during communicated periods.



A pilot landscape functions very similar to an alpha release. You can expect several updates and frequent changes in this landscape. We announce these frequent redeployments using system notifications only, and not through email.

Approximately a week after the Ivanti Service Manager pilot landscape release, Ivanti Service Manager moves into the staging landscape. When the system is in the staging landscape, it is available to all customers. During this time, no content moves from the staging landscape to the production landscape.

Ivanti Software posts the staging landscape update schedule on the web portal. During the staging landscape, there are approximately five updates. Each update is accompanied by a release note. You can request these staging release notes from support or your CSM.

If you find a problem, submit feedback by the end of the first week in order for it to potentially be fixed in the production landscape.

During the second week of the staging landscape, there is usually only one update and that is usually on Wednesday. This gives Ivanti Software a minimum of two days to do final testing on the system before it moves to the production landscape.

## Tenant and Tenant Instance Types

A tenant is a Ivanti Service Manager system for a particular customer or application.

An instance of the Ivanti Service Manager tenant within a landscape. By default, each customer receives a production instance of the tenant and a staging instance of the tenant. Additional optional tenant instances are also available. The name and URL for each tenant instance must be unique.



The term "tenant instance" is NOT related to the term "database instance". The use of the word instance, when referring to a tenant, simply means one copy or version of it.

The available tenant instances are:

- **Production instance of a tenant:** A tenant instance containing a customer's production database (including all configuration and transactional data), managed under a service level agreement. All customers have a production instance of their tenant. The production instance of the tenant resides in a hardened environment in the production landscape. We do not recommend making metadata or configuration changes directly to the production instance of the tenant during production usage. Instead, make those changes in the staging instance of the tenant, push them to the UAT instance of the tenant for testing, and then push them to the production instance of the tenant.

You can make some changes to master data (such as adding employees, role assignments, and so on) directly to the production instance of the tenant.

The production instance of the tenant is the only tenant instance covered under a service level agreement.

- **Staging instance of a tenant:** A tenant instance for developing and configuring a customer's Ivanti Service Manager system and business processes. This tenant instance is generally used for configuration and metadata changes (but not transactional data changes). It undergoes a baselining process, in which it is updated periodically or by customer request from the production instance of the tenant to ensure that it is in sync with the production instance of the tenant prior to configuration and metadata changes. It can also be used to test new releases, new capabilities, and for training. All customers have a staging instance of their tenant; however, the staging tenant is not covered under a service level agreement.
- **UAT instance of a tenant:** A temporary tenant instance used for configuration testing and validation before the new configuration is pushed to an existing production tenant instance. The UAT instance of the tenant is required when reconfiguring an existing production instance of the tenant. It is optional when provisioning the production instance of the tenant for the first time or when upgrading to a new Ivanti Service Manager release.
- **Pilot instance of a tenant:** During the upgrade process to a new release, an optional pilot instance of the tenant is made available to customers to allow early access to a new point release. The pilot instance of the tenant resides in the pilot landscape.
- **Development instance of a tenant:** An optional development instance of the tenant for testing or development. The development instance of the tenant resides in the development landscape.
- **Other instance of a tenant:** An optional instance of the tenant for testing or development. The other instance of the tenant resides in the other landscape.
- **Tenant environment:** Infrastructure that ties together the production, staging, and UAT instances of a tenant. This infrastructure allows a complete change management process to manage and control the development and deployment lifecycles between these instances of the tenant.
- **Tenant template:** A preconfigured ("out-of-the-box") tenant that is typically used to set up a new tenant. A template implements best practices according to ITIL standards, and optimizes processes to allow fast implementation.



## Before You Begin Migrating Data

- "The Four Ways to Migrate Data" below
- "About Data Migration" on the next page
- "Conceptual Information: Migration Process Overview" on the next page
- "Items Excluded from Migration" on page 27

### The Four Ways to Migrate Data

Before you begin migrating data, it is very important that you understand the different ways there are to migrate data. There are four ways that you can migrate data:

- Enabling the Ivanti Service Manager development project and applying a package. With a package, you can migrate a very specific set of configuration changes. For example, you can add service request data to a package and migrate it. See "Appendix C: Using the ISM Development Project to Migrate Data" on page 34.
- Enabling the Ivanti Service Manager development project and using the push link. With this method, you can migrate master data, but not configuration data (because that is moved using packages). See "Appendix C: Using the ISM Development Project to Migrate Data" on page 34.
- Disabling the Ivanti Service Manager development project and using Simple Mode. With Simple Mode, you can only copy service request data and escalation schedules. When you use Simple Mode, you copy all of the configuration data from instance of a tenant to another. See [How to Use Simple Mode to Migrate Data](#).
- Disabling the Ivanti Service Manager development project and using Advanced Mode. With Advanced Mode, you can choose which items to copy, such as all business objects but not workflows. However, you cannot specify which business objects (or whichever item you migrate). You can choose the type of items to migrate but not the specific items within that type. With Advanced Mode, you can also specify if you want to migrate master data, which you can use to copy the contents of a business object. See [Using Advanced Mode to Migrate Data](#).

We plan to remove Simple Mode and Advanced Mode in an upcoming release. The Ivanti Service Manager development project will then be the only data migration method available. This is because Simple Mode and Advanced Mode only migrate the difference in data between the source and the target, and do not take into account the sequence of the data changes between the two systems.

This document includes information about migrating data using all of the methods listed above. However, we highly recommend that you enable and use the Ivanti Service Manager development project, and therefore, this document focuses on those processes.

## About Data Migration

The process of migrating data can be defined as moving information from the production instance of the tenant to the staging instance of the tenant, and then optionally moving information from the staging instance of the tenant to the UAT instance of the tenant, and then finally moving information back to the production instance of the tenant (from either the staging or UAT instance of the tenant).

There are two scenarios:

- Production instance --> staging instance --> UAT instance --> production instance.
- Production instance --> staging instance --> production instance.

The migration process has three parts:

1. **Migrating data from the production instance of the tenant to the staging instance of the tenant.** This process is described in [Creating the First Staging or UAT Instance of the Tenant from the Production Instance of the Tenant](#).
2. **Migrating data from the staging instance of the tenant to the UAT instance of the tenant.** Although this step is optional, we recommend it.
  - a. Migrating the transactional data from the production instance of the tenant to the UAT instance of the tenant. This is the same process as [Creating the First Staging or UAT Instance of the Tenant from the Production Instance of the Tenant](#).
3. **Migrating data from the UAT instance of the tenant to the production instance of the tenant** (if you migrated from the staging instance to the UAT) or **migrating data from the staging tenant to the production tenant** (if you did not migrate data to the UAT tenant).

## Conceptual Information: Migration Process Overview

There are two scenarios when you perform the migration process:

- When the customer initially purchases and provisions their system. See ["About Initially Provisioning the Production Instance of the Tenant"](#) below.
- When the database changes. See ["About Migrating Data After Creating the Initial Production Instance of the Tenant"](#) on page 22.

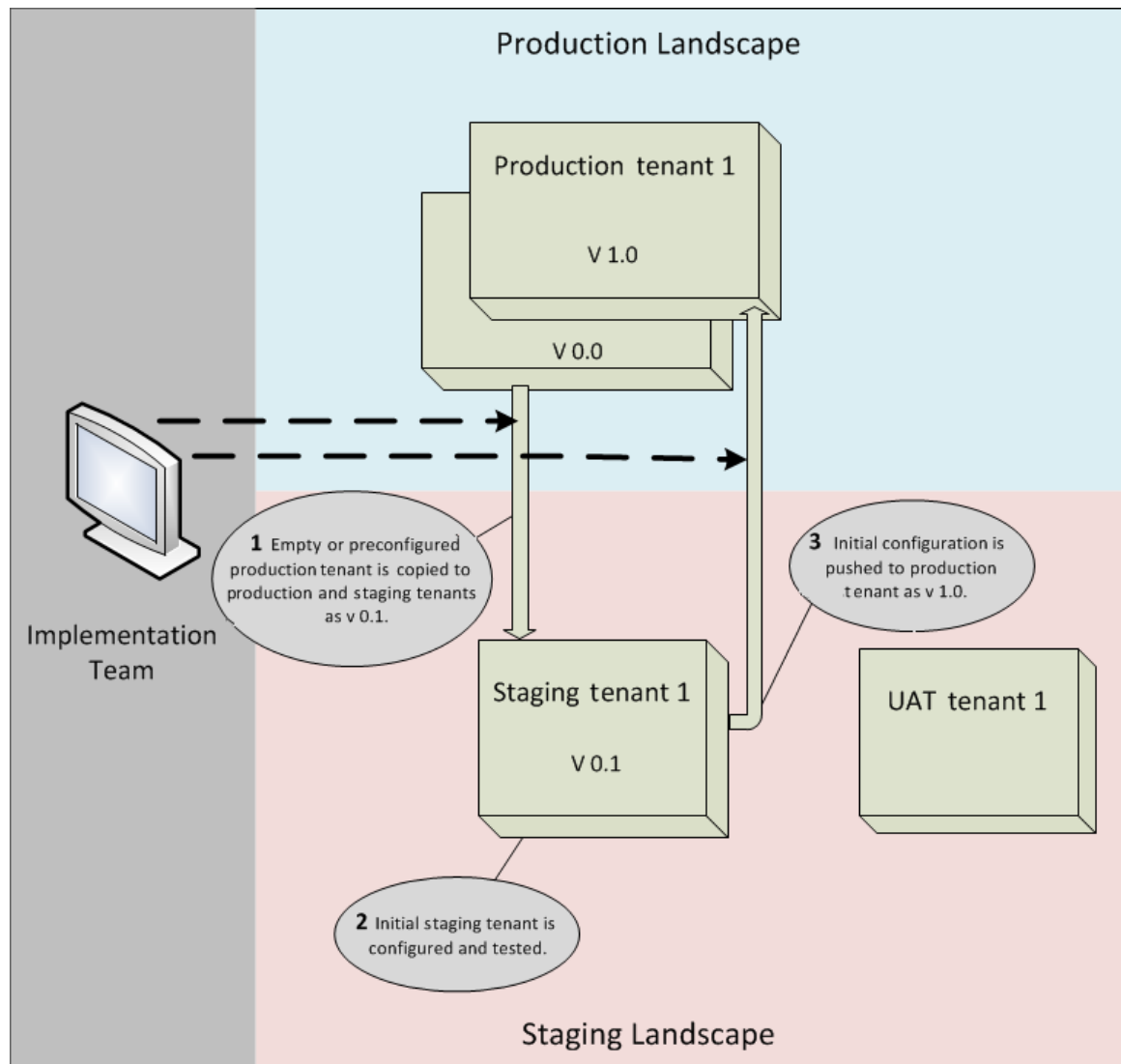
## About Initially Provisioning the Production Instance of the Tenant

- ["Overview of Initially Provisioning the Production Instance of the Tenant"](#) on the next page
- ["Overview of the Initial Production Instance of the Tenant Creation Process"](#) on page 20

## Overview of Initially Provisioning the Production Instance of the Tenant

"Adding a Tenant" below shows the process for migrating data from the initial production instance of the tenant (version 0.0) to the staging instance of the tenant, and then optionally from the staging instance of the tenant to the UAT instance of the tenant, and then from the staging (or UAT) instance of the tenant back to the production instance of the tenant. This process establishes the first production instance of the tenant (version 1.0). See "Overview of the Initial Production Instance of the Tenant Creation Process" on the next page for information about the numbered steps shown in "Adding a Tenant" below.

*Adding a Tenant*



Things to note in "Adding a Tenant" on the previous page:

- The version numbers used in this example are for reference only. The software residing on a tenant does not actually have the version numbers shown here.
- Version 0.0 of the production tenant represents the starting state of the production tenant immediately after it was created. This version (or instance) typically does not contain any transactional (customer) data. You can preconfigure the metadata by either using a customer template, or if no template is available, in a way that provides a default user interface look and feel. Examples of templates are a retail business template and a high tech template.
- When creating the initial version (or instance) of the production tenant (version 1.0), the system usually bypasses the UAT instance of the tenant because there is no baseline configuration to use as a reference for regression testing.

You do not make changes to configuration data and metadata directly in a running database on the production instance of the tenant. Instead, you copy the database to the staging instance of the tenant and makes the changes there. The production instance of the tenant remains unlocked throughout the provisioning and configuration process so that you can update the production instance of the tenant with minimal overhead. The system locks the production instance of the tenant for the first time after you verify the production readiness of the production instance of the tenant.

## **Overview of the Initial Production Instance of the Tenant Creation Process**

The following sections provide a high-level description of the initial production instance of the tenant creation process.

- "Before Starting the Migration" below
- "Step 1a: Creating the Initial Production Instance of the Tenant (Version 0.0)" on the next page
- "Step 1b: Creating the Initial Staging Instance of the Tenant (Version 0.1)" on the next page
- "Step 2: Updating the Data on the Staging Instance of the Tenant (Version 0.1)" on the next page
- "Step 3: Pushing the Data from the Staging Instance of the Tenant (Version 0.1) to the Production Instance of the Tenant (Version 1.0)" on the next page

### **Before Starting the Migration**

- Determine the customer's system requirements to help you design the initial production instance of the tenant (version 0.0).

**Step 1a: Creating the Initial Production Instance of the Tenant (Version 0.0)**

- Create the initial production instance of the tenant, called version 0.0. This initial production instance of the tenant contains basic help desk and other functionality based on industry standards and preliminary requirements. Version 0.0 is used as a starting point for additional configuration.
- The process for creating the initial production instance of the tenant is described in [How to Create a New Tenant](#).
- Send an email to the customer with the credentials for and a link to the initial production instance of the tenant (version 0.0).

**Step 1b: Creating the Initial Staging Instance of the Tenant (Version 0.1)**

- Move the initial production instance of the tenant (version 0.0) to the staging instance of the tenant. The version on the staging instance of the tenant is renamed version 0.1.
- Timing: This activity typically takes 5 days.

**Step 2: Updating the Data on the Staging Instance of the Tenant (Version 0.1)**

- Ask the customer to validate the configuration on the staging instance of the tenant.
- Validate the updated configuration on the staging instance of the tenant.
- Timing: This activity typically takes from two to twelve weeks.

**Step 3: Pushing the Data from the Staging Instance of the Tenant (Version 0.1) to the Production Instance of the Tenant (Version 1.0)**

- Remove any sample and test data that had been temporarily added to the staging instance of the tenant.
- Push the version 0.1 metadata and configuration data from the staging instance of the tenant to the production instance of the tenant, where it becomes the first fully functional release, called version 1.0.
- Ask the customer to verify that the production instance of the tenant configuration (version 1.0) is ready to "go live."
- Lock the production instance of the tenant, either after you receive the "go live" readiness confirmation or 10 days after the "go live" date if you do not receive any confirmation.
- Timing: This activity typically takes from two to eight weeks.

## About Migrating Data After Creating the Initial Production Instance of the Tenant

- "Requirements Before Migrating Data from the Staging or UAT Instance of the Tenant to the Production Instance of the Tenant" below
- "Overview of the Data Migration Process" below
- "Data Migration Process Overview" on the next page

### Requirements Before Migrating Data from the Staging or UAT Instance of the Tenant to the Production Instance of the Tenant

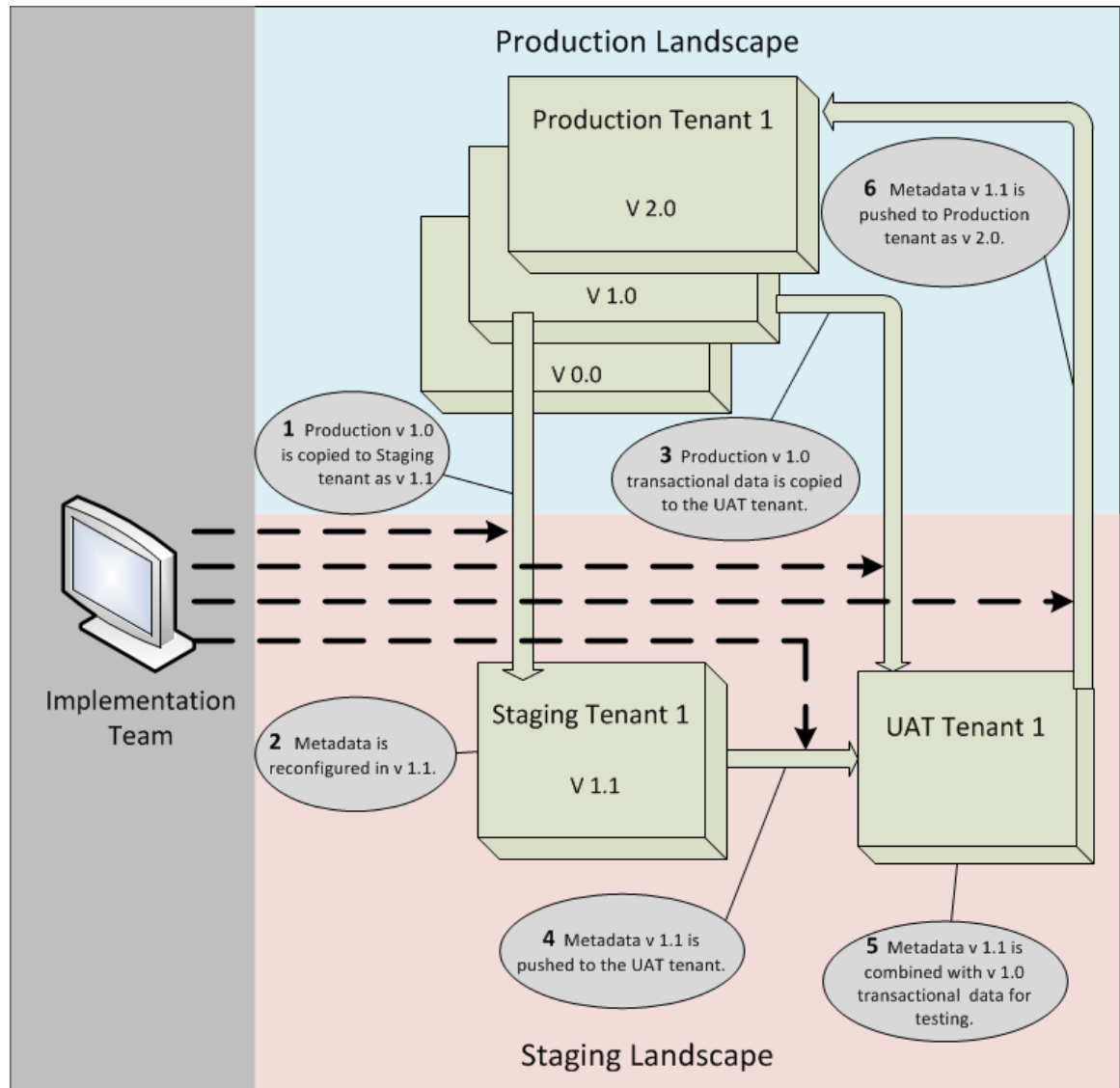
Ensure that you have completed the following requirements before you migrate data from the staging or UAT instance of the tenant to the production instance of the tenant:

- Verify that the logs for the staging or UAT instance of the tenant are free of all configuration-related errors. There cannot be any logs for quick actions, workflow, email, and so on, and there cannot be any "object reference not set to an instance of an object" errors.
- Ensure that all customer-related integration details (such as emails, LDAP, and VPN) have been documented and provided to the implementation team. Complete the information in the tab called **Email and Integration** and attach any associated documentation to the service request.
- Ensure that the implementation team completes the spreadsheet that lists the data to be cleared from the staging instance of the tenant before migrating the data to the production instance of the tenant. This might include details such as "truncate the following tables", "delete ... when...", and so on. Update the standard objects spreadsheet.
- Ensure that a service request contains the required go-live date. This does not guarantee the go-live date, but gives information about when it is planned. The implementation team plans and communicates this to the Solution/Implementation Manager.
- Provide a complete list of custom or updated reports. Export the RDL for all reports and attach it to the service request.

### Overview of the Data Migration Process

"Data Migration Process" below shows the steps that occur during a major update to a functioning production tenant instance (in this example, from version 1.0 to version 2.0).

*Data Migration Process*



Note the following:

- The version numbers used in "Data Migration Process" on the previous page are for reference only. The software residing on a tenant does not actually have the version numbers shown here.
- Version 1.0 of the production tenant instance represents a functional, running production tenant instance whose configuration remains unchanged until step 6 below.

## Data Migration Process Overview

This section describes the actions that occur when data is migrated to the production instance of a tenant.

- "Before Starting the Migration" on the next page

- "Step 1" below
- "Step 2" below
- "Step 3" on the next page
- "Step 4" on the next page
- "Step 5" on the next page
- "Step 6" on page 26
- "Afterward" on page 26
- "If a Step Fails" on page 27

**Before Starting the Migration**

- Work with the customer to create a comprehensive project plan describing the requested changes, change verification testing, project team contact information, schedule constraints, and other relevant information.

**Step 1**

- Copy version 1.0 of the production instance of the tenant in its entirety (that is, master data, metadata, configuration data, and transactional data) to the staging instance of the tenant. The copied version on the staging instance of the tenant is renamed version 1.1.
- Disable all email configurations (@customer.mx.domain) for the production instance of the tenant in the staging instance of the tenant, and enable the staging instance of the tenant email configuration (@tenantname-stg.saasit.com).
- Lock the production instance of the tenant. The production instance of the tenant cannot be reconfigured until approved configuration changes are applied in step 6. While the production instance of the tenant is locked, no one can make changes that will affect the migration process or be overwritten by the new configuration.
- Shrink the data on the production instance of the tenant to 5GB or less.
- The staging instance of the tenant is considered current and active.
- Timing: This step usually takes five days.

**Step 2**

- Edit the metadata (and optionally the configuration data) in version 1.1 to meet the customer's requirements.
- Record the configuration details in a change register.



- Timing: This step typically takes three weeks. It takes two weeks to prepare and migrate the configuration from the staging instance of the tenant to the UAT instance of the tenant, and one week within the UAT instance of the tenant for final testing.

**Step 3**

- Copy the transactional data (that is, customer data records) from the version 1.0 production instance of the tenant to the UAT instance of the tenant and if necessary shrink the data to less than 5 GB.
- Use the transactional data to test the metadata and configuration data updates in step 5. Version 1.0 metadata and other configuration data from the production instance of the tenant is not copied to the UAT instance of the tenant.
- Timing: This step typically takes three days.

**Step 4**

- Ask the customer to approve the metadata push from the staging instance of the tenant to the UAT instance of the tenant. If the customer does not approve it, send the approval request again. If the customer does not approve the request a second time, work with the customer to resolve any issues.
- Send the customer a tenant instance configuration migration template to complete, which provides details about exactly what data should be migrated.
- Review the migration template.
- Push version 1.1 of the metadata and configuration data to the UAT instance of the tenant. Only migrate the items included in the template. This is a full metadata migration from the source to the target; that is, this is not a delta metadata migration.
- Lock the UAT instance of the tenant, making it read-only and under change control.
- Timing: This step typically takes at least one week.

**Step 5**

- Ask the customer to perform acceptance testing on the UAT instance of the tenant using version 1.1 metadata and configuration data combined with version 1.0 transactional data. The customer performs the tests based on documented use cases to ensure that the configuration is "go-live" ready.
- The customer verifies that the UAT instance of the tenant configuration and logs are "go-live" ready. Do not update the production instance of the tenant if the UAT instance of the tenant is not "go-live" ready based on predefined requirements.

- All logs on the UAT instance of the tenant must be clear of any configuration-related failures. For example, there should be no logs reporting failures in workflows, quick actions, email, object references not set to instances of objects, and so on.
- You must approve all configurations. The customer must provide documentation describing integration details and attach it to the service request.
- Documentation must include the completed migration template, updated clean-up scripts (if necessary), descriptions of all configuration changes and additions, and test cases that the implementation team can use to verify tenant behavior.
- The customer approves the configuration push from the UAT instance of the tenant to the production instance of the tenant. If the customer does not approve the configuration, resend the approval request. If the customer does not approve the configuration the second time, work with the customer to resolve issues and validate the changes.
- Timing: Testing on the UAT instance of the tenant typically takes two days.

### **Step 6**

- After the customer has approved the migration and submitted the clean-up scripts, remove any sample and test data that had been temporarily added to the staging instance of the tenant and then migrated to the UAT instance of the tenant.
- Push the UAT instance of the tenant configuration (that is, the version 1.1 metadata and configuration data) to the production instance of the tenant. This is a full metadata migration from the source to the target; that is, it is not a delta metadata migration.
- Timing: Because this step could result in a brief outage of the production instance of the tenant, we recommend that you schedule it to occur during a maintenance window (usually on a Friday or weekend), unless the customer requests otherwise.

### **Afterward**

- The customer verifies that the production instance of the tenant is go-live ready.
- Request approval for the production instance of the tenant to go live. If the customer does not approve the request, work with them to resolve any issues and approve the production tenant.
- Place the UAT instance of the tenant in maintenance mode. It is no longer available to the customer.
- Lock the production instance of the tenant.
- Copy the configuration for the new production instance of the tenant to the staging instance of the tenant while the staging instance of the tenant is still locked.

- Unlock the staging instance of the tenant. From now on, the customer cannot use the staging instance of the tenant for further configuration without refreshing the production instance of the tenant. Refresh the staging instance of the tenant from the production instance of the tenant periodically to ensure that both tenant instances remain synchronized. The customer can use the staging instance of the tenant for testing and debugging purposes.
- Timing: This activity typically takes three business days.

### **If a Step Fails**

**Step 4:** If the metadata and configuration data migration from the staging instance of the tenant to the UAT instance of the tenant in step 4 fails, work with the customer to do the following:

- Obtain a corrected patch on the staging instance of the tenant.
- Re-initiate the process of migrating the metadata and configuration data from the staging instance of the tenant to the UAT instance of the tenant.

**Step 6:** If the metadata and configuration data migration from the UAT instance of the tenant to the production instance of the tenant in step 6 fails:

- Roll back the changes (that is, the backup copy is restored).
- Reinitiate the process of migrating the metadata and configuration data from the staging instance of the tenant to the UAT instance of the tenant.

## **Items Excluded from Migration**

- "Configuration Data (Settings)" below
- "Master Data" on the next page

### **Configuration Data (Settings)**

Configuration data, also called settings, is defined as core configuration attributes that define Ivanti Service Manager.

The only configuration data (settings) that the system migrates when you migrate configuration data (settings), are the following:

- Allowed attachments
- Authentication providers such as LDAP, SAML, OpenId, and Windows Integrated Security
- Bulk upload profiles
- Dynamic styles
- Global constants
- Integration settings

- LDAP configurations
- Object matching lists
- Password policies
- Voice integrations
- Watch list items
- Email configurations (including approval status keywords, email status update mapping settings, ignore message with subject settings, object mappings for email, task assignment status keywords, truncate body settings, and trusted sender domains)

You cannot migrate other types of configuration data (settings) using the Operations Console. To migrate this data, do so manually outside of the Operations Console by using Microsoft SQL commands.

## Master Data

You cannot selectively update master data. If you need to update only certain records of master data (for example, only one record in a table), use Microsoft SQL commands to do so outside of the Operations Console.

# Troubleshooting and Known Issues

- "Debugging the Migration" below
- "Known Issues in the Operations Console" on page 31

## Debugging the Migration

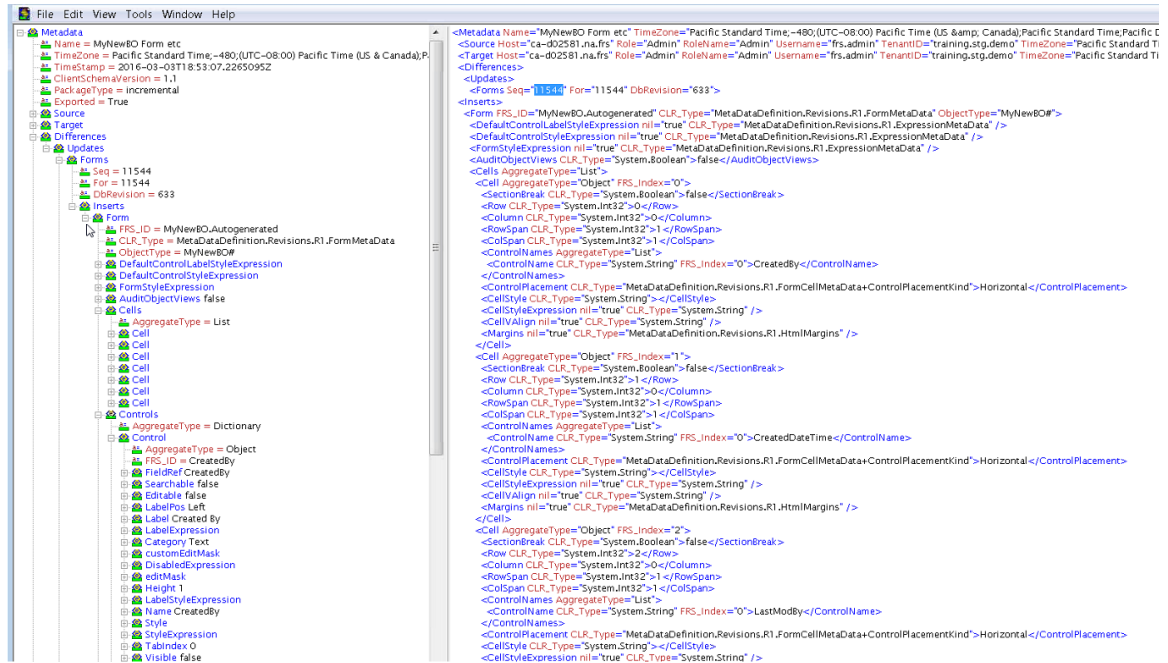
There are several ways to debug the data migration:

- "Previewing the Package Before Migrating It" below
- "Viewing the Warning and Error Messages" on the next page
- "Reviewing the Package Content" on page 31
- "Viewing the Operations Console Log Files" on page 31
- "Running Packages in the Correct Order" on page 31
- "Checking the API Keys" on page 31

## Previewing the Package Before Migrating It

If you used the Ivanti Service Manager development project, as described in "Appendix C: Using the ISM Development Project to Migrate Data" on page 34, you can preview the changes to ensure that they are correct. On the **Applying Package To** screen, click **Preview**. The system downloads an XML file with the proposed changes. Open the file in an XML editor and ensure that the changes are correct. The system associates all warnings to a change record by using a unique sequence number.

*Example of XML File with Proposed Changes*



## Viewing the Warning and Error Messages

After you perform the first migration from the production instance of the tenant to the staging instance of the tenant, review the error messages. You can review them in any of the following places:

- The patch log in the Ivanti Service Manager target tenant instance. See the Ivanti Service Manager online help for information.
- If you used the Ivanti Service Manager development project, as described in "Appendix C: Using the ISM Development Project to Migrate Data" on page 34, the system displays errors and warnings on the bottom of the **Applying Package To** screen.

*Sample Errors for a Package*

**Tenants** | **Users** | **System Status** | **Landscape** | **Events & Audit History**

**Applying Package To**

Target Tenant: training.uat.demo  
 Target Landscape: uat  
 From  
 Source Tenant: training.stg.demo  
 Source Landscape: staging  
 Source Package: MyNewBO Form etc  
 Skip Target Tenant Backup: ☒  
 Type of Execution: ☒ Validate ☐ Validate and apply if no errors ☐ Apply without validation  
 [Execute] [Preview] [Cancel]

**Package "MyNewBO Form etc" has been validated with the following 1 error and 1 warning**

**Error Message:**

1. Error during metadata commit operation for seq = 11546: System.Exception: Inserting Grid (MyNewBO.Autogenerated) - List MyNewBO Autogenerated definition is invalid: Table MyNewBO# is not found.

**Warning Message:**

1. Warning during metadata update operation for seq = 11545. MetadataType: BusinessObject, ID: MyNewBO#, Item does not exist.

## Reviewing the Package Content

Review the package content:

- Check the **Transaction Details** workspace in Ivanti Service Manager. This workspace lists the details of the package migrations.

## Viewing the Operations Console Log Files

Find and open the log files from the source tenant instance and the target tenant instance and compare the metadata.

## Running Packages in the Correct Order

If you used the Ivanti Service Manager development project, as described in "Appendix C: Using the ISM Development Project to Migrate Data" on page 34, and your migration contains multiple packages, ensure that you migrate the packages in the correct order. If items must be migrated in a specific order and you do not follow that order, the system may display an error. To avoid this, we recommend creating a master package that contains the incremental packages in the correct order.

## Checking the API Keys

Check the API keys. Each API key must be unique and cannot be shared between tenant instances. See About System Information.

## Known Issues in the Operations Console

This section contains information about issues and limitations in the Operations Console.

- "Migrating Logos" below
- "Cannot Migrate Data if a Business Object Contains More than 100,000 Records" below
- "Some Dashboard and Saved Search Properties are Not Migrated" on the next page
- "Problems Creating Packages" on the next page

## Migrating Logos

### Symptom

When migrating data that includes a logo, the updated logo does not migrate.

### Affected Release

All releases.

### Resolution

The Operations Console does not touch the logo during data migration. It does not remove the logo, but it also does not update the logo. Therefore, if you change the logo and then migrate data, after the data migration you must manually upload the new logo.

Also, changing the logo is not something that gets tracked in Ivanti Service Manager. Therefore, you cannot use the package process for this item.

## Cannot Migrate Data if a Business Object Contains More than 100,000 Records

### Symptom

The Operations Console cannot migrate database tables with more than 100,000 records. In some deployments, the **ServiceReqParam** table can have more than 100,000 records.

### Affected Release

All releases.

### Resolution

We have removed the **ServiceReqParam** tables so they are no longer part of the service request copy in the Operations Console.

We recommend as a best practice that your database tables be smaller than 100,000 records. We also recommend that if you do have a table with more than 100,000 records, you do not migrate it using the Operations Console.



If you do have a table with more than 100,000 records that needs to be migrated, use Microsoft SQL Server commands to manually copy the data and its relationships, or break the table into smaller tables, migrate the smaller tables, and then reform the data into a larger table after the migration.

## **Some Dashboard and Saved Search Properties are Not Migrated**

### **Symptom**

Some properties (such as "Set as default for myself", "Default for ...", and "My default") for dashboards and saved searches are not migrated.

### **Affected Release**

All releases.

### **Resolution**

These "default" properties are not metadata and are therefore not copied.

## **Problems Creating Packages**

### **Symptom**

When creating packages, it can be difficult to create the correct package for the migration. It can be challenging to find the right pieces of metadata to package and to find all of the dependencies.

### **Affected Release**

All releases.

### **Resolution**

- Before making any changes, always add a Ivanti Service Manager development project.
- Before making any changes, select a project. It is easier for you to include all changes in a project to a package.
- Review the transaction details to find and review transaction sets. You can search based on the object type, business object, entity name, and the name of the person who made the change. You can then add the transaction sets to the package.
- You may need to do a few iterations before you get the right package.

## Appendix C: Using the ISM Development Project to Migrate Data



The information in this appendix is copied directly from the Ivanti Service Manager online help. It describes how to use the Ivanti Service Manager Configuration Console to create packages, which can be used when migrating data. The online help contains additional information about the Ivanti Service Manager development project, such as cautions and limitations about the using the Ivanti Service Manager development project, terminology, information about logs and validation, and examples. (See "Related Documentation" on page 5 for more information about accessing the online help.)

You perform these tasks in the Ivanti Service Manager Configuration Console *and not in the Operations Console*.

This appendix contains the following sections:

- Create a project. See "Creating a Project" on the next page.
- Optional. View the project. See "Viewing a Project" on the next page.
- Make changes to Ivanti Service Manager. See "Creating a Transaction Set" on page 36.
- Optional. View the transactions sets. See "Viewing Transaction Sets" on page 36.
- Create a package. See "Creating a Package" on page 38.
- Optional. View the package. See "Viewing a Package" on page 38.
- Assign transaction sets to a package. See "Assigning Transaction Sets to a Package" on page 39.
- Add an aspect to a package. See "Adding an Aspect to a Package" on page 40.
- Map a package. See "Adding Packages to a Master Package (Mapping)" on page 41.
- Close the package. See "Closing a Package" on page 42.
- Optional. Export a package to another system. See "About Exporting a Package" on page 42.
- Optional. Import a package from another system. See "Appendix C: Using the ISM Development Project to Migrate Data" above. If you get errors, see "Appendix C: Using the ISM Development Project to Migrate Data" above.
- Optional. Save the package as a file. See "Saving a Package as a File" on page 47.
- Optional. Delete a package. See "Deleting a Package" on page 47.

After performing all of these steps, you can migrate the data to the target system.

For information about the packages that have been migrated, see the "Viewing the Patch Log" topic in the Ivanti Service Manager online help. (See "Related Documentation" on page 5 for more information about accessing the online help.)

## Creating a Project

You must create a project before you make any configuration changes. If you do not make any projects, all changes are logged to the default project. Use projects to organize the changes made to Ivanti Service Manager. After you create a project and *before* you make any configuration changes, always select your project from the **Project** drop-down list on the top right.

Follow these steps, on the system (such as staging) on which to make changes, to create a project:

1. Log in to Ivanti Service Manager using your named account and not as an administrator. See the Ivanti Service Manager topic called "About Using Your Named Account to Log In" for information about why you must use your named account to log in. (See "Related Documentation" on page 5 for more information about accessing the online help.)
2. Click **Configure Application** to go to the Configuration Console.
3. Click **Build > HEAT Development Project > Projects**. The system displays the **Projects** workspace.
4. Click **New Record**. The system displays a blank project page.
5. Enter a name and description for the project and click **Save**.
6. Click **Refresh**. You can now see the new project name under the **Project** drop-down menu in the top right section of the toolbar (next to the **Help** link).

## Viewing a Project

1. From the Configuration Console, click **Build > HEAT Development Project > Projects** to display the **Projects** workspace. The system displays a list of all the projects.
2. Double click a project name to see details of the project.
3. Click the **Transaction Set** tab to see a list of the transaction sets in this project. See "Viewing Transaction Sets" on the next page.
4. Click the **Transaction Detail** tab to see the details of each transaction set. See "Viewing Transaction Sets" on the next page.
5. View the activity history by clicking the **Activity History** tab, any attachments by clicking the **Attachment** tab, or the audit history by clicking the **Audit History** tab.

## Creating a Transaction Set

Every time you add, delete, or modify the Ivanti Service Manager configuration, the system records the changes. These recorded changes are called a transaction set. For example, every time that you save changes for a business object, the system records the changes in a transaction set. The system tracks every change and assigns each change a sequential number.

1. From the Configuration Console, click **Build > HEAT Development Project > Projects** to display the **Projects** workspace. The system displays a list of all the projects.
2. From the **Project** drop-down menu in the toolbar, select the project to which the changes you are going to make will belong.
3. Make changes to Ivanti Service Manager as normal. For example, you can create a new business object, form, or list.

---

In certain cases, when you make changes to Ivanti Service Manager, you must enter your user name and password. When you create and later export the transaction set as part of a package, the package file exposes the user name and password.



To avoid exposing this potentially sensitive information, we recommend that when setting the user name and password for an integration connection, instead of entering a value, you use a global constant. See the "Using Global Constants to Hide Login ID and Password Information" topic in the Ivanti Service Manager online help for information. See the "Viewing, Cloning, and Deleting Web Service Connections" topic in the Ivanti Service Manager online help for an example of using a global constant in this situation. (See "Related Documentation" on page 5 for more information about accessing the online help.)

If you do not use a global constant for the user name and password, the system removes that information from the transaction set.

---

## Viewing Transaction Sets

From the **Transaction Sets** page, you can view the relevant changes that are linked to a transaction set. You can search by transaction set ID. Use the transaction set ID when creating a package.

1. From the Configuration Console, click **Build > HEAT Development Project > Transaction Sets** to open the **Transaction Sets** workspace. The system displays a list of all the transaction sets.

The system displays the following information:

Field	Description
ID	The ID for this transaction set.
Created On	The date when the transaction set was created.

Field	Description
User ID	The ID of the user who created the transaction set.
Session ID	The ID of the session when the transaction set was created.
DB Revision	The database revision.
Source	Specifies where the transaction was made. <i>AdminUI</i> means that the change was made in the Configuration Console, and a tenant URL means that the change was made from a package migration.
Project Name	The name of the Ivanti Service Manager development project to which this transaction set is attached.

2. Double click a transaction set and click the **Transaction Detail** tab to see the following information:

Field	Description
Project Name	The name of the Ivanti Service Manager development project to which this transaction set is attached.
Transaction Set ID	A sequential number linked to the change. A transaction set may contain zero or more changes. (A zero change indicates the change was aborted, and was not recorded.)
Detail ID	A sequential number linked to the change. A detail may contain zero or more changes. (A zero change indicates the change was aborted, and was not recorded.)
User ID	The ID of the user who created the transaction set.
Operation Date Time	The date and time when the change was recorded.
Operation Type	The type of operation. Can be <b>C</b> for new record, <b>U</b> for modification, or <b>D</b> for deleted.
Object Type	The type for which the change applies. The values can be <b>BusinessObject</b> for a business object change, <b>Form</b> for a form change, or a business object name such as <b>AccountStatus#</b> for change in its data, etc.
Business Object	The name of the business object to which the change is linked, if any.
Entity Name	The key for the business object record itself. In some cases, it is the GUID and in other case, it is a name, such as a business object name, a form name, or a role name.
Source ID	The key for the business object record itself. In some cases, it is the GUID and in other case, it is a name, such as a business object name, a form name, or a role name.
To Publish	Specifies if this transaction set can be published:

Field	Description
	<b>Y:</b> The change is packaged.
	<b>N:</b> This change is a side effect of another change that is published.

## Creating a Package

Follow these steps, on the system (such as staging) on which you have made changes, to create a package:

1. From the Configuration Console, click **Build > HEAT Development Package > Packages**. The system displays the **Packages** workspace.
2. Click **New Record**. The system displays a blank package page.
3. Enter a descriptive name for the package. The name must be unique across the system.
4. From the **Type** drop-down list, select if this is an incremental or master package. You cannot edit the type after you save create the package.
  - An incremental package is a single change. It can contain a set of metadata changes and can be imported and exported. There are two types of incremental packages: the first type is exported from the tracked changes in the data and metadata. Every data and metadata change is tracked with a sequence number, which specified the import order. The second type is a result of comparing the full metadata from two tenant instances (such as UAT and staging). There is no sequence associated with these changes. This content comes from the traditional Operations Console migration.
  - A master package contains one or more incremental, master, or file packages. The packages in the master package are called sub-packages. For example, you may want to create a master package with several incremental packages (sub-packages) in it.
5. In Ivanti Service Manager Release 2020.2, the **Predicate** field is for internal use only. Do not enter anything here.
6. Enter a description.
7. Click **Save**.
8. Click **Refresh** to see the **Close Package**, **Save As File**, and **Export Package** icons on the top and the tabs at the bottom.

## Viewing a Package

1. From the Configuration Console, click **Build > HEAT Development Package > Packages** to open the **Packages** workspace. The system lists the Ivanti Service Manager development packages.
2. Double click a package to view its details.

The system displays the following tabs:

- **Transaction Set** tab (for incremental packages only). See "Viewing Transaction Sets" on page 36.
- **Transaction Detail** tab (for incremental packages only). See "Viewing Transaction Sets" on page 36.
- **Package Mapping** tab (for master packages only). See "Adding Packages to a Master Package (Mapping)" on page 41.
- **Package Aspect** tab. See "Adding an Aspect to a Package" on the next page.
- **Activity History** tab. See the "About Activity History" topic in the Ivanti Service Manager online help.
- **Attachment** tab. See the "Adding an Attachment" topic in the Ivanti Service Manager online help.
- **Audit History** tab. See the "About Accessing the Audit History" topic in the Ivanti Service Manager online help.

(See "Related Documentation" on page 5 for more information about accessing the Ivanti Service Manager online help.)

## Assigning Transaction Sets to a Package

You must create an incremental type package before you can assign transaction sets to it. See "Creating a Package" on the previous page.

Follow these steps, on the system (such as staging) on which you have made changes, to assign transaction sets to a package:

1. From the Configuration Console, click **Build > HEAT Development Project > Transaction Sets** to open the **Transaction Set** workspace.
2. Select the transaction sets to include in the package.
3. Click **Assign To Package**.
4. Use the drop-down menu to select the name of the package to which to assign the transaction sets and click **OK**.



You can also assign a transaction set to a package by clicking **Build > HEAT Development Package > Packages** to open the **Packages** workspace, clicking the **Transaction Set** tab, and clicking **Add Change Set**.

---

5. To check that the transactions set are in the package, do the following:
  - a. Click **Build > HEAT Development Package > Packages**. The system displays the **Packages** workspace.

- b. Open the package. (If you do not see it, click **Refresh**.) The system opens the package.
    - c. Click the **Transaction Set** tab to view all of the transaction sets.
  6. To remove a transaction set from a package, do the following:
    - a. Click **Build > HEAT Development Package > Packages**. The system displays the **Packages** workspace.
    - b. Open the package. (If you do not see it, click **Refresh**.) The system opens the package.
    - c. Click the **Transaction Set** tab to view all of the transaction sets.
    - d. Highlight the transaction set.
    - e. Click **Remove From Package**.
  7. To add a transaction set to another package, do the following:
    - a. Click **Build > HEAT Development Package > Packages**. The system displays the **Packages** workspace.
    - b. Open the package. (If you do not see it, click **Refresh**.) The system opens the package.
    - c. Click the **Transaction Set** tab to view all of the transaction sets.
    - d. Highlight the transaction set to add to the package.
    - e. Click **Assign To Package**.
    - f. Select a package name from the drop-down list and click **OK**.

## Adding an Aspect to a Package

You can only add aspects to a package if it is open. When you created the package, the system copied the aspects from the original package. You can remove the original aspects and add new ones.

1. From the Configuration Console, click **Build > HEAT Development Project > Transaction Sets** to open the **Transaction Sets** workspace.
2. Assign one or more transaction sets to a package.
3. Click the **Package Aspect** tab.

A package can have zero or more aspects. When the package is imported into target tenant instance, these aspects are entered into or removed from the target tenant instance to flag the aspect presence.

4. View the following information on the **Package Aspect** tab:



Field	Description
To Uninstall	Specifies if the aspect should be removed.
Version	Numeric whole number for this aspect version. The version number for an aspect grows in its development life.
ID	A unique name for the aspect.
Vendor Name	The name of the vendor who provides the aspect. An example is (such as feature/component), Ivanti Software.
Vendor ID	A unique ID that identifies the aspect provider. Currently, the system uses "Ivantisoftware" for aspects provided by Ivanti Software. In future releases, other providers will have a unique vendor ID.
Created By	The ID of the administrator who added the aspect.
Created On	The date and time when the aspect was created.
Modified By	The ID of the administrator who modified the aspect.
Modified On	The date and time when the aspect was modified.

- Click **New Package Aspect**. The system displays the **New Package Aspect** dialog box.
- Check **Is Uninstalled** if this package aspect is uninstalled.
- Enter the following information:
  - Version number:** Any number.
  - ID:** Any text.
  - Vendor ID:** Any text.
- Click **Save**.

## Adding Packages to a Master Package (Mapping)

If you have a master package, you can add incremental or other master packages to it. In this way, the master package contains many changes. This eliminates the need to export incremental packages one at a time.

- From the Configuration Console, click **Build > HEAT Development Package > Packages** to open the **Packages** workspace.
- Open a master package.
- Click the **Package Mapping** tab.
- Click **Assign Package**. The system displays the **New Master Package Mapping** form.

5. In the **Sub Package** field, click the search icon to display a list of packages that can be mapped.
6. Select a sub-package and click **Use Selected HEAT Release Package**.
7. Enter a value into the **Relative Order** field. This determines the relative order of this sub-package compared to other sub-packages in the master package. The system applies the sub-packages to the tenant in this order (when you export and import the package).
8. Click **Save**.
9. Repeat steps 4 through 8 to add additional sub-packages to the master package.

## Closing a Package

You must close a package before you can use the Operations Console to migrate it from staging to production. Closing a package means that you cannot do any more work on it, such as adding or removing packages, adding or removing predicates, changing aspects, and so on. You do not have to close a package to export it.

We recommend that you only close a package after you have completed testing on it and you are ready to migrate it from either staging or UAT to production. After you close a package, you cannot reopen it.

When a package is closed, the system does not display the **Add Change Set** and **Remove From Package** buttons on the **Transaction Set** tab, nor does it display the **New Package Aspect** and **Delete** buttons on the **Package Aspect** tab.

Follow these steps to close a package.

1. From the Configuration Console, click **Build > HEAT Development Package > Packages**. The system displays the **Packages** workspace.
2. Highlight the name of the package to close.
3. Click **Close Package**.
4. At the confirmation prompt, click **OK**.

## About Exporting a Package

You can export a package of any type as an XML file and send it to another system, even if the package is empty. You can export any open or closed package. You can use any text editor to view the exported package.

- "About Compacting Export Data" below
- "Exporting a Package" on the next page

## About Compacting Export Data



Compacting export data is a beta feature. Use it with caution.

---

Starting in HEAT Service Management Release 2016.1, you can compact the transaction sets for a package when you export it. If you make multiple changes to one business object consecutively, compacting combines the individual changes into one compact change. When you import the package, the system only implements the one compacted change instead of the individual changes, which speeds up the process of importing a package.

For example, if you changed a quick action 12 times consecutively, then without compacting, the package would have 12 transaction sets. With compacting, there is only one transaction set.

There are a few caveats to this:

- The system only compacts changes to metadata.
- The system does not combine creating a business object and changing a business object into one compact package. The system only compacts changes to a business object and in this instance, changing a business object does not include creating it.
- The system only compacts changes to the same business object of the same metadata type. If you change two business objects, the system does not compact these two changes together.

## Exporting a Package

1. From the Configuration Console, click **Build > HEAT Development Package > Packages** to open the **Packages** workspace.
2. Highlight the name of the package to export.
3. Click **Export Package**. The system gives you the option of compacting the transaction sets. This is a beta feature.
4. Click **Yes** to compact the export data or **No** to not compact the export data. The system exports the package.

## Importing a Package



You should never import a package into the production environment from the Configuration Console when you have the Operations Console in place, as doing so could cause system instability.

We highly recommend that you always back up your database before importing a package.

Importing a package is an irreversible operation and cannot be undone.

---

You can import a package that has been previously exported. For master packages, the system imports the subpackages in the given relative order. The system imports the package if its predicate evaluates to true. If the predicate evaluates to false or an error, the system does not import the package or its subpackages. After importing the package, the system uses the package aspects to update the tenant.

Starting in HEAT Service Management Release 2016.1, the system scans the import package and assigns one of three impact levels (high, medium, and low) to each package.

Impact Level	Definition
High	<ul style="list-style-type: none"> <li>Contains a database schema change.</li> <li>Deletes a business object.</li> <li>Updates a stored property for a field of a business object.</li> <li>Updates the <b>IsLocalized</b> property for a field of a business object.</li> <li>Updates the data type, length, or precision for a field of a business object.</li> <li>Updates a unique property for a field of a business object.</li> <li>Updates a nullable property for a field of a business object.</li> <li>Changes the index.</li> <li>Changes the full-text search index.</li> <li>Adds or deletes an audit field.</li> </ul>
Medium	<ul style="list-style-type: none"> <li>Contains changes to the metadata for a business object.</li> <li>Adds a business object.</li> <li>Changes the metadata of a business object, other than the changes described above.</li> <li>Changes or overrides a business rule.</li> <li>Changes or deletes a form, list, layout, quick action, pick list definition, dashboard, chart part, list part, or special part.</li> <li>Changes or deletes a role.</li> <li>Changes or deletes branding.</li> <li>Changes a workflow, counter, view part, global constant, request offering, or report.</li> <li>Changes the email configuration.</li> <li>Changes integration settings.</li> <li>Changes the LDAP configuration or authentication provider.</li> <li>Changes the password policy.</li> <li>Changes an escalation schedule.</li> </ul>
Low	<ul style="list-style-type: none"> <li>Does not contain a database schema change or changes to the metadata for a business object, or anything listed above.</li> <li>Adds a form, list, layout, quick action, pick list definition, dashboard, chart part, list part, special part.</li> <li>Add a role.</li> <li>Adds branding.</li> </ul>

Unlike in previous releases, starting in HEAT Service Management Release 2016.1, you can sometimes import a package even when the system is in metadata read-only mode.

When the system is in metadata read-only mode, note the following:

- For high impact packages, you can only validate the import package. You cannot import the package.
- For medium impact packages, you can validate and import the package, but there may be risks with importing the package. We highly recommend that you back up the database first.
- For low impact packages, you can validate and import the package. We highly recommend that you back up the database first.

Follow these steps to import a package:

1. From the Configuration Console, click **Build > HEAT Development Package > Packages** to open the **Packages** workspace.
2. Click **Import Package**.
3. Click **Browse...** to navigate to the import package.
4. Highlight the file and click **Open**. The system displays a dialog box that lists the file name of the import package.
5. Check **Show Detail** to see the details of the changes in the import package and then click **Preview Impact** to scan the complexity of the import package. Depending on the impact of the import package, the system displays a screen that lists the details of the import package impact.  
  
If you do not check **Show Detail**, the system only displays the first high or medium level change in the package.
6. Click **Close**. The system displays the **Import Package** dialog box stating that you should back up the database before importing the package, and that this action is irreversible.
7. Do one of the following:
  - Select **Validate** to check the integrity of the package without applying it. If there are any errors, you can go back and fix them before importing the package.
  - Select **Validate and apply if no errors** to import the package only if there are no errors. Note that the system may import the package even if there are errors on the application side. The system displays a message stating if there are any errors and where they are, either on the application side or on the validation side.
  - Select **Apply without validation**. We do not recommend this option. With this option, you assume all responsibility to correct any errors after the import. This imports the package even if there are errors. You should always validate the package and understand the errors first, and then select this option only if you agree to ignore the errors.
8. Click **Execute**
9. At the confirmation prompt, click **Yes**. The system starts the import process.



If the package contains many changes, the import process can take a long time. Your system may time out or you may accidentally get disconnected from the system due to loss of wifi or VPN. If this happens, log in again. The system continues to import the package and display the importing message. However, the system may not dismiss the importing message, even after the import process has finished. In this case, click **Cancel** or **Refresh** to dismiss it.

The system displays a message about the success or failure of the import, including specifying if there are any warnings or errors.



If there are any errors, you cannot import the package. If you do not understand the impact of the errors, we recommend that you restore your database back to the production environment, as the error may disrupt your production operation. See "Appendix C: Using the ISM Development Project to Migrate Data" on page 34 or contact Ivanti Software for further information about the errors.

10. Click **Close**.

11. To view the imported package, do the following:

- a. Click **Build > HEAT Development Project > Projects** to open the **Projects** workspace.
- b. Under the **Name** column, double-click the project that you just imported, to open it.

## Troubleshooting Package Import Errors

If you get any errors while importing a package, the package import fails. You must correct the errors before importing the package. If you get any warnings, the import continues; however, we strongly recommend that you correct the warnings before importing the package.

The system displays warnings in orange text and errors in red text.

Try the following for troubleshooting import package warnings and errors:

- If the error message references a sequence number, open the package in a text editor such as Notepad and look for that sequence number. Examine the package for information.
- From the Configuration Console, click **Build > HEAT Development Project > Transaction Sets** to open the **Transaction Set** workspace. Open the transaction set associated with the warning or error. Examine the transaction set for information.

## Saving a Package as a File


You can save an incremental or master package as a file (but you cannot save a package as a file if it is already a file) so that you can have a record of it at a certain time. This is similar to versioning the package. You can export or import packages saved as files. All aspects in the package are copied to the file, but you can edit them after they are copied to the file.

Follow these steps to save a package as a file:

1. From the Configuration Console, click **Build > HEAT Development Package > Packages** to open the **Packages** workspace.
2. Highlight the name of the package to save.
3. Click **Save As File**.
4. Enter a new name or description, or leave the default values.
5. Click **Save**. The system saves the package as a file.
6. Click **Refresh** to see the new file in the file list.

## Deleting a Package

You can only delete a package if it is open and if it is not being used by another package.

1. From the Configuration Console, click **Build > HEAT Development Package > Packages** to open the **Packages** workspace.
2. Highlight the name of the package to delete.
3. Click the **delete** icon .

## Appendix D: Working with Packages from the App Store

- "About the App Store" below
- "Accessing the App Store" below
- "Downloading and Importing an App Store Package" on the next page
- "Troubleshooting Problems with App Store Packages" on page 50

### About the App Store

Starting in HEAT Service Management Release 2015.1, we released a new applications store (App Store) where you can download optional software created by both Ivanti Software and other companies. The optional software in the App Store is bundled into packages.

The App Store is a collection of apps that can extend your Ivanti Software applications. Some apps include special packages that you can import into Ivanti Service Manager to modify the behavior of the system; other apps are integrations to other applications.

To create your own app, see the link "How to Develop an App" on the home page of the App Store.

The App Store provides different certification levels for the apps, which help you understand the level of support provided for a given app. Some apps are available on an "as-is" basis as a sample, and are available to use at your own risk. Some apps are supported directly by Ivanti Software and some apps come from certified software partners.

Not all apps have an associated package; some require licensing additional software from partners to enable the functionality. These requirements are described in the app description.

### Accessing the App Store

Follow these steps to access the App Store:

1. Navigate to <https://support.heatsoftware.com>.
2. Do the following:
  - a. Enter your user name and password.
  - b. Press **Login**.

The system may require you to log in using external authentication. If so, enter your user name and click **Sign in with your\_company\_name**.

3. If the system prompts you, select the Self Service role and click **Submit**. The system opens to the Self Service home page.
4. From the top tool bar, click **More...** and select **App Store**. You are now in the App Store.



## Downloading and Importing an App Store Package

1. Access the App Store as described in "Accessing the App Store" on the previous page.
2. Open the app to use. Download the documentation associated with the app, and ensure that you meet all of the prerequisites.
3. Download the .metadatapatch files associated with the app. These are the actual package files to be imported.
4. Import the package into the staging instance of your tenant by following the directions in "Importing a Package" on page 43. This automatically creates a project for this package in your staging system.



You should always validate any new package before applying it to a production environment, even partner certified or Ivanti Software-supplied packages. Package behavior can vary based on your specific configuration. By doing the validation in staging, any necessary changes can also be applied.

---

5. Test the package on the staging instance of your tenant.
6. If you need to make any changes to your system based on your testing, do the following:
  - a. Open the project that the system created when you imported the package. Ensure that this is the active project so that the system adds any changes that you make to this project.
  - b. Make the required changes.
  - c. Create a new package for the changes. See "Creating a Package" on page 38.
  - d. Export the package. See "Exporting a Package" on page 43.



If you did not make any changes, you do not need to create a new package as you can just use the one that you imported in step 3.

---

7. Refresh the UAT instance of your tenant. It is very important that the UAT instance accurately reflects the current state of the production instance of your tenant, to get an accurate UAT assessment.
8. Import the package into the UAT instance of your tenant by following the directions in "Importing a Package" on page 43. This automatically creates a project for this package in your UAT system.
9. Test the package on the UAT instance of your tenant.
10. If you need to make any changes to your system based on your testing, do the following:

- a. Open the project that the system created when you imported the package. Ensure that this is the active project so that the system adds any changes that you make to this project.
- b. Make the required changes.
- c. Create a new package for the changes. See "Creating a Package" on page 38.
- d. Export the package. See "Exporting a Package" on page 43.



If you did not make any changes, you do not need to create a new package as you can just use the one that you imported in step 7.

---

11. Close the package when you are satisfied with it and know that you will make no more changes.
12. Back up the production instance of your tenant.
13. Import the package into the production instance of your tenant by following the directions in "Importing a Package" on page 43. The system analyzes the impact of the package.
14. If the impact is green or yellow, you can import the package directly. We recommend doing this during off hours, so users are not impacted by the change.
15. If the impact is red, create an incident with Support requesting that they push the package to the production instance of your tenant. After they receive the request, Support moves the package into the production environment.

## Troubleshooting Problems with App Store Packages

- You must import the package into the staging or UAT instance of your tenant. Support cannot import the package for you.
- You must close the package before you send it to Support.
- If Support finds any errors while migrating the package, they will send it back to you with the log that contains the error messages. You must fix the errors and then repeat the process of testing the package in the staging instance of your tenant, exporting the package to the UAT instance of your tenant, and requesting a new migration from Support.
- If there are any problems with the data migration, Support will restore the database from a backup. Report any issues quickly to minimize losing any new transactions because of the restoration from a backup.