

1. User Guide	2
1.1 Tasktop Editions	2
1.2 Installation Primer	3
1.3 User Management	19
1.4 Key Concepts	38
1.5 Quick Start Guide	53
1.5.1 Step 1: Connect to Your Repository	54
1.5.1.1 Standard Repository Connection	55
1.5.1.2 Database Repository Connection	62
1.5.2 Step 2: Create or Reuse a Model	65
1.5.3 Step 3: Create Your Collection(s)	75
1.5.3.1 Repository Collection (Standard)	76
1.5.3.2 Repository Collection (Database)	104
1.5.3.3 Gateway Collection	111
1.5.4 Step 4: Configure your Integration	115
1.5.4.1 Synchronize Integration Template	116
1.5.4.2 Create via Gateway Template	152
1.5.4.3 Modify via Gateway Template	173
1.5.4.4 Enterprise Data Stream Template	191
1.5.5 Step 5: Expand or Modify your Integration	220
1.6 Settings	226
1.7 Upgrading	234
1.8 Troubleshooting	236
1.9 Extensions	243
1.10 Business Continuity	267
1.11 Resources	269
1.12 Appendix A- Error Messages	269

User Guide



Welcome to User Guide

17.4 Release (October 24, 2017)

New to Hub? You can:

- Learn about our product's [key concepts](#)
- Read about [hardware requirements and installation](#)
- Explore our [Quick Start Guide](#)
- Learn about our [Connectors](#)
- Check out our [Release Notes](#)
- Learn about [new features](#) in this release

Search the
Documentation

Need help? [Contact support here](#)

Tasktop Editions





Tasktop: 17.4 Release

Tasktop Integration Hub is available in three editions.

We've included the table below to help you understand which features are included in your edition.

If you are interested in learning more about other editions, please [contact us](#)

	Pro	Enterprise	Ultimate
Lifecycle Connectors			

Included Lifecycle Connectors	Connect Any 2 Lifecycle Tools	Connect up to 5 Lifecycle Tools	Unlimited
Automation			
Gateway Integration Style (Create via Gateway Template; Modify via Gateway Template)	Available as add-on	Available as add-on	Unlimited
Visibility			
Enterprise Data Stream (Enterprise Data Stream Template)		Available as add-on	Unlimited
Integration Landscape View			

Installation Primer


Tasktop: 17.4 Release

Overview

This section describes how to install Tasktop Integration Hub and covers some basic information you should know before proceeding with the installation. If you are working on a deployment with Tasktop, your Solutions Architect will assist you with the installation.

Hardware Requirements

Tasktop Integration Hub must be installed in a server environment. You will need an account with administrative privileges on your server to install and configure Tasktop Integration Hub.

 **Note:** Only one instance of Tasktop should be installed on each server.

Supported Operating Systems

The following 64-bit operating systems and versions are supported:

- Windows 7 SP1
- Windows 10
- Windows Server 2008 R2 SP1
- Windows Server 2012 R2
- Windows Server 2012


- Overview
- Hardware Requirements
 - Supported Operating Systems
 - Supported Browsers
 - Supported Databases for storing Tasktop Operational Data
 - Supported Databases for use in Enterprise Data Stream Integrations
 - Java Runtime Environment
 - Hardware Sizing for Deployment Scenarios
- Sandbox Environment
- Installation

- Windows Server 2016
- Red Hat Enterprise Linux 6.x
- Red Hat Enterprise Linux 7.x
- Ubuntu Linux 12.04 LTS
- Ubuntu Linux 14.04 LTS
- Ubuntu Linux 16.04 LTS
- SUSE Linux Enterprise Server 11.x
- SUSE Linux Enterprise Server 12.x


Supported Browsers

The Tasktop Integration Hub web interface is supported on the following browsers:

- Internet Explorer 11 or later
- Firefox 46.0.1 and up
- Chrome 50.0.2661.102 and up

 Tasktop Integration Hub has been developed to run with a minimum screen resolution of 1280 pixels by 800 pixels.

Supported Databases for storing Tasktop Operational Data

 Tasktop automatically stores operational data to a built-in database. However, for production environments, we strongly recommend that operational data is stored to an external database for improved maintainability. This will enable you to perform frequent back-ups without having to stop Tasktop Integration Hub, and ensure that your Tasktop Integration Hub practices are consistent with your existing disaster and recovery process.

For details on how to store your operational data to an external database, rather than Tasktop's built-in database, please refer to the [Settings page](#).

You can also learn more about Disaster Recovery on Tasktop Integration Hub [here](#).

The following databases and versions are supported for storing Tasktop operational data:

- Microsoft SQL Server 2008 (including SP1, SP2, SP3, SP4)
- Microsoft SQL Server 2008 R2 (including SP1, SP2, SP3)
- Microsoft SQL Server 2012 (including SP1, SP2)
- Microsoft SQL Server 2014 (including SP1)
- Microsoft SQL Server 2016
- Oracle 11g
- Oracle 12c

Supported Databases for use in Enterprise Data Stream Integrations

The Tasktop Database add-on allows you to create integrations that send artifact information to one central database.

- Where to Download Tasktop Integration Hub
- Installation on Windows
- Installation on Linux
- Getting Started
- Default File Locations
 - Default File Locations on Windows
 - Default File Locations on Linux
- Endpoint Preparations
 - Preparing Your ALM Systems
 - Firewalls and Proxies
- Advanced Configuration
 - Container Configuration
 - Port Configuration
 - HTTPS Configuration
 - Increasing Available Memory
 - Logging

If your license includes the Tasktop Database add-on and you would like to configure an [Enterprise Data Stream Integration](#), the following databases and versions are supported:

- Microsoft SQL Server 2008 (including SP1, SP2, SP3, SP4)
- Microsoft SQL Server 2008 R2 (including SP1, SP2, SP3)
- Microsoft SQL Server 2012 (including SP1, SP2)
- Microsoft SQL Server 2014 (including SP1)
- Microsoft SQL Server 2016
- MySQL 5.5
- MySQL 5.6
- MySQL 5.7
- Oracle 11g
- Oracle 12c

Java Runtime Environment

Tasktop Integration Hub is packaged with a JRE and there is no need to install a JRE separately. Tasktop Integration Hub uses and ships with Oracle Java.

Hardware Sizing for Deployment Scenarios

Following are recommendations on sizing hardware and virtual machine capacity to meet the needs of typical deployment scenarios.

Tasktop Integration Hub is a web application which runs centrally on a server. Users interact with it through a web browser from any computer that has network access to the server. These sizing recommendations apply to the server machine running Tasktop Integration Hub.

These recommendations are guidelines intended to provide a starting point when deciding on hardware allocation for a specific deployment. We recommend monitoring system load including CPU usage, memory pressure and disk queue length and adjusting the system sizing accordingly.

For best results, Tasktop Integration Hub should be deployed in an environment that has good network throughput and low latency to all repositories and databases involved in an integration.

Small Deployment

A deployment managing up to 20,000 ALM artifacts and up to 200 active users.

- 4 GB system memory
- 3 GHz processor, 2 cores
- 50 GB free disk space

Small deployment system sizing is roughly equivalent to an EC2 T2 Medium instance.

Medium Deployment

A deployment managing up to 100,000 ALM artifacts and up to 1,000 active users.

- 8 GB system memory
- 3 GHz processor, 2 cores
- 150 GB free disk space

Medium deployment system sizing is roughly equivalent to an EC2 T2 Large Instance.

Large Deployment

A deployment managing many ALM repositories and 200,000+ ALM artifacts and over 2,000 active users.

- 8 GB system memory
- 2 x 3 GHz processors, 4 cores
- 250 GB free disk space

Large deployment system sizing is roughly equivalent to an EC2 M4 Large or M3 Large Instance.

Sandbox Environment

It is recommended that you prepare a sandbox environment to test your Tasktop Integration Hub configuration before deploying it in production. This sandbox environment should include a sandbox server to install Tasktop Integration Hub on, and sandbox instances of all ALM systems you will be integrating, with the same project structure and customizations as, and a comparable number of artifacts to your production ALM systems.

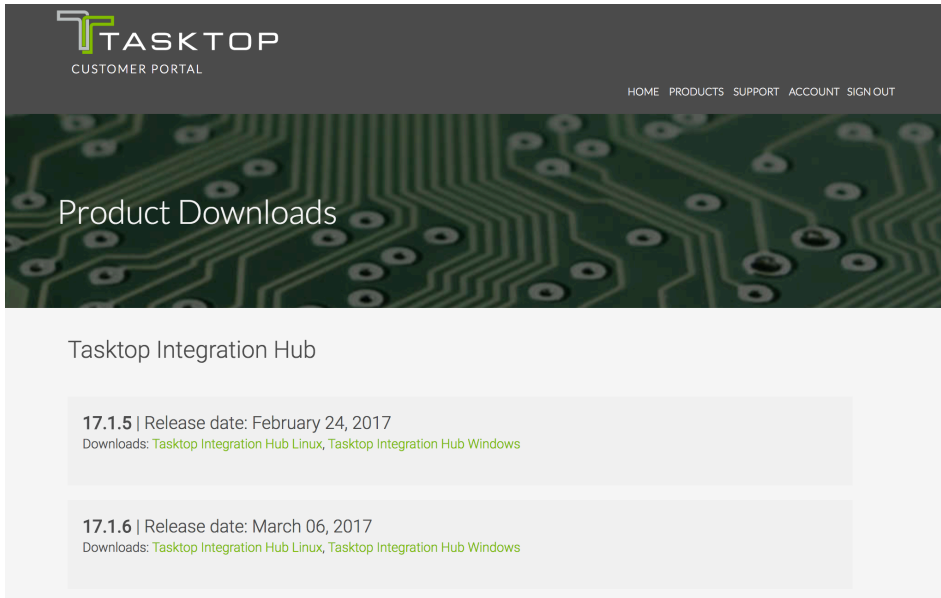
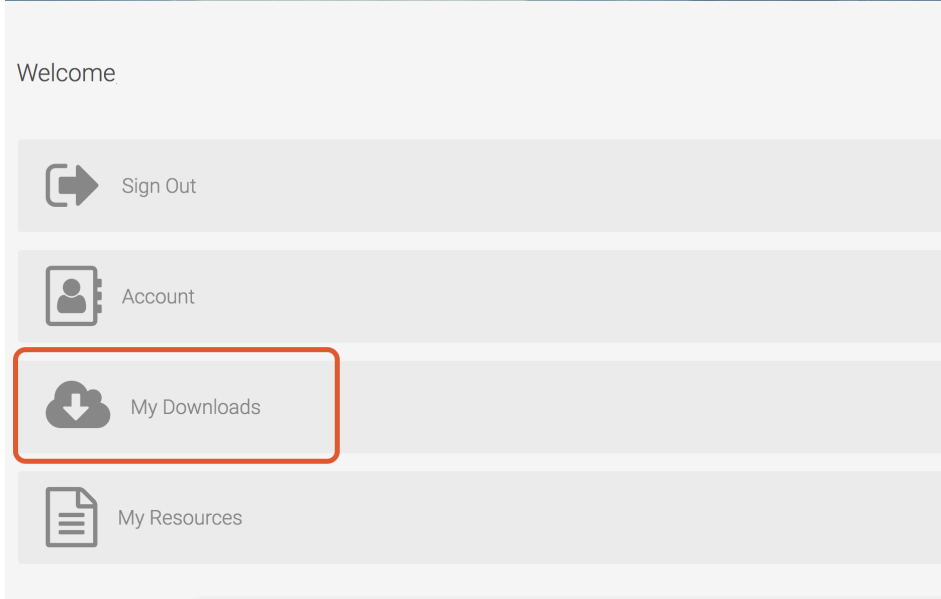
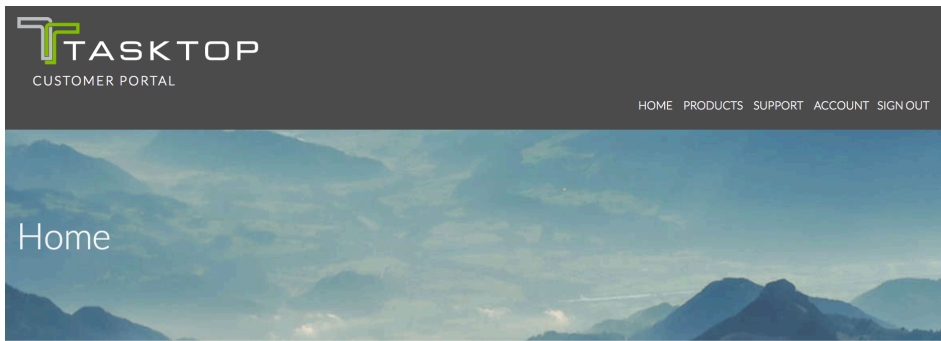
After you have configured Tasktop Integration Hub on the sandbox server and are happy with the way it is running against your sandbox ALM systems, you can install Tasktop Integration Hub on your production server and recreate the configuration against your production ALM systems.

Installation

Where to Download Tasktop Integration Hub

To get the latest version of Tasktop Integration Hub, first create an account on <http://my.tasktop.com>, then contact your Solutions Architect or Tasktop Support (support@tasktop.com) and ask them to enable the latest Tasktop Integration Hub download for your account.

Once on <http://my.tasktop.com>, click the 'My Downloads' button. This will lead you to http://my.tasktop.com/download_products.php, where you will be able to download the latest version of Tasktop Integration Hub.

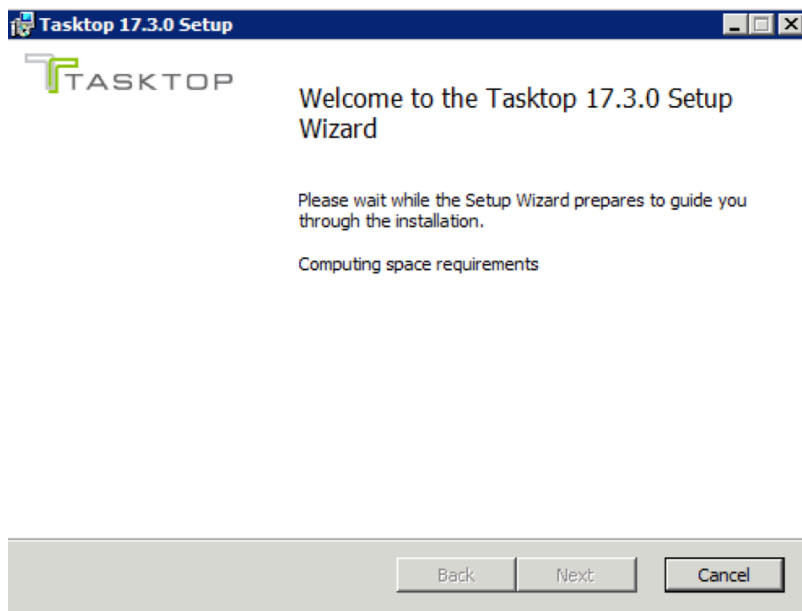
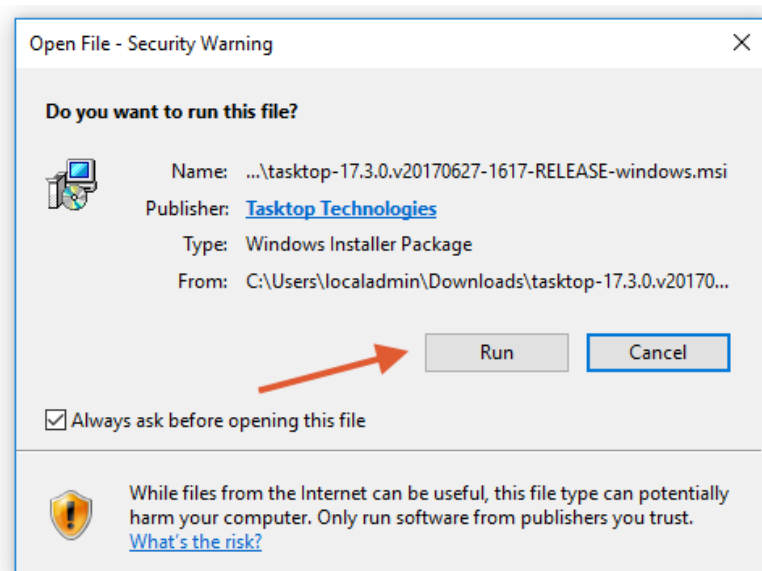


Installation on Windows

Click on the 'Windows' download link on the Product Downloads page of my.tasktop.com.

You will be provided with an installation package for Tasktop Integration

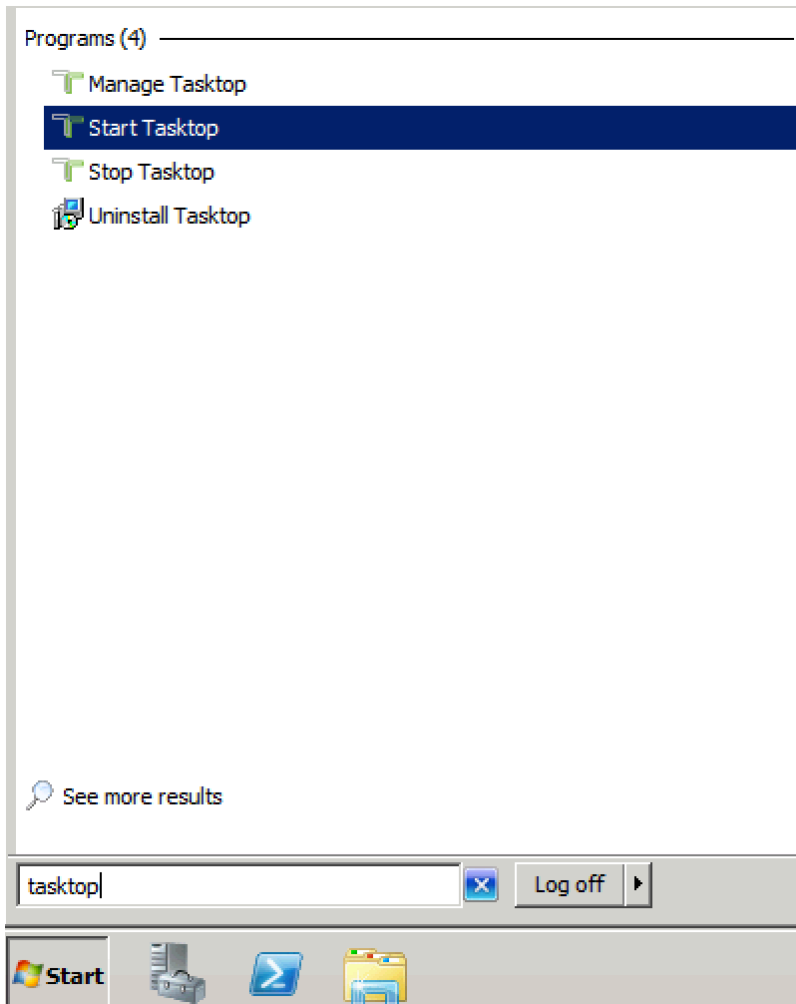
Hub as a standard Windows MSI installer. If prompted, click 'Save File,' and then open the file once it downloads.



You will then be lead through the installation wizard. Follow the prompts to install Tasktop.

To start Tasktop, click the 'Start' menu, and select 'Start Tasktop'. This will start both Tasktop and Keycloak User Management services. To stop both services click on the 'Stop Tasktop' shortcut.

💡 Please make sure you follow the steps in the [Getting Started](#) section upon starting up Tasktop Integration Hub for the first time.



Installation on Linux

For Direct Customers


Click on the 'Linux' download link on the Product Downloads page of my.tasktop.com.

You will be provided with an installation package for Tasktop Integration Hub as a `.tar.gz` archive.

To extract this archive to your desired location, copy the archive to the correct location on your Linux system and use following command to extract:

```
$ tar xzvf tasktop-linux-x64-<version>.tar.gz
```

To start Tasktop Integration Hub, run the `start-tasktop.sh` script from the installation directory. This will start both Tasktop and Keycloak User Management services. To stop both services, use the `stop-tasktop.sh` script in the same folder.

 Please make sure you follow the steps in the [Getting Started](#) section upon starting up Tasktop Integration Hub for the first time.

For OEM Customers

You will be provided with an installation package for Tasktop Integration Hub with no file extension in the name.


To execute the file, run these commands:

```
chmod+x tasktop-linux-x64-<version>
```

```
./tasktop-linux-x64-<version>
```

Once you approve the End User License Agreement that pops up, the file will automatically unzip, allowing you to run Tasktop Integration Hub.

To start Tasktop Integration Hub, run the `start-tasktop.sh` script from the installation directory. This will start both Tasktop and Keycloak User Management services. To stop both services, use the `stop-tasktop.sh` script in the same folder.

 Please make sure you follow the steps in the [Getting Started](#) section upon starting up Tasktop Integration Hub for the first time.

Tasktop Integration Hub Service on Linux

There are multiple ways to configure a Tasktop Service that starts automatically on system startup. It is recommended to use a dedicated account for running Tasktop Integration Hub. Here are examples for SysVinit and Systemd.

Tasktop Integration Hub Service with Systemd

1. Navigate to `/etc/systemd/system`
2. Create a new file named `tasktop.service`
3. Paste the following into that file

```
# Systemd unit file for tasktop
[Unit]
Description=Tasktop Integration Hub
After=syslog.target
network.target

[Service]
Type=forking

ExecStart=/path/to/tasktop/start-tasktop.
sh
ExecStop=/path/to/tasktop/stop-tasktop.sh

User=user
Group=group

[Install]
WantedBy=multi-user.target
```

- a. Be sure to change both instances of '/path/to/tasktop' to the full path to your Tasktop Integration Hub installation directory
- b. Be sure to change the User and Group variables to the username and group of the account you want to run the Tasktop Integration Hub service

4. Reload Systemd

```
$ systemctl daemon-reload
```

5. Enable the new Tasktop Integration Hub service to start on system startup

```
$ systemctl enable tasktop
```

To manually start and stop the Tasktop Integration Hub Service, use the following commands:

```
$ systemctl start tasktop
$ systemctl stop tasktop
```

Tasktop Integration Hub Service with SysVinit

1. Navigate to /etc/init.d

2. Create a new file named `tasktop`
3. Paste the following into that file:

```
#!/bin/bash
# description: Tasktop Start Stop Restart
# processname: tasktop
# chkconfig: 2345 20 80
TASKTOP_HOME=/path/to/tasktop
case $1 in
start)
sh $TASKTOP_HOME/start-tasktop.sh
;;
stop)
sh $TASKTOP_HOME/stop-tasktop.sh
;;
restart)
sh $TASKTOP_HOME/stop-tasktop.sh
sh $TASKTOP_HOME/start-tasktop.sh
;;
esac
exit 0
```

- a. Be sure to change the `TASKTOP_HOME` variable to the full path to your Tasktop Integration Hub installation directory
 - b. You may also wish to change the `chkconfig` run levels and start and stop priorities
4. Set the permissions of Tasktop to make it executable

```
$ chmod 755 tasktop
```

5. Use the `chkconfig` utility to make Tasktop Integration Hub start at system startup (you may wish to change the run levels in this command)

```
$ chkconfig --add tasktop
$ chkconfig --level 2345 tasktop on
```

To manually start and stop the Tasktop Integration Hub Service, use the following commands:


```
$ service tasktop start
$ service tasktop stop
$ service tasktop restart
```

Derby Database Location on Linux

Hub has an internal Derby database that stores Tasktop operational data. You may want to change the location of this database so that only specific individuals may access it on the Linux machine where Hub is installed.

Here's how to change the location of the Derby database:

1. Download and then unzip the Linux file.
2. Go to `container/bin/setenv.sh`. You can change the database location under `derby.system.home`.
3. Now, when you start the Hub service, your database will be in the location you specified.

 **Note:** if you have already made configuration changes, you will have to manually move the existing Derby database to the new location. Otherwise, your Hub instance will appear brand new without any configuration.

Getting Started

Once installation is complete, you can begin using Tasktop Integration Hub by opening <http://localhost:8080> or <https://localhost:8443> in any of our [supported browsers](#).

Before logging on to Tasktop, you must log into the User Administration Console in order to create your admin user(s). The Tasktop User Administration Console can be accessed via the 'User Administration Console' link, at the bottom of the Tasktop Integration Hub sign-in page. Please review the [User Management](#) section for detailed instructions on how to create a user, log in, and manage your user accounts.

Once logged in, you will be prompted to set a [Master Password](#), which will be used to encrypt your repository credentials.

You will also need to apply your license before configuring your integrations. You can learn how to apply your license [here](#).

Default File Locations

Default File Locations on Windows

When Tasktop Integration Hub is installed on Windows using the MSI installer, the program files (i.e. the executable files and binaries) are located in `C:\Program Files\Tasktop`, and the configuration files and logs are located in `C:\ProgramData\Tasktop` (ProgramData may be a hidden folder, so you will need to change your Windows Explorer settings to show hidden files and folders to find it).

Default File Locations on Linux

When Tasktop Integration Hub is installed on Linux, the program files

(i.e. the executable files and binaries), configuration files, and logs are all located in the installation directory where you extracted the distribution archive.

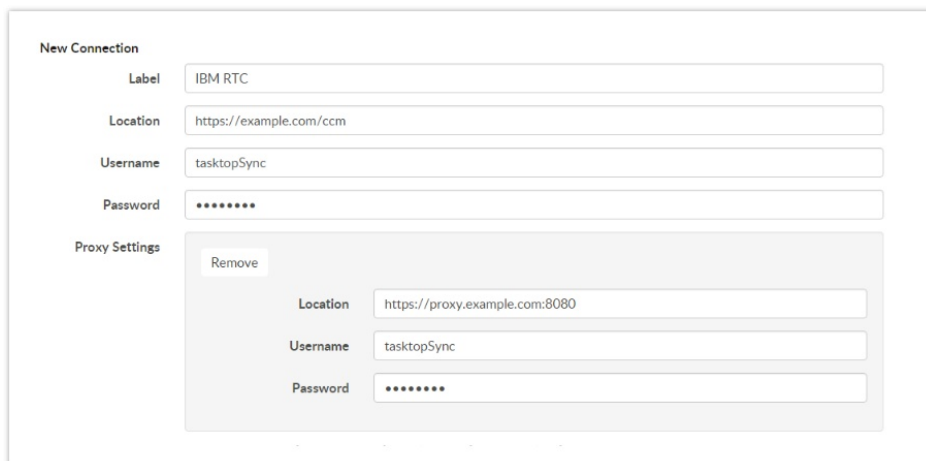
Endpoint Preparations

Preparing Your ALM Systems

Before using Tasktop Integration Hub with your ALM Systems you will need to perform some simple preparation on each ALM System you will be integrating. This preparation includes creating a user account for Tasktop Integration Hub with the appropriate permissions, and possibly other steps. Please refer to the specific preparation document for each of your ALM systems for detailed instructions.

Firewalls and Proxies

If Tasktop Integration Hub is installed behind a firewall, you may need to connect to external ALM systems (e.g. hosted or cloud ALM systems) through a proxy. To create a connection to such external ALM systems in Tasktop Integration Hub, you can make Tasktop Integration Hub connect through your proxy by configuring the proxy settings when creating a new repository connection (see Figure 1). It is recommended to create login credentials specifically for Tasktop Integration Hub on the proxy server.



The image shows a 'New Connection' configuration form. It has the following fields and sections:

- Label:** IBM RTC
- Location:** https://example.com/ccm
- Username:** tasktopSync
- Password:** [masked]
- Proxy Settings:** A shaded box containing:
 - Remove:** A button.
 - Location:** https://proxy.example.com:8080
 - Username:** tasktopSync
 - Password:** [masked]

Advanced Configuration

Container Configuration

Tasktop is distributed with the Apache Tomcat Servlet Container.

For information on configuring the container, please refer to the Apache Tomcat documentation at <http://tomcat.apache.org/tomcat-7.0-doc/>.

On Windows, configuration and log files are installed under C:\Progra

mData\Tasktop while program files are located under C:\Program Files\Tasktop.

For information on configuring the service, please refer to the Apache Tomcat Service Howto at <https://tomcat.apache.org/tomcat-7.0-doc/windows-service-howto.html>.

Further configuration, including JVM options and memory allocation, can be performed for the Windows service by launching "Tasktop Properties" located at C:\Program Files\Tasktop\container\bin\tasktopw.exe.

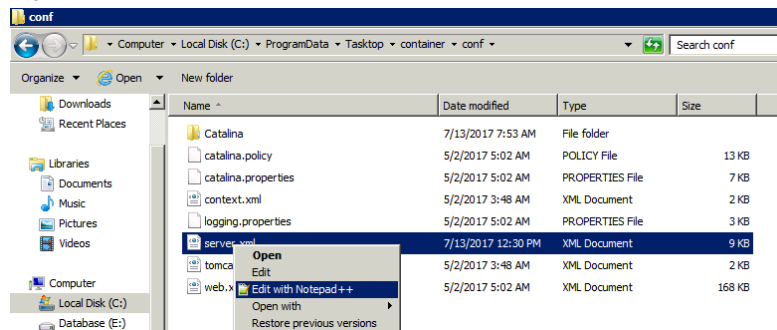
Port Configuration

💡 To view a list of all ports being used on your system, you can use the `netstat -a` command. This will help you determine which available port you'd like to use for Tasktop or Keycloak.

Tasktop Integration Hub

The default port Tasktop uses is 8080 for http and 8443 for https. To change this port, follow these instructions:

1. In the Tasktop workspace (default: C:\ProgramData\Tasktop), open container/conf/server.xml
 - a. Note: You may need to right click and select 'Edit with Notepad,' or some other similar option in order to edit the file



2. To change the HTTP port:
 - a. Find the HTTP connector configuration (the <Connector> element with `protocol="HTTP/1.1"`)
 - b. Change the port attribute to the port you wish to use (e.g. to use port 8888: `<Connector port="8888" protocol="HTTP/1.1" connectionTimeout="20000" redirectPort="8443" />`)

```

60
61
62
63 <!-- A "Connector" represents an endpoint by which requests are received
64 and responses are returned. Documentation at :
65 Java HTTP Connector: /docs/config/http.html (blocking & non-blocking)
66 Java AJP Connector: /docs/config/ajp.html
67 APR (HTTP/AJP) Connector: /docs/apr.html
68 Define a non-SSL HTTP/1.1 Connector on port 8080
69
70 <Connector port="8088" protocol="HTTP/1.1"
71 connectionTimeout="20000"
72 redirectPort="8443" />
73
74 <!-- A "Connector" using the shared thread pool-->
75 <!--
76 <Connector executor="tomcatThreadPool"
77 port="8080" protocol="HTTP/1.1"
78 connectionTimeout="20000"
79 redirectPort="8443" />

```

- c. Save the file
3. To change the https port
 - a. Find the HTTP connector configuration (the <Connector> element with protocol="HTTP/1.1")
 - b. Change the redirectPort attribute to the port you wish to use (e.g. to use port 9443: <Connector port="8080" protocol="HTTP/1.1" connectionTimeout="20000" redirectPort="9443" />)
 - c. Find the SSL HTTP connector configuration (the <Connector> element with protocol="org.apache.coyote.http11.Http11NioProtocol")
 - d. Change the port attribute to the port you wish to use (e.g. to use port 9443: <Connector port="9443" protocol="org.apache.coyote.http11.Http11NioProtocol" ... />)

If you change the port, the address used to access Tasktop (i.e. <http://localhost:8080>) will need to be updated with the new port number in place of '8080.'

Please refer to the official documentation for additional configuration options: <http://tomcat.apache.org/tomcat-7.0-doc/config/http.html>

Keycloak User Management

The default port the User Management uses is 8081. To change this port, follow these instructions:

1. In the Tasktop workspace (default: C:\ProgramData\Tasktop), open container/webapps/ROOT/WEB-INF/web.xml
 - a. Find the targetURI parameter for the KeycloakProxy servlet (<param-name>targetUri</param-name>)
 - b. Change the param-value attribute to the port you wish to use (eg.g. to use port 9081 : <param-value><http://127.0.0.1:9081/auth></param-value>)
2. In the Tasktop workspace (default: C:\ProgramData\Tasktop), open keycloak/standalone/configuration/standalone.xml

- a. Find the socket-binding for http configuration (`<socket-binding name="http" port="{jboss.http.port:8081}"/>`)
- b. Change the port attribute to the port you wish to use (e.g. to use port 9081: `<socket-binding name="http" port="{jboss.http.port:9081}"/>`)

HTTPS Configuration

By default, the application is available via HTTPs on port 8443. A default SSL certificate is provided for testing purposes. This SSL certificate is insecure. Before use in a production environment, the provided SSL certificate must be replaced. To replace the certificate, it is necessary to create a new keystore with a valid certificate. Follow the steps below to create and configure the keystore:

1. Locate the Java keytool utility in Tasktop installation directory (default `C:\Program Files\Tasktop`), at `<Tasktop Installation Directory>/jre/bin/keytool`
2. Generate a keystore file with the following command and fill in the information as prompted

```
keytool -genkey -alias tomcat -keyalg RSA
-keystore <location to save keystore
file>
```

3. (Optional steps) Obtain and import a Certificate from a Certificate Authority (CA)
 - a. Generate a Certificate Signing Request (CSR) with the following command

```
keytool -certreq -keyalg RSA -alias
tomcat -file certreq.csr -keystore
<your keystore file>
```

- b. Submit your CSR to a CA to obtain a Certificate (see your CA's documentation for detailed instructions)
 - c. Download a Chain Certificate from your CA (see your CA's documentation for detailed instructions)
 - d. Import the Chain Certificate into your keystore with the following command

```
keytool -import -alias root
-keystore <your keystore file>
-trustcacerts -file <your chain
certificate file>
```

- e. Import your Certificate into your keystore with the

following command

```
keytool -import -alias tomcat
-keystore <your keystore file> -file
<your certificate file>
```

4. Place your keystore file in In the Tasktop installation directory (default C:\Program Files\Tasktop), at *<Tasktop Installation Directory>/container*)
5. In the Tasktop workspace (default: C:\ProgramData\Tasktop) , open *container/conf/server.xml*)
 - a. Find the SSL HTTP connector configuration(the `<Connector>` element with `protocol=protocol="org.apache.coyote.http11.Http11NioProtocol"`)
 - b. Change the `keystoreFile` attribute to point to the new keystore file
 - c. Change the `keystorePass` attribute to the password you entered when generating the new keystore file
6. Restart Tasktop Integration Hub Service

By default the SSL configuration has been configured to disable known weak ciphers. As new security information becomes available, the list of enabled ciphers should be updated accordingly.

For more details about Tomcat SSL configuration, please refer to <http://tomcat.apache.org/tomcat-7.0-doc/ssl-howto.html>.

Increasing Available Memory

On Linux, Tasktop runs with the default JRE memory settings. This is typically a 1/4th of the physical memory or 1 GB whichever is less. To change the available memory edit `container/bin/setenv.sh` and add the following line replacing 1536 with the desired amount of heap memory:

```
JAVA_OPTS=-Xmx1536m
```

On Windows, the available memory defaults to 512 MB and can be changed through the Manage Tasktop application. The desired amount of memory is specified on the Java tab under "Maximum memory pool".

Logging

Logging is configured with log4j. See the included "log4j.xml" to configure log levels, location, and rolling policy.

The included "log4j-troubleshooting.xml" configures log4j for the troubleshooting log level when set via the settings page of the

application.

User Management

Tasktop: 17.4 Release

Getting Started

Once [installation](#) is complete, you can begin using Tasktop Integration Hub by opening <http://localhost:8080/> or <https://localhost:8443> in any of our [supported browsers](#).

Before logging on to Tasktop, you must log into the User Administration Console in order to create your admin user(s). The Tasktop User Administration Console can be accessed via the 'User Administration Console' link, at the bottom of the Tasktop Integration Hub sign-in page.



Connecting the World of Software Delivery

Sign in to continue to Tasktop

A sign-in form with a light gray background. It contains two input fields: 'Username' with the text 'admin' and 'Password' which is empty. Below the password field is a 'Remember me' checkbox which is checked. To the right of the checkbox is a blue 'Sign In' button.

Username

Password

Remember me

Visit the [User Administration Console](#) to add and configure users.

This will lead you to the Keycloak log-in screen:

- Getting Started
- Creating Additional Users
- Resetting a User's Password
- Managing Groups
 - Viewing Members of a Group
 - Adding or Removing Users From a Group
- Modifying Your Own User Information
- Advanced User Management
 - Configuring LDAP User Management
 - Required Directory Information
 - Accessing Keycloak Configuration Tool
 - Configuring LDAP for Active Directory
 - Required Settings
 - Kerb



The Tasktop User Administration Console comes pre-configured with a root user. Use those credentials to log into Keycloak.

Username: root

Password: Tasktop123

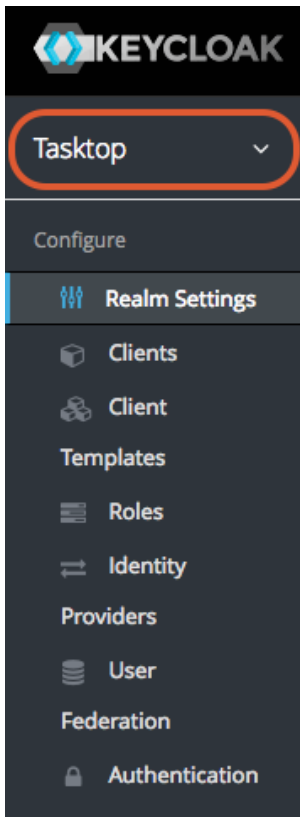
You will be prompted to change your root password.

⚠ WARNING: There is only one initial root user. If the credentials for this user are lost, access to the [advanced User Management features](#) will be lost. All functionality of Tasktop Integration Hub, however, will continue uninterrupted.

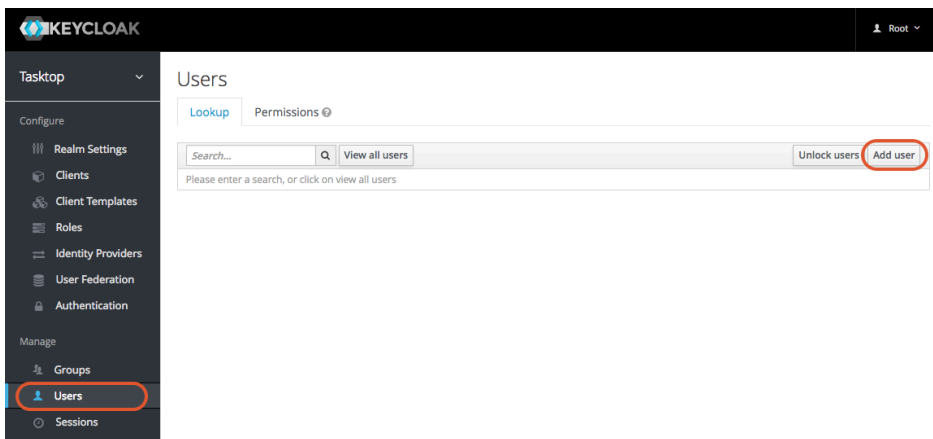
After logging in, you will need to make at least ONE new Tasktop Admin user for Tasktop Integration Hub. After this first user is created, you can create additional users directly from the Tasktop Integration Hub interface.

To create a Tasktop Admin, ensure "Tasktop" is selected in the upper left:

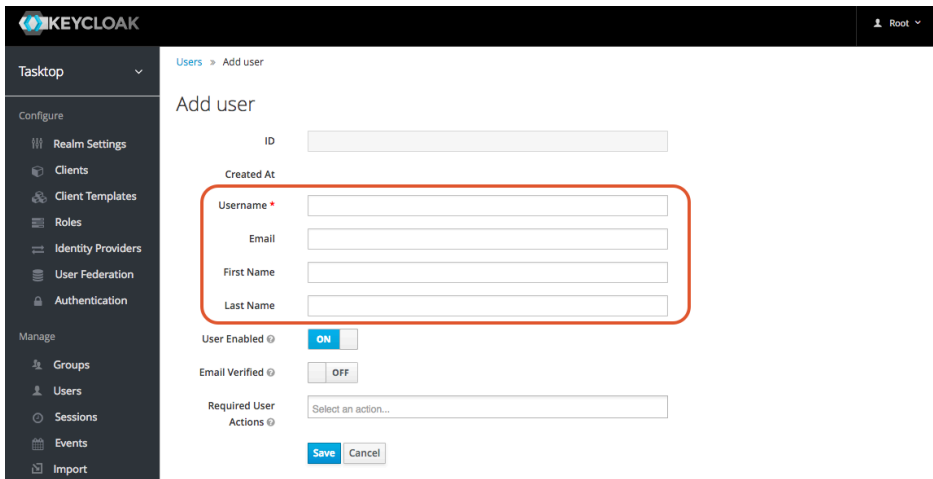
- er
- os
- Sync Settings
- Additional LDAP Information
 - Testing
 - Default User Access
- User Management and SSL
- User Management and Security Constraints
- DNS Settings
- Alternative User Management



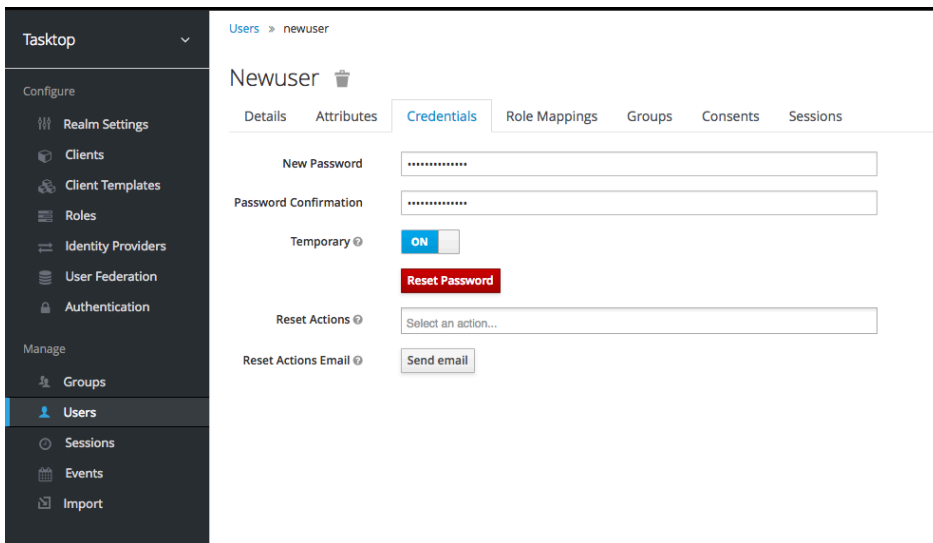
Select the 'User' section in the left column and click on the 'Add user' button on the upper right.



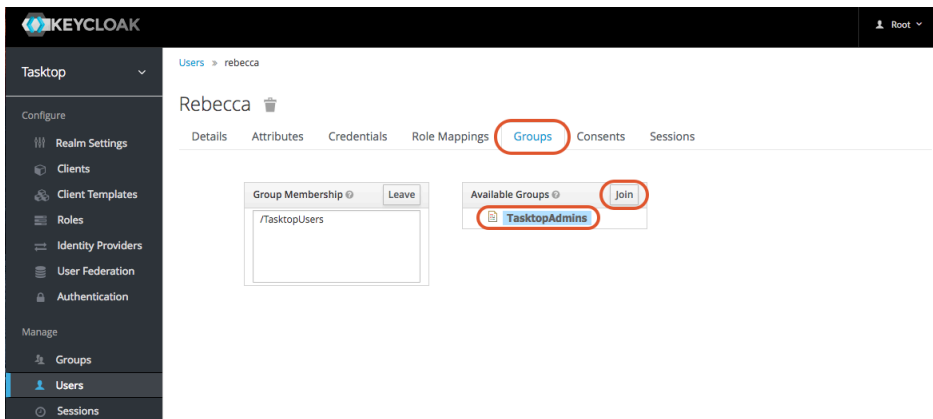
On the Add User screen, populate the Username, E-mail, First Name, and Last Name sections. The rest of the sections can be ignored.



After clicking 'Save', select the 'Credentials' tab and give the user a temporary password. Make sure 'temporary' is set to 'on'. This will allow them to set a new password upon their first log-in. Then click 'Reset Password'.



Next, select the 'Groups' tab to assign the user as a Tasktop Admin. Highlight 'TasktopAdmins' and click 'Join'. By becoming a Tasktop Admin, this user will be able to add new users directly from the Tasktop Integration Hub interface.



Ignore the Attributes, Role Mappings, Consents and Sessions tabs.


Your Tasktop Admin user has been added.

Now, sign out of the User Administration console and go to <http://<server>:8080>. You will be able to log in with the user account you just created. Once the admin user has been created, you generally will not need to log into the User Administration Console.

















Types of Users

There are two types of users: Admins and Users

The only differences between the two user types are regarding user management. An admin can create new users, update users' passwords, and change users' group membership (from user to admin or vice-versa). A user cannot. Both user types have the same permissions with regard to Tasktop functionality (meaning that both have all permissions needed to create, modify, and run integrations).

 We recommend configuring at least two admin users. This way, if one admin forgets their password, the other admin will be able to log in and re-set the other admin user's password.

We also recommend changing the default password of the Advanced User Administration console. Please see the [Getting Started](#) section above for information on how to re-set passwords.

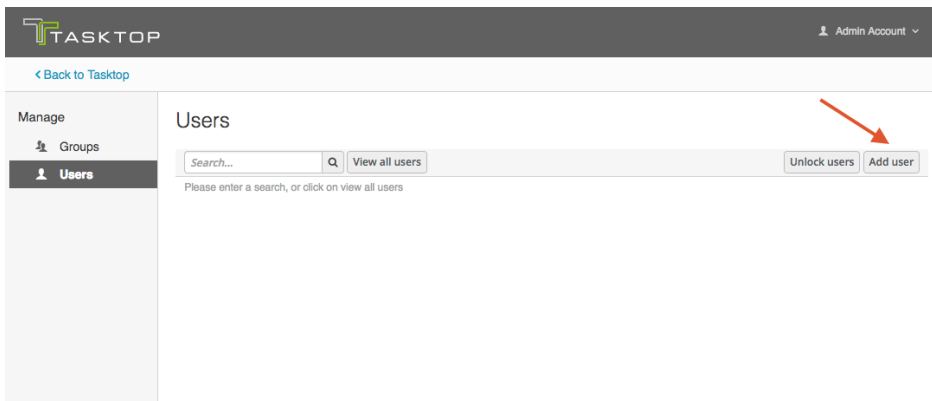
Capability	Admin	User
Create New User		
Reset Any User's Password		
View and Modify Any User's Group Membership		
Reset Own Password, Name, or E-mail		
Create and Modify Repository Connections		
Create and Modify Models		
Create and Modify Collections		
Create, Modify, and Run Integrations		

Creating Additional Users

To create an additional user, you must have admin capabilities. To create a user, select 'User Administration' from the upper right corner of the application.



From the User Administration screen, select 'Add user'



On the Add User screen, populate the Username, Email, First Name, and Last Name sections. The rest of the sections can be ignored.

Users > Add user

Add user

ID	<input type="text"/>
Created At	<input type="text"/>
Username *	<input type="text" value="NewUser"/>
Email	<input type="text" value="newuser@email.com"/>
First Name	<input type="text" value="New"/>
Last Name	<input type="text" value="User"/>
User Enabled @	<input checked="" type="checkbox"/> ON
Email Verified @	<input type="checkbox"/> OFF
Required User Actions @	<input type="text" value="Select an action..."/>
	<input type="button" value="Save"/> <input type="button" value="Cancel"/>

Click the 'Credentials' tab and give the user a temporary password. Make sure 'temporary' is set to 'on'. This will allow them to set a new password upon their first log-in. Then click 'Reset Password.'

The screenshot shows the 'Credentials' tab for a user named 'newuser'. It features a 'New Password' field with a masked password, a 'Password Confirmation' field with a masked password, and a 'Temporary' toggle switch set to 'ON'. Below these fields is a red 'Reset Password' button. At the bottom, there are two dropdown menus: 'Reset Actions' with the option 'Select an action...' and 'Reset Actions Email' with the option 'Send email'.

Click on the 'Groups' tab. Add the user to a group - either TasktopUsers or TasktopAdmins, depending on the permissions you'd like the user to have.

⚠️ If the new user is not added to a group, they will not be able to successfully access the Tasktop Integration Hub.

The screenshot shows the 'Groups' tab for a user named 'newuser'. It features two main sections: 'Group Membership' and 'Available Groups'. The 'Group Membership' section has a 'Leave' button and a list containing '/TasktopUsers'. The 'Available Groups' section has a 'Join' button and a list containing 'TasktopAdmins'.

You can ignore the following tabs: Attributes, Role Mappings, Consents, and Sessions.

Your user has been added, and can log in with their temporary password.

⚠️ Note that Tasktop will not send the new user an e-mail notification. The admin must notify the user of the new account and password.

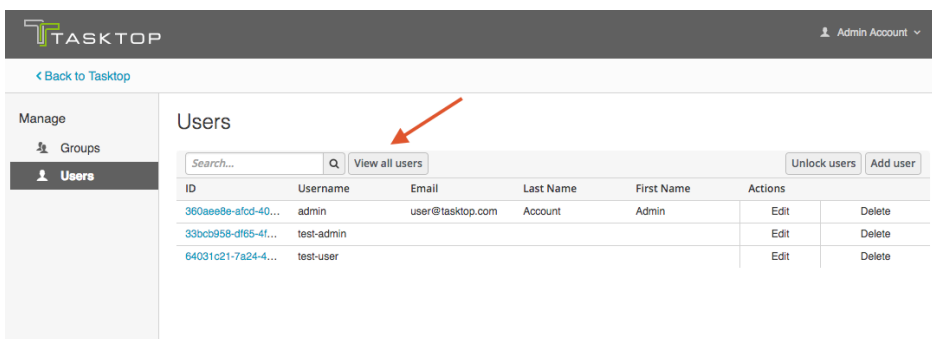
Resetting a User's Password

To re-set a user's password, you must have admin capabilities.

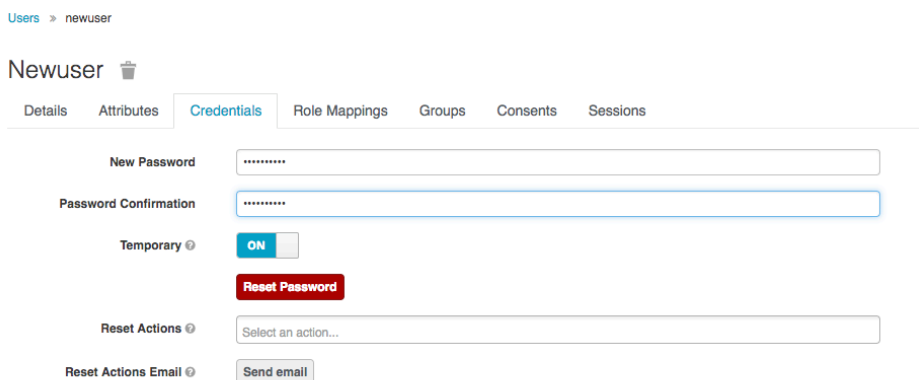
To re-set a user's password, select 'User Administration' from the upper right corner of the application.



Click 'View all Users.'



Click on the ID for the user whose password you would like to re-set. Then, click on the 'Credentials' tab and give the user a new temporary password. Make sure 'temporary' is set to 'on'. This will allow them to set a new password upon their first log-in. Then click 'Reset Password.'



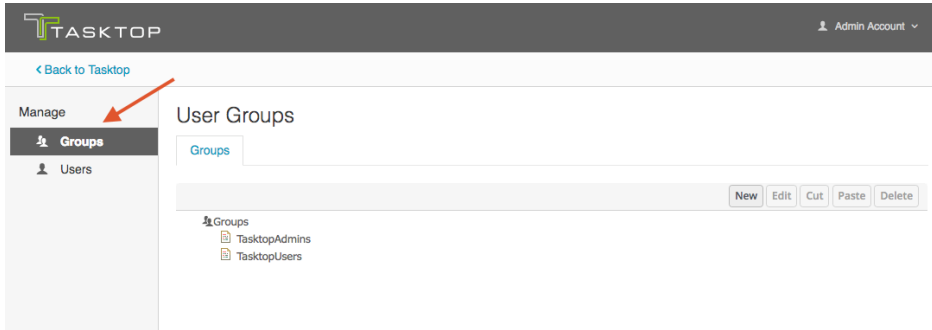
⚠ Note that Tasktop will not send the user an e-mail notification. The admin must notify the user of the new temporary password. The user will be prompted to set a new password upon their next log-in.

Managing Groups

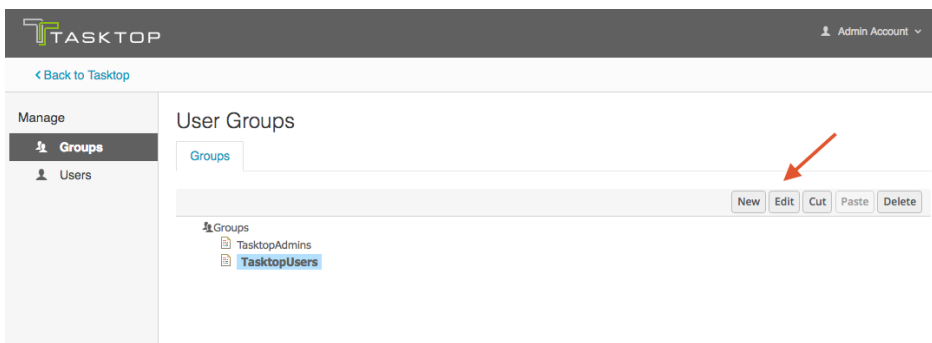
Viewing Members of a Group

To view members of a group, you must have admin capabilities.

To view the members of a group, click 'Groups' on the left pane of the User Management screen.

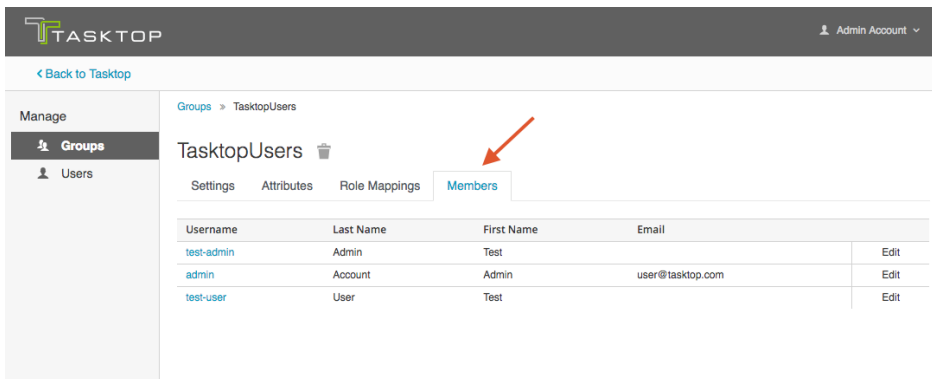


Select the group you'd like to review, and click 'edit.'



Click the 'Members' tab to view current members.

Remember that a user can be a member of multiple groups.



Adding or Removing Users From a Group

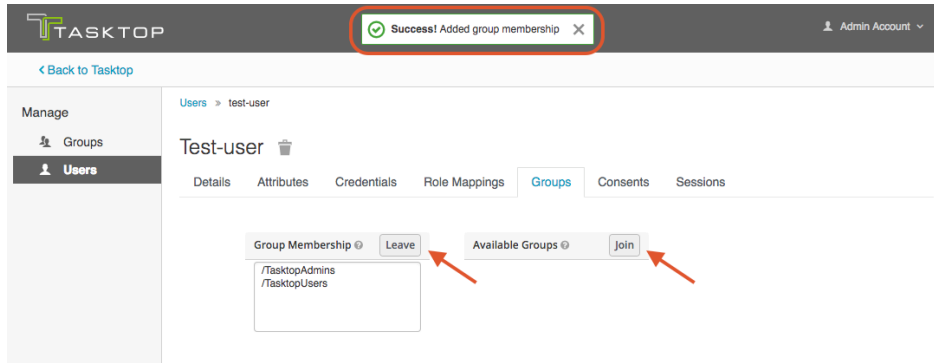
To modify a user's group membership, you must have admin capabilities.

Select 'Users' from the left pane of the User Administration screen. Click 'View all Users' and select the ID of the user you would like to modify.

Click on the 'Groups' tab, select the group whose membership you'd like to modify, and use the 'leave' and 'join' buttons to modify their group

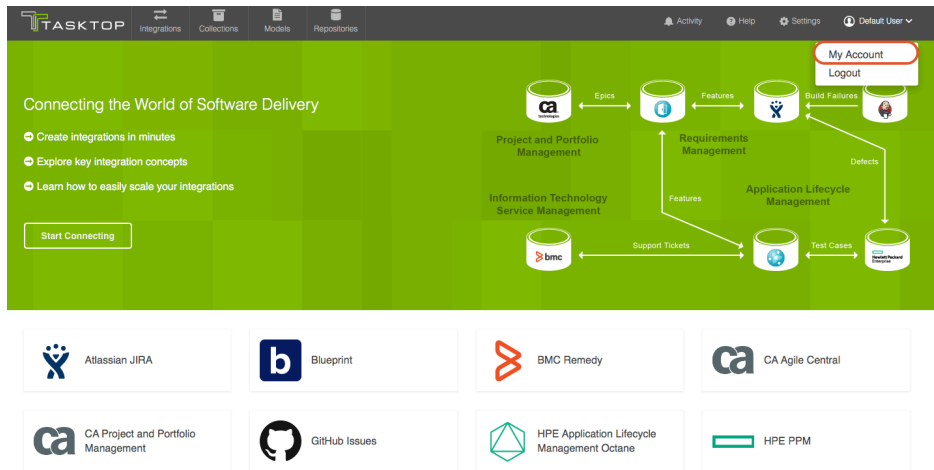
membership. There is no saving necessary here; once you click the 'leave' and/or 'join' button, you will see a notification at the top of the screen letting you know that your change has been made.

⚠️ Note that a user must be a member of at least one group in order to be able to log into Tasktop successfully.



Modifying Your Own User Information

Both Users and Admins can modify their own account information. To change your own password or other user information, right click your name at the upper right corner of the screen, and select 'My Account.'



This will bring you to the Account Info screen, where you can update your name or e-mail address:

The screenshot shows the 'Account' page in the Tasktop interface. The header includes the Tasktop logo and a 'Sign Out' link. A navigation sidebar on the left contains links for 'Account', 'Password', 'Sessions', and 'Applications'. The main content area is titled 'Account' and includes a 'Back to Tasktop' link. The form contains the following fields: Username (pre-filled with 'user'), Email (pre-filled with 'user@tasktop.com'), First name (pre-filled with 'John'), and Last name (pre-filled with 'Smith'). At the bottom right of the form are 'Cancel' and 'Save' buttons. A note 'All fields required' is visible in the top right corner of the form area.

You can also click 'Password' on the left sidebar in order to change your password:

The screenshot shows the 'Change Password' page in the Tasktop interface. The header includes the Tasktop logo and a 'Sign Out' link. A navigation sidebar on the left contains links for 'Account', 'Password', 'Sessions', and 'Applications'. The main content area is titled 'Change Password' and includes a 'Back to Tasktop' link. The form contains the following fields: Password, New Password, and Confirmation. At the bottom right of the form is a 'Save' button. A note 'All fields required' is visible in the top right corner of the form area.

The 'Sessions' and 'Applications' sections can be ignored.

Advanced User Management

Tasktop Integration Hub has some advanced user management capabilities not accessible via the Tasktop Integration Hub interface.

To access advanced user management capabilities, please click the 'User Administration Console' link at the bottom of the Tasktop Integration Hub sign-in screen.



Connecting the World of Software Delivery

Sign in to continue to Tasktop

Username

Password

Remember me

Visit the [User Administration Console](#) to add and configure users.

You can log in using the credentials you set when you first installed and began using Tasktop.

⚠ WARNING: there is only one initial root user. If the credentials for this user are lost, access to the advanced User Management features will be lost. All functionality of Tasktop Integration Hub, however, will continue uninterrupted.

Some of the advanced features include:

- User Federation Configuration for:
 - LDAP
 - Kerberos
- Identity Provider login for:
 - SAML v2.0
 - OpenID Connect v1.0
- Social Login for:
 - Google
 - LinkedIn
 - GitHub
 - Facebook
 - Twitter
 - Microsoft
 - StackOverflow
- Enforcing custom password policies such as:

- Set password expiration
- Require special characters
- Setting minimum password length

⚠ Note: While Tasktop officially supports LDAP, other advanced features (including but not limited to Keycloak, Federation, Social, and IDP) are not supported or tested by Tasktop.

To learn more about these advanced features, go to <https://keycloak.gitbooks.io/server-adminstration-guide/content/> or <http://www.keycloak.org/documentation.html>

⚠ WARNING: Do not make changes or updates to the Roles or Groups section. Altering these settings may prevent your Tasktop Integration Hub users from accessing the tool.

Configuring LDAP User Management

Required Directory Information

Before configuring LDAP, please check you have the following required pieces of information available for your specific Active Directory (AD) domain.

- The fully qualified domain name (FQDN) for the AD service,
 - *example: 'demo.tasktop.com'*
- An AD user account and credentials; The user will need read / view access to Users, Groups and Organizational Units (OU). We suggest a specific restricted account be setup in AD for this purpose.
 - *example: 'service_tasktophub'*
- An AD user group; The group(s) will be used to store specific users, who will have access to Tasktop.
 - *example: 'Tasktop Hub Users'*
- A tool such as ADSIEdit, which is able to give your the specific information about the structure of your AD domain setup.
 - ADSIEdit is part of Microsoft Windows Remote Server Administration Toolset (RSAT). This can be downloaded from [Microsoft RSAT page](#), or enabled on a server by adding the RSAT feature.
 - *Alternatively* ask your Domain Administrators for all of the following information:
 - CN/DN for Tasktop User (mentioned above)
 - CN/DN for the Tasktop User Group (mentioned above)
 - User, mail; username and name attributes (the specific name for each attribute)
 - OU root for all users
 - LDAP FQDN server URL

Accessing Keycloak Configuration Tool

1. To access advanced user management capabilities, please click the 'User Administration Console' link at the bottom of the Tasktop

Integration Hub sign-in screen.



Connecting the World of Software Delivery

Sign in to continue to Tasktop

A sign-in form with two input fields: 'Username' and 'Password'. Below the fields is a checkbox labeled 'Remember me' and a blue 'Sign In' button.

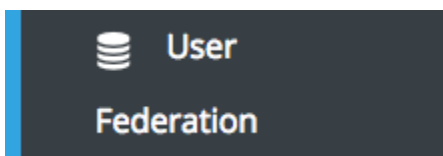
Visit the [User Administration Console](#) to add and configure users.

© Tasktop Technologies 2017

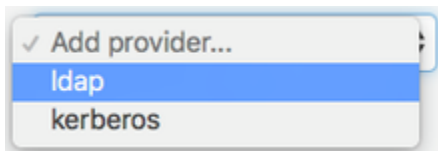
2. Log in using the default credentials listed in the [Getting Started](#) section above.

A login form with a dark background. The 'Username or email' field contains 'root' and the 'Password' field contains a series of dots. A blue 'Log In' button is at the bottom right.

3. Select the 'User Federation' link from the side-menu



4. Choose the 'ldap' option from the dropdown for 'Add provider ...'



You are now on the LDAP configuration screen.

Configuring LDAP for Active Directory

This section will guide you through creating a connection to an LDAP authentication server.

💡 Note that images provided are only a sample of settings; please ensure that you enter information specific for your environment.

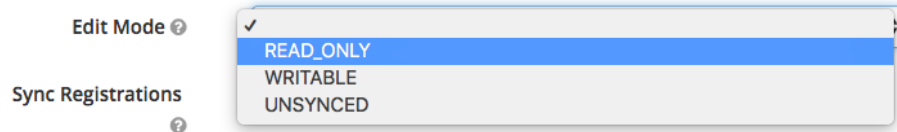
Required Settings

1. Follow steps above to access the LDAP configuration page.
2. Console Display name: This is any label you would like to give your connection.

Console Display Name  Tasktop Demo LDAP Server

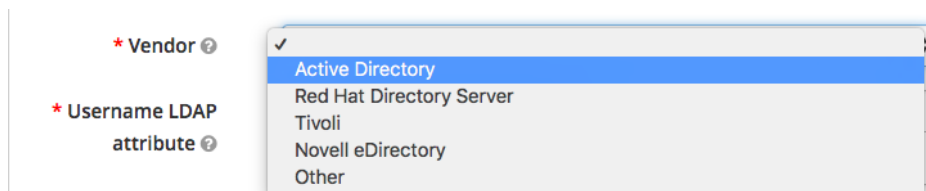
3. Priority: If you have more than a single User Federation configured, the priority specifies which order to search each user federation service, 0 is first.
4. Edit Mode:
 - READ_ONLY: This will read the attributes from Active Directory. It will not attempt to modify the AD service or store any local changes to user information.
 - WRITABLE: This may enable some changes to be written back to AD. The user account communication with AD will need access to modify the specific objects attribute
 - UNSYNCED: This will read the attributes from AD and synchronise them to a local store in the internal Keycloak database. Users and Administrators can make changes to the user objects, but those changes will only be stored for the local Tasktop instance. This will not write back to Active Directory.

The recommend mode is READ_ONLY.



5. Sync registrations: If a new user is created in Tasktop, this will allow that user to also be created in AD, if you have WRITABLE selected and access to create user objects in the AD domain. The default setting is 'OFF'.

6. Vendor: Specify which vendor software to use for this LDAP configuration. If you are using something other than Active Directory, then the attributes and locations may be different. This will also pre-fill some default values.



7. Username LDAP attribute: This should be the default username attribute as specified in your domain. The default for Microsoft AD is 'sAMAccountName'.

* Username LDAP
attribute ⓘ

sAMAccountName

8. RDN LDAP attribute: This is the Relative Distinguished Name LDAP attribute. This is a list of attributes which will be searched when a user attempts to authenticate to Tasktop. The attributes listed here should be unique within an OU level or better-yet unique within a domain. The following options are a good base to use:

- cn (conical name), also known as the full name; *example "John Doe"*
- sAMAccountName, also known as the username; *example john.doe*
- mail, also known as email-address; *example john.doe@demo.tasktop.com*

* RDN LDAP
attribute ⓘ

cn,sAMAccountName,mail

9. UUID LDAP attribute: This is the User Unique IDentification attribute. It is a complicated long string of characters which will always uniquely identify a single object within AD. For unix based LDAP this is often 'uid'. The default for Microsoft AD is 'objectGUID'.

* UUID LDAP
attribute ⓘ

objectGUID

10. User Object Classes: These are the 'types' of objects which can be used to authentication against. You can specify more if your organization has other specific identifiers such as 'staff' or 'contractor'. The default for Microsoft AD is: person, organizationalPerson, user.

* User Object Classes ⓘ

person, organizationalPerson, user

11. Connection URL: This is the specific string which should be the FQDN of your LDAP service. It's default format for AD will be 'ldap://demo.tasktop.com'. If you have SSL configured then you can also use ldaps://demo.tasktop.com (SSL is not enabled by default in Microsoft AD).

* Connection URL ⓘ

ldap://demo.tasktop.com

At this point, we recommend selecting the 'Test connection'

Test connection

button to check that Tasktop is able to communicate with your LDAP server. You should see a green message at the top of your screen indicating a successful connection to your LDAP server

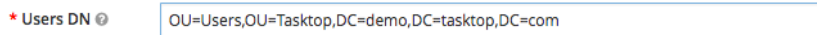
Success! LDAP connection successful. X

12. Users DN: This is the Distinguished Name for the location where you can find your users. You can find out the Users DN (and any other Distinguished Names via the ADSIEdit tool in Windows. Once the tool is open, you will need to connect to the AD domain for your company. Once connected, the domain will be presented in a tree-view on the left, where you can drill down to the specific branches until you find the

specific OU or User object you want details for. We recommend using this utility as it will allow you to copy/paste the specific DN information directly (as typing mistakes will result in error when testing).

The format for this string will be a number of 'OU=' followed by a number of 'DC=' separated by a comma. Spaces are allowed in this string if they exist in your structure.

example: OU=Users,OU=Tasktop,DC=demo,DC=tasktop,DC=com



13. Authentication Type: If you are using Microsoft Active Directory, you will be required to authenticate. Some non-Microsoft systems do not require authentication. If that is the cause for your LDAP, then select 'none'

14. Bind DN: This is the Distinguished Name for the user account which you will use to authenticate against your LDAP service in order to allow Tasktop to authenticate users. The Bind DN user account can be anywhere within the AD domain, however we suggest that you have a dedicated account specifically for Tasktop. The format for this string will be a singular 'CN=' for the Conical Name of the user account, followed by possible 'OU=' which is followed by the 'DC=' items all separated by a comma. Spaces are allowed in this string if they exist in your structure

example: CN=service_tasktophub,OU=Service Accounts,OU=Tasktop Infrastructure,DC=demo,DC=tasktop,DC=com

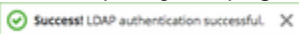


15. Bind Credential: This the password for the user account configured in the Bind DN.

Once you have entered the password, press the 'Test authentication'

Test authentication

button to confirm that Tasktop is successful in authenticating itself against your Active Directory domain. You should see a green message at the top of your page as an indication of a successful authentication



16. LDAP Filter: This is where you will configure a filter to specify which user accounts will have access to authenticate in Tasktop. If you leave this blank, all users within your 'Users DN' OU in the AD environment will have access. The structure of the string is as follows:

- () : braces to start and finish
- Either
 - &() : for performing an 'AND' operation (i.e. all items must match)
 - |() : for performing an 'OR' operation (i.e. where any items

can match)

- Specific attribute related condition, for examples matching objects in a group
- Users in a specific group you can user "memberOf=" =>
 - `memberOf=CN=Tasktop Hub Users,OU=Resource Groups,OU=Groups,OU=Tasktop,DC=demo,DC=tasktop,DC=com`
- Users and (nested) Groups in a specific group, you specifically require "memberOf:1.2.840.113556.1.4.1941:=" "
 - `memberOf:1.2.840.113556.1.4.1941:=CN=Tasktop Hub Users,OU=Resource Groups,OU=Groups,OU=Tasktop,DC=demo,DC=tasktop,DC=com`
- You can also specify that a particulate attribute is equal to some value, example
 - `objectCategory=Person`

Custom User LDAP
Filter ⓘ

`(&(memberOf:1.2.840.113556.1.4.1941:=CN=Tasktop Hub Users,OU=Re`

17. Search Scope: The Configuration of this depends on whether you have all of your AD users in a single OU, or if you would like to search through the OU hierarchy structure. If searching, then the Users DN field configured above will need to be the root or lowest-level OU.

- If all users are in a single OU, set this to 'One Level'
- If users are hierarchically organized in OUs, set this to 'Subtree'

Search Scope ⓘ

✓ One Level
Subtree

18. Use Trusted SPI: This is used if your environment uses SSL and a client certificate is required. This is not a default AD configuration.

19. Connection Pooling: This will allow connections to your AD server to remain open if set to 'ON'

ON

(for specific timeframe) rather than creating a new connection each time a user authentications.

20. Pagination: This allows you to page (or cache) information for active connections from your AD servers.

Kerberos

Kerberos setup is not shown in this guide.

Sync Settings



1. Batch Size: Indicates how many accounts will process at once
2. Periodic Full Sync: Allows for a sync of all users to occur between Tasktop and Active Directory. If you have a large number of users constantly authenticating into Tasktop, it may be useful to enable this. Default is set to OFF.
3. Periodic Changed Users Sync: Allows for newly created or

updated users to be synced from Active Directory to Tasktop. If you have the Periodic Full Sync enabled, then you should also enable this. Default is set to OFF.

Save your configuration using the save button


Save

at the bottom of the page. A green message at the top will indicate that your save was successful.

 **Success!** The provider has been created. 

Additional LDAP Information

Testing

 **Note:** The configuration utility for LDAP requires its own internal authentication. As such, when you test account access, it is recommended that you use a separate browser or select a 'private' or 'incognito' browser mode. If you are already logged into Tasktop, you will first need to logout before testing.

1. Direct your browser to the default web address of your Tasktop server, such as <https://demo.tasktop.com/>
2. Enter credentials which should be allowed access to authenticate from the LDAP connection you have just setup
3. Retry with a set of credentials which should not have access to Tasktop. If you are able to login then check the 'filter' settings again.

Default User Access

By default, all LDAP users will be granted 'user' level access to Tasktop. If desired, you will be able to set all new accounts, including LDAP user accounts, to default into a specific group. You can also assign different 'members' to either of the TasktopUsers or TasktopAdmins groups.

To change the default group, follow these instructions:

1. Select 'Groups' (under the 'manage' section) of the right-side bar menu
2. Select the 'Default Groups' tab
3. Add or Remove the TasktopUsers and / or TasktopAdmins groups to the Default Groups list.

User Management and SSL

By default, Tasktop with User Management is configured to require HTTPS so that user credentials are transmitted securely. Any attempts to connect using HTTP will be redirected to use HTTPS. If you wish to disable this behavior and allow insecure connections (this is not recommended), you can remove the following from `tasktop/container/webapps/ROOT/WEB-INF/web.xml`:

```
<user-data-constraint>  
<transport-guarantee>CONFIDENTIAL</transport-  
guarantee>  
</user-data-constraint>
```

User Management and Security Constraints

Tasktop with User Management uses Security Constraints as described in the Java Servlet Specification to limit access to authenticated users. Adding additional Security Constraints to the Apache Tomcat configuration can interfere with the Security Constraints provided by Tasktop and enable unauthenticated users to access Tasktop.

DNS Settings

The server Tasktop is installed on must be able to resolve the hostname clients will use to access it. This can be accomplished through the DNS configuration. A less preferred option is to configure using the server's hosts file.

Alternative User Management

By default, Tasktop comes with a user management solution. In the rare scenario where your company has decided to not use Tasktop's provided user management solution and you still need to ensure that only authorized users are able to access your Tasktop instance, you can set up Basic Authentication for the Tomcat web server.

Instructions for configuring Tomcat authentication can be found here: <http://www.avajava.com/tutorials/lessons/how-do-i-use-basic-authentication-with-tomcat.html>.

Please note, using this style of user management will mean that all of your users will have the exact same permissions within Tasktop. There will be no separate roles or permissions within the application.

Key Concepts

Tasktop: 17.4 Release

Tasktop is a powerful tool for connecting your software delivery systems to empower teams, enhance communication, and improve the process of software development as a whole. Below is a look at some of the concepts Tasktop utilizes to facilitate integration.

- Integration
- Repository
- Artifact
 - Work Item
 - Container
- Collection
 - Repository

The key concepts to understand are:

-  Integration

-  Repository

-  Artifact

-  Collection


-  Model

-  Flow Specification

-  Template

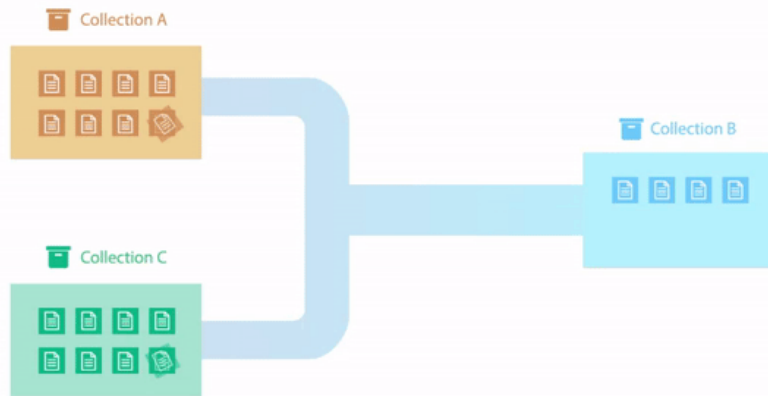
- Collection
 - Gateway Collection
- Model
- Flow Specification and Templates
 - Integration Style
 - Canvas Layout
- Artifact Relationship Management (ARM)

You can learn more about these concepts in the short video below:

 Unknown macro: 'html'

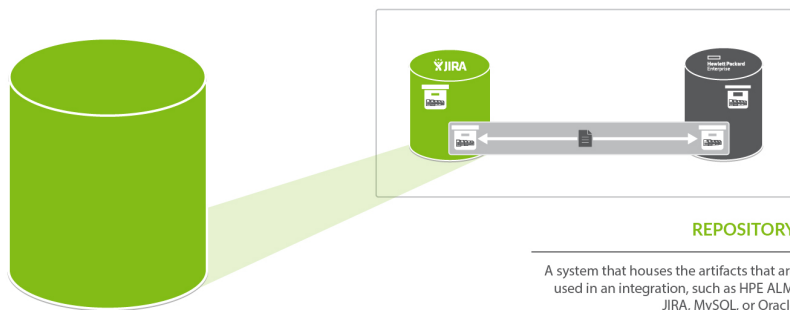
Integration

At the highest level, the definition of an integration is simply the flow of information between 2 or more systems. If you dig a little bit deeper, the definition of an integration is the flow of information, defined by the flow specification, between two or more collections. And collections are sets of artifacts. But that is probably too much to swallow right at the beginning – so don't try to! Take a look at a conceptual picture of what an integration looks like in the figure below, and just keep that in mind as we walk through all of the other concepts – then when you come back to this it will make a lot more sense!



So let's first talk about the underpinnings of how Tasktop communicates with end systems, which we call *Repositories*. For all repositories Tasktop connects to, we create what we call a *Repository Connection*. Once we've introduced those concepts we'll talk about *Artifacts* and *Collections* and then we will come back to *Integrations* and talk more about the *flow specification*.

Repository



A *repository* is any system that houses the artifacts that can be used in an integration. Repositories can be systems used as part of the software delivery process, like *HPE ALM*, *CA Agile Central*, *JIRA*, etc., or repositories can be more generic databases, like *MySQL* or *Oracle*.

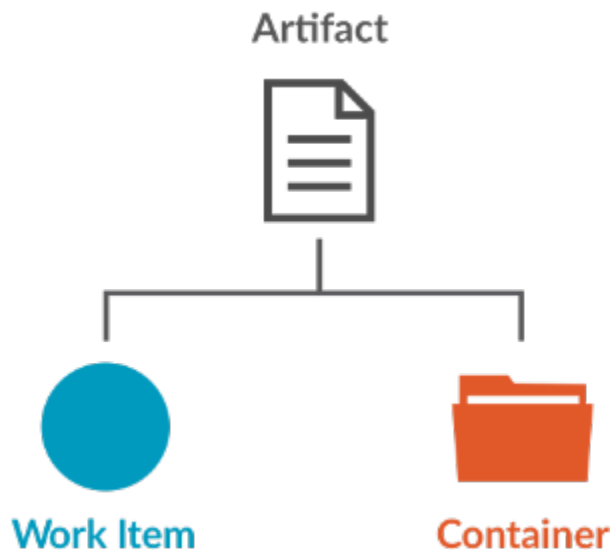
A *repository connection* is a connection to a specific instance of a given repository that permits Tasktop to communicate with that repository. To configure a *repository connection*, users will need to provide base credentials such as a server URL, a username, and a password.

You can learn how to set up a *repository connection* [here](#).

Artifact

An *artifact* is any object containing metadata that resides within your repository. There are two main types of artifacts: *work items* and *contain*

ers. Work Items and Containers have some similarities, and some key differences, with regard to how they behave within Tasktop Integration Hub.

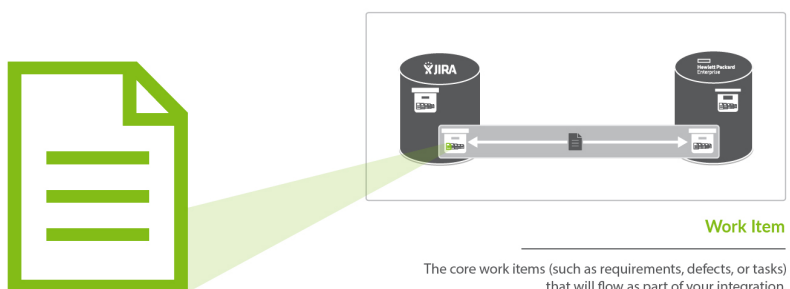


Work Item

Some examples of common work items are defects, stories, requirements, test cases, and help tickets, to name just a few. *Work Items* are the artifacts that are produced by different teams during software development. They are the core items that will flow as part of your integration. Serving as the core currency of communication, work items are the means by which all the work around software production is recorded and tracked. Work Items are at the core of any integration and are the entities that Tasktop can create or modify as a part of an integration.

Within Tasktop, you will primarily use work items to:

- Serve as the entity that flows from one repository to the other as part of your integration. For example, you can flow requirements in your source repository to your target repository, where they will create corresponding requirements.

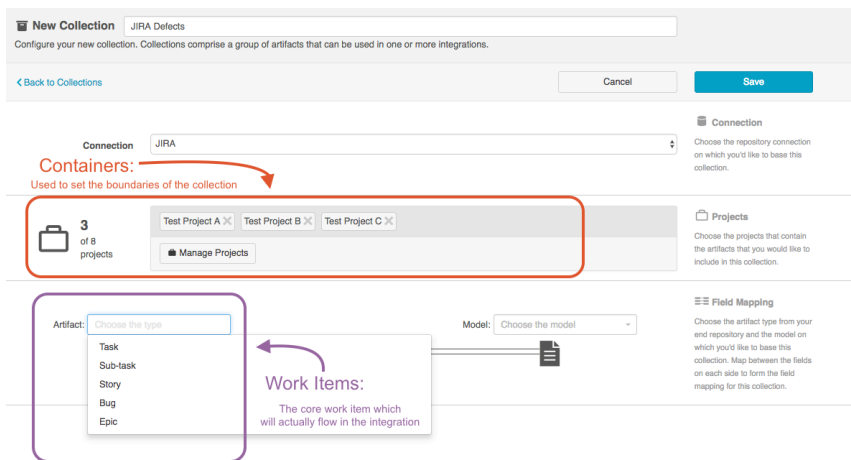


Container

Some examples of common containers are projects, folders, modules, workspaces, and sets. *Containers* are artifacts that are used to group work items. They define where, within the repository, each work item resides. The main purpose of a container is to define a set of work items.

Within Tasktop, you primarily use containers to:

- Define the scope of your collection. For example, you could add Project A and Project B to your collection, which would mean only artifacts within those projects would be eligible to flow in your integration (we'll explain this more in the 'Collection' section, below).
- Define routing for your collection. Routing defines *where* artifacts will be created within your target collection. For example, if you route Project A in JIRA to Project B in HPE ALM, that will tell Tasktop to flow artifacts in Project A in JIRA over to Project B in HPE ALM, where they will create corresponding artifacts.



High-Level Containers vs. Low-Level Containers

Some repositories contain *high level containers*, such as workspaces, which are then broken into *low level containers*, such as projects.

Container types



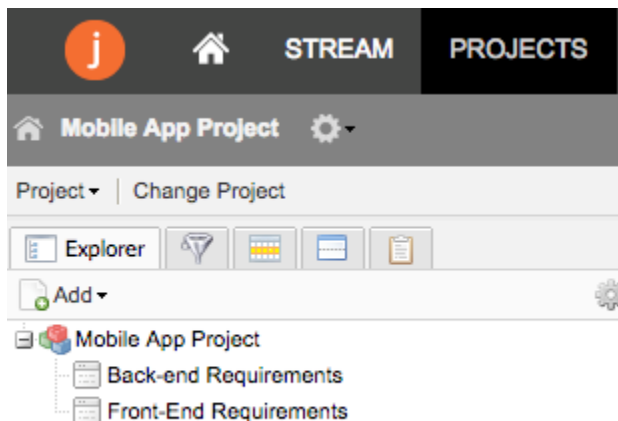
Containers are a key component of creating your collection, as each collection is defined by its artifact type (i.e. defect, requirement, test case, etc), by the model it is mapped to, and by the *high level containers* it includes. In this way, containers are essential for how you define which artifacts can flow as part of your integration.

You can learn more about how to select the containers included in your collection [here](#).

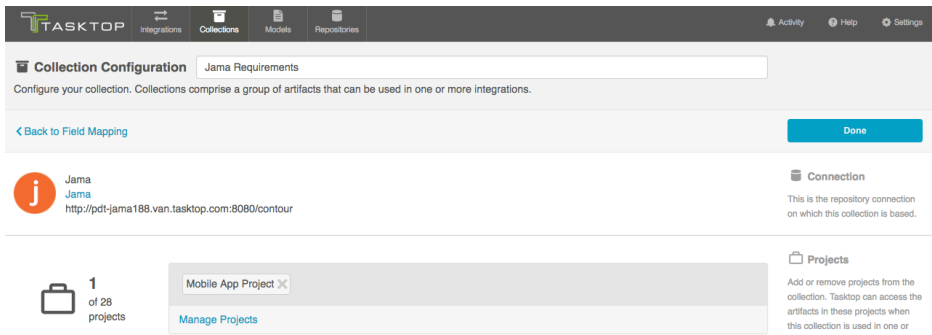
Your containers also become important during the Artifact Routing stage of configuring your integration. On the Artifact Routing Screen, you are able to determine how artifacts should flow from one collection's containers to the other's. Some repositories allow you to route at only the *low level container* level, some allow you to route at the *high level container* level, and others allow a mixed approach.

You can learn more about how to configure artifact routing [here](#).

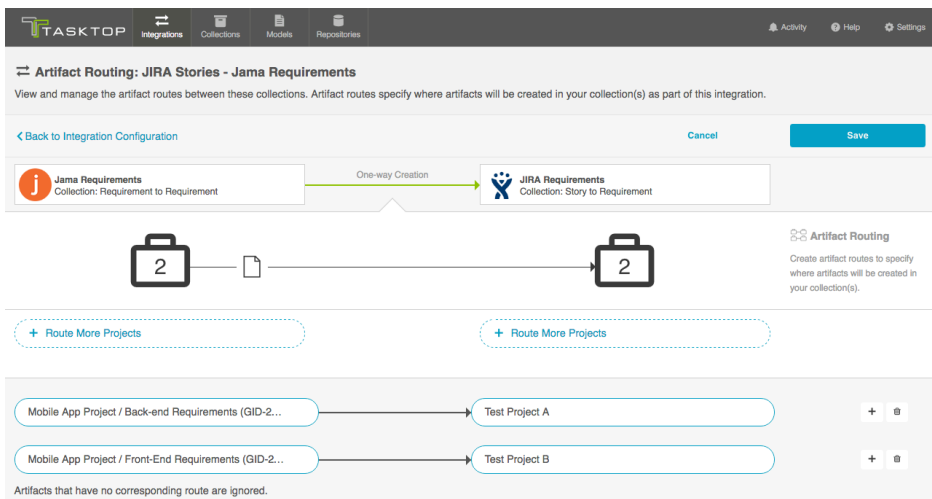
To understand this better, let's look at an example in Jama. Jama contains *high-level containers* (projects) which are then divided into several *low-level containers* (sets), which contain *work items* (requirements, in this case). Here, our *high-level container* is the Mobile App Project, which is then divided into two *low-level containers*: the Back-End Requirements set and the Front-End Requirements set.



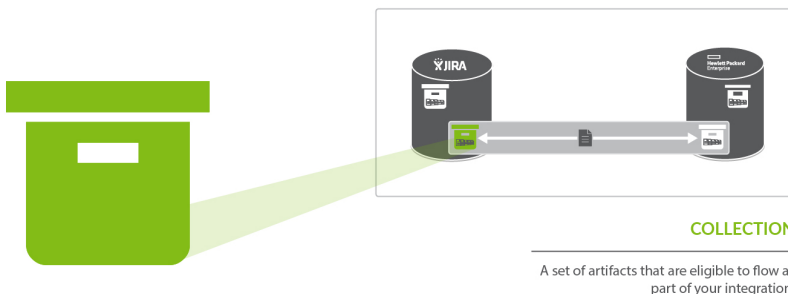
When we configure our Jama collection, we will define that collection at the *high-level container* level: this means that we can define the collection based on projects. Here, we have selected the Mobile App Project for use in our collection.



However, when routing artifacts, we will utilize *low-level containers* (sets) to determine which container Jama artifacts will flow to in our target repository. In the example below, the Back-end Requirements set in Jama will flow to Test Project A in JIRA, and the Front-End Requirements set in Jama will flow to Test Project B in JIRA. Both the Front End Requirements set and the Back End Requirements set are contained within the high level Mobile App Project.



Collection

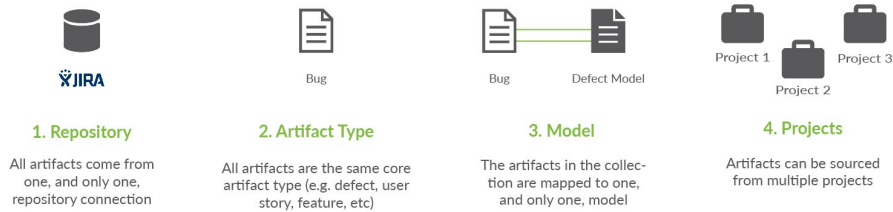


A *collection* is the set of artifacts that are eligible to flow as part of your integration. They have the following characteristics:

1. All artifacts in the collection are the same core artifact type (e.g.

defect, user story, feature, etc)

2. The artifacts in the collection are mapped to one model
3. Artifacts can be sourced from multiple projects (containers)



A concrete example of a collection would be a set of defects from an organization's *JIRA* instance.

The artifacts in a collection can come from one or more projects from a given repository connection. Getting back to the example provided, if your *JIRA* instance had 50 projects, you could include artifacts from any or all of those projects. Once projects are added to a collection, those artifacts are eligible for inclusion in an integration.

(Note: The term "project" is used here generically— sometimes repositories have different names for "project", or may not have more granular projects at all, but let's stick with this for simplicity's sake.)

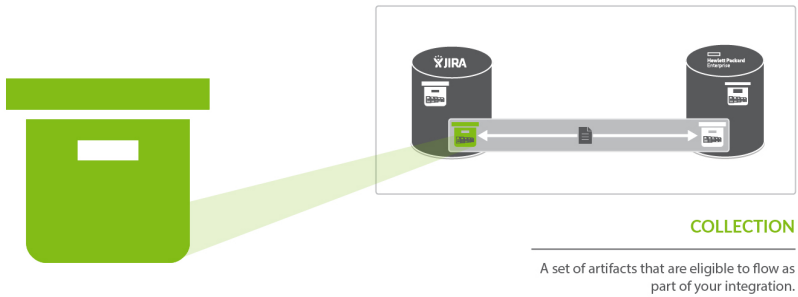
The artifacts in a collection share a set of fields that have repository-specific names and values. Part of creating a collection involves choosing a model on which to base the collection and then mapping these repository specific fields and values to those defined in the model. The concept of models will be discussed in the next section.

There are two types of collections in Tasktop: *Repository Collections* (which include collections from typical repositories, such as *JIRA* or *HPE Octane*, as well as *Database Collections*, which connect to databases such as *MySQL*) and *Gateway Collections*.

You can learn how to create your collection(s) [here](#).

Repository Collection

Standard Repository Collections

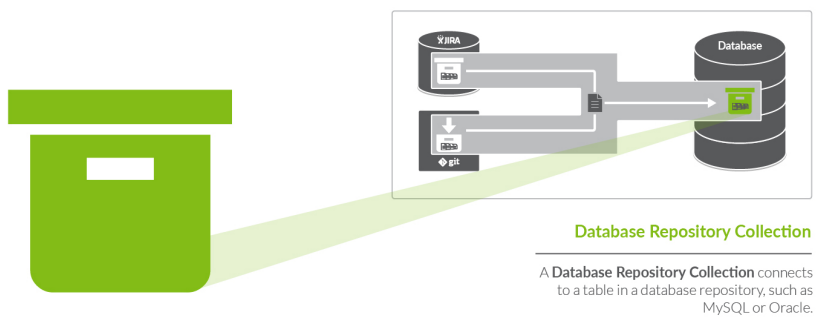


Standard Repository Collections comprise artifacts from an ALM, PPM, or ITSM repository like *Atlassian JIRA*, *HPE ALM*, *CA Clarity*, or *Zendesk*. When used in an integration, artifacts in a repository collection can be created, can be updated, and/or can trigger the creation of artifacts in another collection.

What can Tasktop do to artifacts in a repository collection?

Action
Create artifacts in collection
Update artifacts in collection
Detect additions or updates to artifacts in collection in order to create or update artifacts in another collection

Database Collections (a type of Repository Collection)



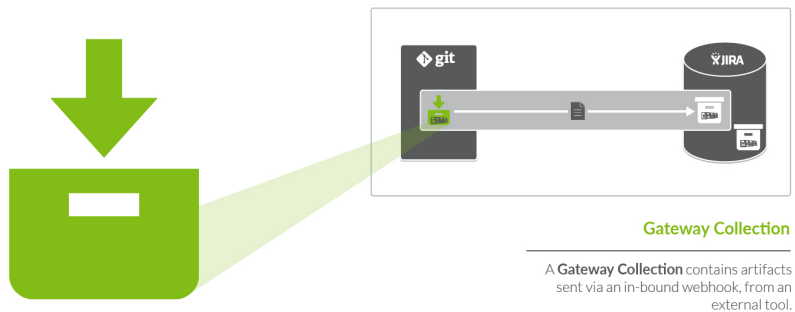
Databases collections, a type of Repository Collection, connect to a table in a database repository, such as MySQL or Oracle. Artifacts in the source repository will flow data to the fields in that table.

When used in an integration, artifacts in a database collection can be created, but cannot be updated nor trigger the creation of artifacts in another collection.

What can Tasktop do to artifacts in a database collection?

Action
Create artifacts in collection
Update artifacts in collection
Detect additions or updates to artifacts in collection in order to create or update artifacts in another collection

Gateway Collection



Unlike repository collections and database collections, which rely on Tasktop actively making various API calls to communicate with a given repository, artifacts in a Gateway collection are sent to Tasktop via our own REST API. This means that you don't need to create a repository connection to create a gateway collection--as long as you can send Tasktop a simple REST call, those artifacts can then be used to achieve a specific goal within the context of an integration.

Gateway collections are particularly useful when the artifacts you want to integrate come from smaller, purpose-built systems for practitioners in various disciplines, such as Selenium for QA; when the artifacts you want to integrate come from systems that are largely event-driven, such as an application performance monitoring repositories; when artifacts come from home-grown tools your organization might have developed on their own; or when you'd like to pull information that is not considered a standard artifact from a repository supported by Tasktop, like capacity information from a PPM tool. When creating a gateway collection, you'll specify a path to generate a webservice to which you'll post information. You'll also choose the model to which you would like incoming artifacts from this collection to conform. You'll then be given an example payload and script that can be used to send artifacts to Tasktop:

Gateway Collection Build Failures

View your existing collections and create new ones. Collections comprise a group of artifacts that can be used in one or more integrations.

[Back to Collections](#) Done

Path:

Token:

Model:

Relationship Field Configuration

Parent Artifact:

Access Details

Uri:

Method:

Content-Type:

Example Payload

```
{
  "severity": "Urgent",
  "status": "Closed",
  "summary": "String",
  "description": "String",
  "release": "1.0",
  "sprint/iteration": "Sprint 1",
  "owner/assignee": "userId",
  "priority": "Low"
}
```

Example Script

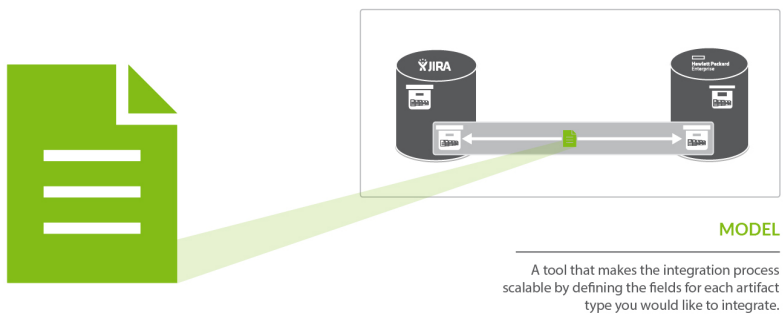
```
curl -H 'Content-Type: application/json' --data-binary '{"severity":"Urgent","status":"Closed","summary":"String","d
```

When used in an integration, artifacts in a gateway collection can trigger the creation or modification of artifacts in another collection.

What can Tasktop do to artifacts in a gateway collection?

Action
Create artifacts in collection
Update artifacts in collection
Detect additions or updates to artifacts in collection in order to create or update artifacts in another collection

Model



When integrating data from multiple collections, there are three factors that are critical to success:

1. The ability to normalize disparate definitions of artifacts between

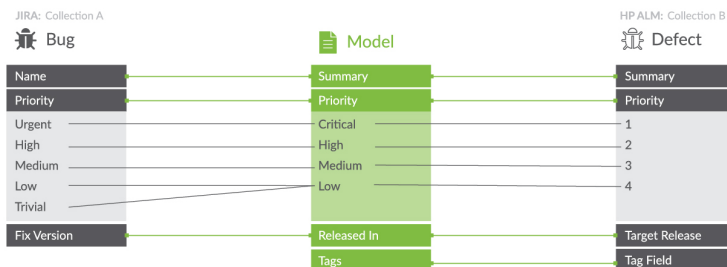
different collections

2. The ability to scale the integrations to support many collections with hundreds or even thousands of projects and artifacts.
3. Efficient flow of data – meaning, only flow information that is necessary between collections

These three critical success factors are met with our usage of "models". In very basic terms, a model is simply a list of fields or attributes that define a certain artifact that you want to integrate. For example, below is a very basic defect model:

Defect Model	
Field	Field Type
Description	String
Priority	Single Select: <ul style="list-style-type: none">• High• Medium• Low
Status	Single Select: <ul style="list-style-type: none">• New• In Progress• Complete

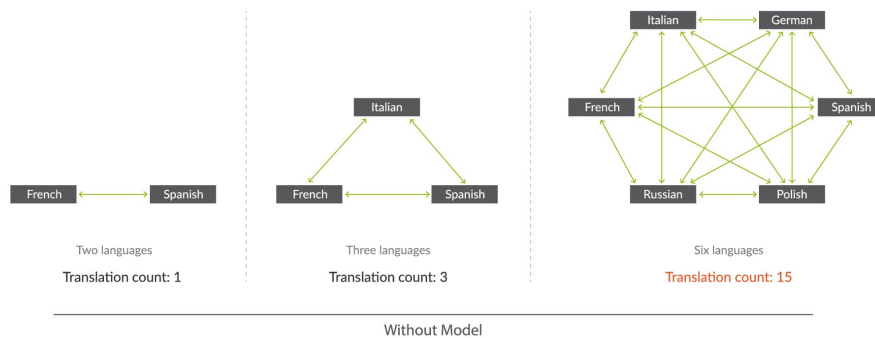
Let's talk about the first critical success factor – the ability to normalize disparate definitions of artifacts between different collections. Or, another way of thinking of it, the classic "you say tomato, I say tomahto" conundrum. In the diagram below it is apparent that the JIRA bug is similar, but not the same, as the HPE ALM defect. The solution to this problem is to be able to "map" each defect to a common definition of a defect and "normalize" the fields and field values. Then, when you are communicating about "defects", everyone is speaking the same language via the "model" definition. Like this:



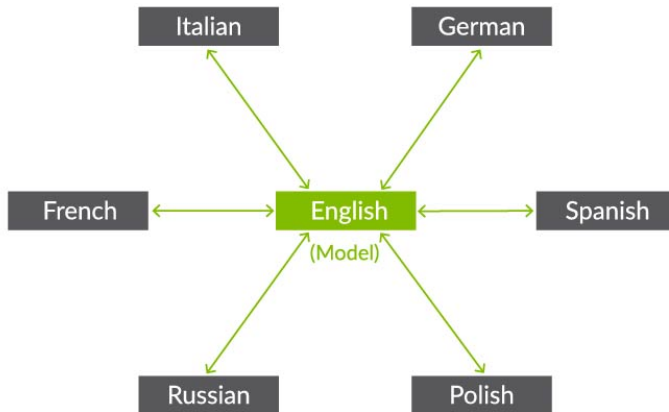
A good analogy to help understand why models are so important is the act of translating between people who speak different languages. If you have two people that speak two different languages, you need to translate only between those two points. If, however, you have three different languages, you have three points of disconnect in

communication that need to be translated. But, as you add more and more languages, the number of disconnects blocking communication does not grow linearly – even if you have just 6 languages, you have 15 points of disconnect to translate between! And if you have 10 languages you will have 45! As you can see, resolving these point-to-point disconnects individually quickly becomes unsustainable given the sheer number of them that can arise. It is in this way that models save the day, acting as a “universal translator,” overcoming all of the communication disconnects that are present by translating between all of the points at once. Now that we have the ability to solve the “*you say tomato, I say tomahto*” problem, the second critical success factor comes into play, which is the desire to *scale your integration landscape* to support many collections with hundreds or even thousands of projects and artifacts.

Integrating Without Models



Integrating With Models



Six languages

Translation count: 6

With Model

Now that we've solved the first two critical success factors, there is one more that might not seem as obvious but is actually quite important to your overall success. When flowing large volumes of data, you need *efficient flow of data*, not the 'drink from the firehose' approach where all fields of all artifacts are flowing everywhere. There is no business value in that and, worse, you will end up with significant performance issues. Instead, by using *models*, you can limit, or target, the exact data that you need to flow between collections – nothing more, and nothing less, than what is necessary.

In summary, models solve the critical three success factors for large scale integration landscapes – giving users the ultimate in flexibility, scalability, and consistency at the same time.

You can learn how to create a model [here](#).

Flow Specification and Templates

Now that we have introduced the concepts of *artifacts*, *collections*, and *models*, we can come back to the concept of an *integration*. As discussed earlier, the basic concept of an integration is the flow of information between two or more collections.

The last two concepts to introduce relate to integrations as a whole. First, the *flow specification*. This is probably the trickiest aspect of an integration, which is why we also have introduced another concept, called *templates*, to help.

Defining how you'd like data to flow between collections requires a lot of nuance and forethought. For instance, would you like to create new

artifacts, or modify existing artifacts? Would you like artifacts and fields to flow in both directions or just one direction? What types of collections (and how many of them) would you like to integrate?

Picking a template jump-starts your integration, bundling many of the flow specification elements to facilitate quicker configuration.

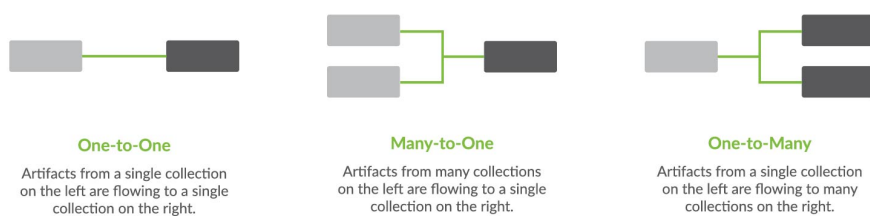
You can learn how to configure your integration using a template [here](#).

Integration Style

Each *template* is based on an underlying style that defines whether you want to *create new artifacts* in collections or *modify already existing artifacts* in collections.

Canvas Layout

Each template follows a certain canvas layout, determining the quantity and types of collections that can be added to the canvas. The canvas will either follow a many-to-one, one-to-many, or one-to-one layout.



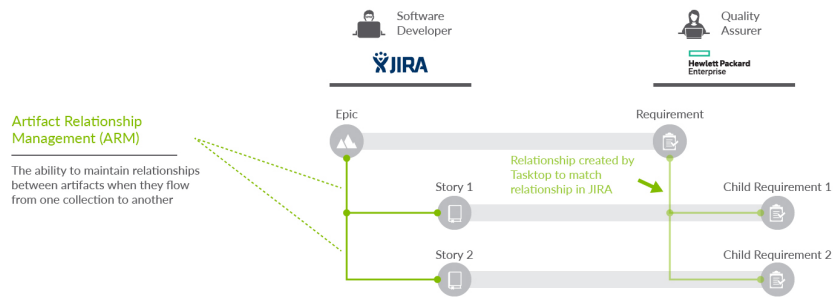
By picking a given template, you are, in essence, also picking the style of integration and canvas layout, which in turn influences other configuration options such as the artifact flow directionality, field flow directionality, and routing directionality, making the act of integrating your collections quick and painless.

Artifact Relationship Management (ARM)

Artifact Relationship Management refers to the ability to maintain relationships between artifacts when they flow from one collection to another. By utilizing the Relationship Specification Screen when configuring your collection, you can ensure that relationships are preserved between your artifacts. You'll learn more about how to configure Artifact Relationship Management in the [Quick Start Guide](#).

In the example below, you can see an example of an Integration from JIRA to HPE which utilizes Artifact Relationship Management (ARM) to do the following:

- Flow JIRA Epics to HPE Requirements
- Flow JIRA Stories to HPE Child Requirements
- Utilizes Artifact Relationship Management (ARM) to preserve the relationships between the artifacts in each repository



Quick Start Guide

Tasktop: 17.4 Release

Overview

Setting up a new integration takes four simple steps.

1. Connect to your repository
2. Create a new model or use an existing model
3. Create your collection(s) (which includes mapping your collection to the model you've picked)
4. Configure the integration using one of our out of the box templates

Finally, once you've configured your integration, you can easily expand or modify your Integration.

TASKTOP Integrations Collections Models Repositories Activity Help Settings

4 3 2 1

Connecting the World of Software Delivery

- Create integrations in minutes
- Explore key integration concepts
- Learn how to easily scale your integrations

[Start Connecting](#)

Atlassian JIRA	Blueprint	BMC Remedy	CA Agile Central
CA Project and Portfolio Management	GitHub Issues	HPE Application Lifecycle Management Octane	HPE PPM
HPE QC / ALM	IBM Rational ClearQuest	IBM Rational DOORS	IBM Rational DOORS Next Generation
IBM Rational Team Concert	IBM RequisitePro	iRise	Jama

Step 1: Connect to Your Repository

Tasktop: 17.4 Release

Types of Repositories


The first step to take when configuring an integration is to connect to your repository. Your repositories refer to the external tools that Tasktop will flow information between.


You can create two types of repository connections:

Unknown macro: 'html'

<p>Standard Repository</p>	<p>Database Repository</p>
----------------------------	----------------------------

<i>Standard Repositories are available in all Editions.</i>	<i>Database Repositories are only available in Editions that contain the Enterprise Data Stream add-on. See Tasktop Editions table to determine if your edition contains this functionality.</i>
A 'standard repository' refers to an external tool, such as HPE ALM or JIRA. These are software lifecycle tools that contain artifacts, such as defects or requirements.	A 'database repository' refers to an external database, such as MySQL or Oracle Database repositories are used as part of the Enterprise Data Stream add-on.
Learn More	Learn More

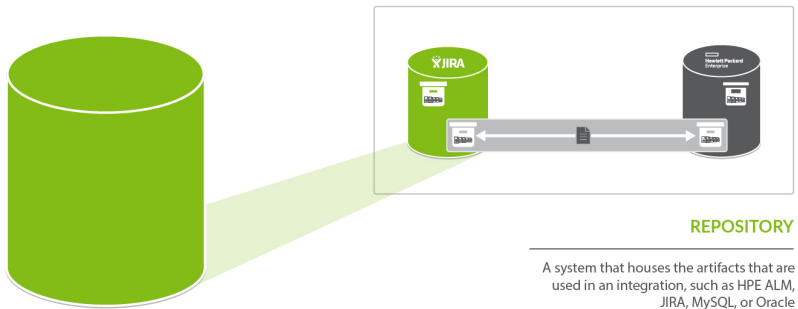
 Note: If you are creating a Gateway collection, for use with our Gateway add-on, no step needs to be taken on the Repository screen.

 Unknown macro: 'html'

Standard Repository Connection

Tasktop: 17.4 Release

What is a Repository?



A *repository* is any system that houses the artifacts that can be used in an integration. Repositories can be systems used as part of the software delivery process, like *HPE ALM*, *CA Agile Central*, *JIRA*, etc., or repositories can be more generic databases, like *MySQL* or *Oracle*.

A *repository connection* is a connection to a specific instance of a given repository that permits Tasktop to communicate with that repository. To configure a *repository connection*, users will need to provide base credentials such as a server URL, a username, and a password.

A standard repository is software lifecycle tool, such as JIRA or HPE ALM, that contain artifacts such as defects or requirements.

Video Tutorial

- What is a Repository?
- Video Tutorial
- Before You Begin
- How to Connect to a Standard Repository
 - Creating a New Connection
 - Authentication
 - Standard Authentication
 - SSO Authentication
 - HTTP POST
 - Login Form
 - Script (H

Check out the video below to learn how to create a new repository connection:

Unknown macro: 'html'

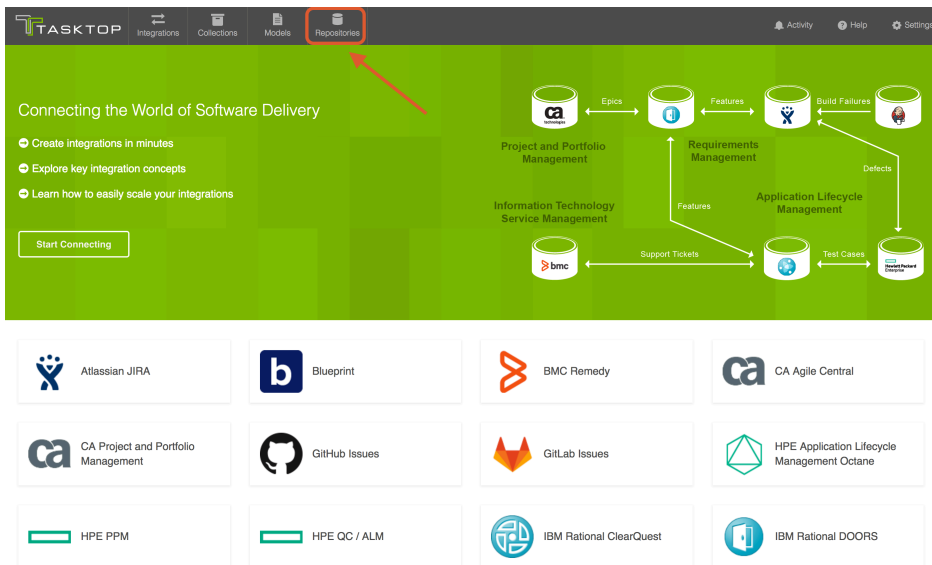
Before You Begin

- When you start up Tasktop, you will be prompted to log in. Please review the [User Management](#) section for instructions on how to log in and manage your user accounts.
- Next, you will be prompted to set a [Master Password](#), which will be used to encrypt your repository credentials.
- Before connecting to your repository, make sure that you have applied your license on the Settings screen. You can learn how to apply your license [here](#).

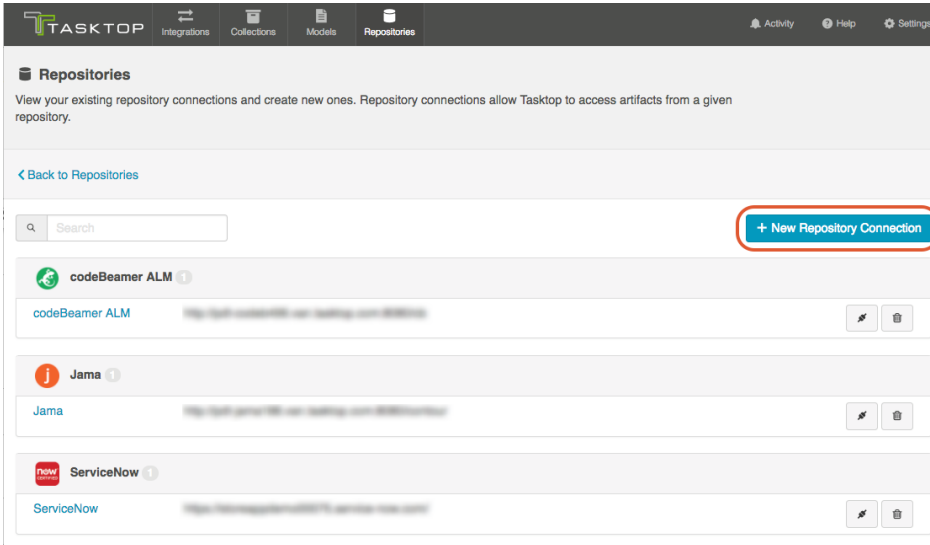
How to Connect to a Standard Repository

Creating a New Connection

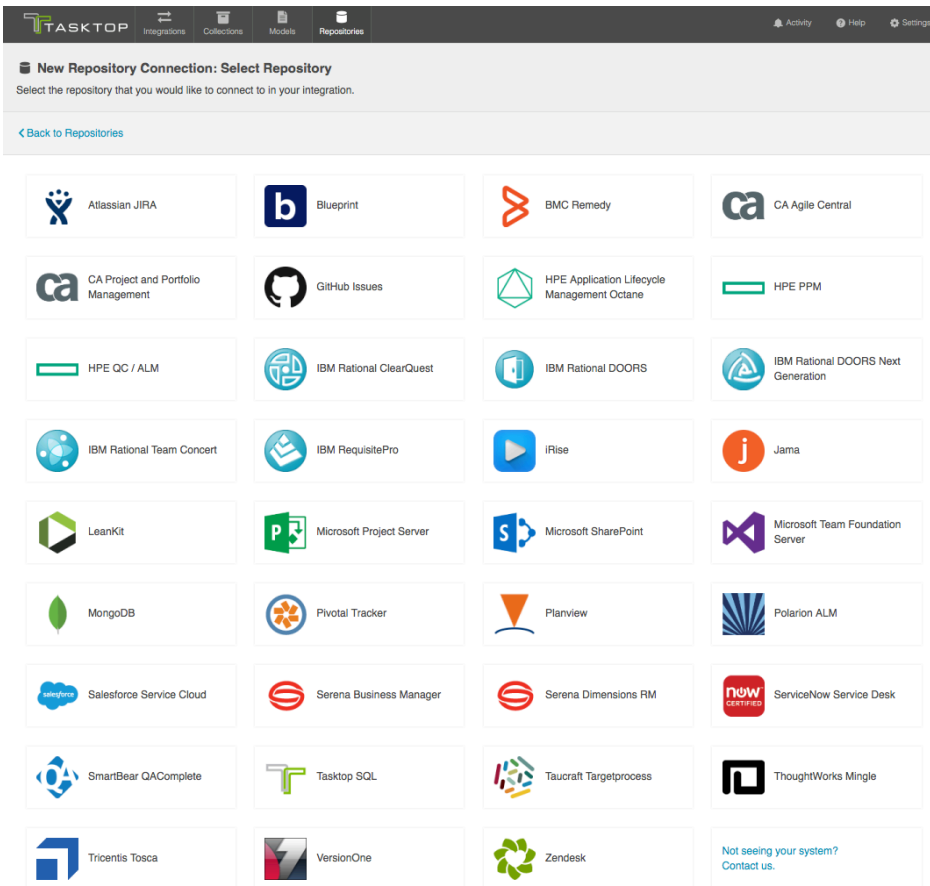
To create a repository connection, select 'Repositories' at the top of the screen



Click the '+ New Repository Connection' button



Click the logo of the repository you would like to connect to:



This will lead you to the New Repository Screen.


To connect to a repository, you must populate the following fields:

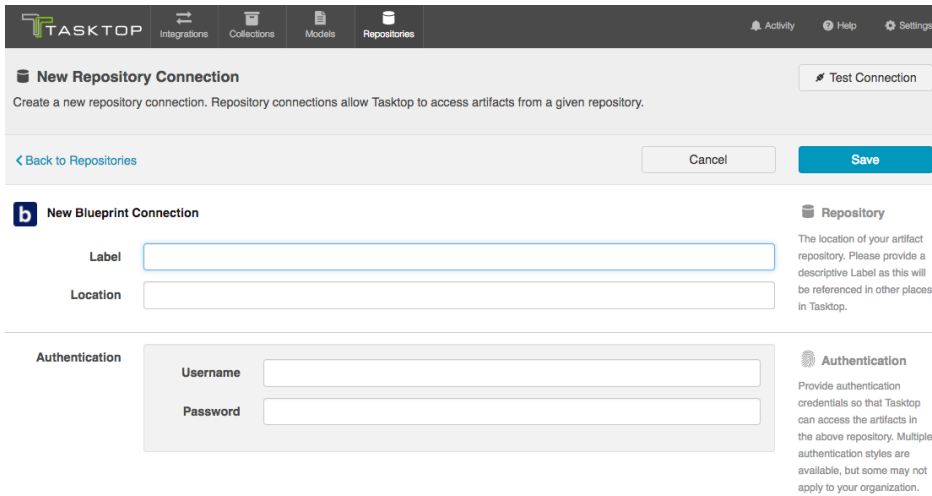
- Label: This is the name you will give to your Repository Connection. This is how it will be referenced throughout the Tasktop Application
- Location: This is the URL used to access the repository.
- Authentication Details (see authentication section below for more

cookies)
X.509 Certificate

- Proxy Server
- Additional Settings
 - Repository Query
 - Concurrency Limit
- Testing Your Repository Connection

details):

 You may see different fields depending on which repository you are connecting to. See our [Connector Documentation](#) for repository-specific information.



Tasktop Integrations Collections Models Repositories Activity Help Settings

New Repository Connection

Create a new repository connection. Repository connections allow Tasktop to access artifacts from a given repository.

[Test Connection](#)

[Back to Repositories](#)

b New Blueprint Connection

Repository
The location of your artifact repository. Please provide a descriptive Label as this will be referenced in other places in Tasktop.

Label

Location

Authentication

Username

Password

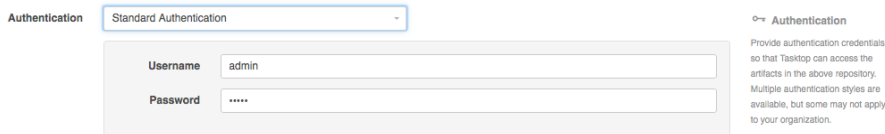
Authentication
Provide authentication credentials so that Tasktop can access the artifacts in the above repository. Multiple authentication styles are available, but some may not apply to your organization.

Authentication

For most repositories, you will see a username and password field in the Authentication section. However, some repositories include additional Authentication options.

Standard Authentication

For most scenarios, you will select 'Standard' Authentication.' This is where you will enter the username and password used to access the repository. We recommend creating login credentials specifically for Tasktop to access your repository.



Authentication

Authentication
Provide authentication credentials so that Tasktop can access the artifacts in the above repository. Multiple authentication styles are available, but some may not apply to your organization.

Username

Password

SSO Authentication

If you connect to a repository utilizing CA SSO authentication, you can select one of the additional authentication options offered.

Tasktop currently supports the following SSO implementations:

- CA Siteminder/CA Single Sign-On (HTTP POST)
- CA Siteminder/CA Single Sign-On (Login Form)
- Script (HTTP cookies)
- X.509 Certificate

HTTP POST:

The HTTP Post option, pictured below, will generate the authentication form for you to fill in. Only the first 3 fields are required.

Authentication

Standard Authentication
CA Single Sign-On (HTTP POST)
 CA Single Sign-On (Login Form)

POST target

Username

Password

Username Field Name

Password Field Name

Target URL

Authentication

Provide authentication credentials so that Tasktop can access the artifacts in the above repository. Multiple authentication styles are available, but some may not apply to your organization.

Login Form

The 'Single Sign-On (Login Form)' option, pictured below, will allow you to enter the URL for your SSO log-in form.

Authentication

CA Single Sign-On (Login Form)

Location of the login form

Form URL

Authentication

Provide authentication credentials so that Tasktop can access the artifacts in the above repository. Multiple authentication styles are available, but some may not apply to your organization.

Once the URL is entered, Tasktop will auto-generate the fields that must be populated to connect to the repository.

Authentication

CA Single Sign-On (Login Form)

Form URL

USER

PASSWORD

Authentication

Provide authentication credentials so that Tasktop can access the artifacts in the above repository. Multiple authentication styles are available, but some may not apply to your organization.

Script (HTTP cookies)

To use the *Script (HTTP cookies)* authentication method, select the script to upload from your local machine. The script will be executed by the machine that hosts Tasktop. Since Tasktop supports both Windows and Linux, you will need to ensure your script is able to be executed on the appropriate operating system. The script is stored in the Tasktop database, but is written to disk upon Tasktop startup and deleted from disk upon Tasktop shutdown.

The Cookie Script will be executed and the standard out (and standard error) must read as a \n separated list of key/value pairs themselves separated by Cookie Key/Value Delimiter (default is '='). Since Tasktop supports both Windows and Linux, you will need to ensure your script is able to be executed on the appropriate operating system: .bat for windows or shell script for Linux.

The Cookie Domain and Cookie Path arguments will then be used in the construction of a cookie for each of those key values pairs.

Authentication

Script (HTTP cookies)

Cookie Script

Cookie Key/Value Delimiter

Cookie Domain

Cookie Path

Authentication

Provide authentication credentials so that Tasktop can access the artifacts in the above repository. Multiple authentication styles are available, but some may not apply to your organization.

X.509 Certificate

To use the *X.509 Certificate* authentication method, select the X.509 Certificate to upload from your local machine. The certificate is stored in the Tasktop database, but is written to disk upon Tasktop startup and deleted from disk upon Tasktop shutdown.

Authentication: X.509 Certificate


Certificate (.p12): Choose File

Password:

Authentication
Provide authentication credentials so that Tasktop can access the artifacts in the above repository. Multiple authentication styles are available, but some may not apply to your organization.

Proxy Server

If Tasktop is installed behind a firewall, you may need to connect to external ALM repositories (e.g. hosted or cloud ALM repositories) through a proxy. To create a connection to such external ALM repositories in Tasktop, you can make Tasktop connect through your proxy by configuring the proxy settings when creating a new repository connection. It is recommended to create login credentials specifically for Tasktop on the proxy server.

 Note that the Proxy Location must be a URL in order for the proxy connection to work. If a .pac script is used in your browser, you will need to open the script and find the URL/port to enter in the Location field.

To use a proxy server, check the 'user proxy server' box and fill in your proxy details in the 'Proxy Server' section on the New Repository Screen:

Proxy Server Use proxy server


Proxy Host Address:

Username:

Password:

Proxy Server
If your organization uses a proxy server to access the above repository, please provide the proxy server credentials.

Additional Settings

 In general, it is recommended that you do not configure the Additional Settings unless you have consulted with Tasktop Support.

Additional Settings

Repository Query: Enable collections to be refined by setting a repository query

Concurrency Limit:

Additional Settings
In general, it's recommended that you do not configure the Additional Settings unless you have consulted with Tasktop Support.

Repository Query

If you plan to utilize a repository query, select the checkbox here.

⚠ Repository Queries are advanced functionality, and should only be used when you are truly unable to filter as desired using the built-in Tasktop functionality of Repositories, Collections, and Artifact Filtering. You can learn more about artifact filtering [here](#).

Concurrency Limit

In general, we recommend leaving the Concurrency Limit field blank. However, in cases where there is concern regarding high Tasktop load on a repository, a value can be set to limit how much work Tasktop can do in parallel on that repository.

⚠ Caution should be used when setting this value. Setting the value too low when there is a large number of projects configured in collections and a low "Change Detection Polling Interval" setting can potentially cause Tasktop to be unable to process artifact changes. Please consult with Tasktop Support before setting a value here.

Testing Your Repository Connection

To test your repository connection, click the 'Test Connection' button on the Repository Connection screen, or click the icon on the Repositories screen.

Repository Connection
View and configure your repository connection.

[Test Connection](#)

[Back to Repositories](#) [Done](#)

Blueprint

Label:

Location:

Repository

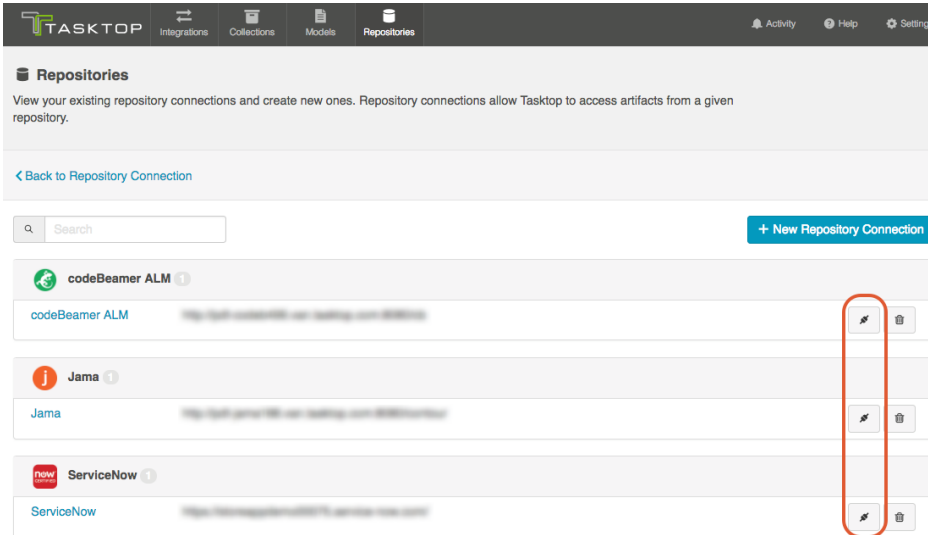
The location of your artifact repository. Please provide a descriptive Label as this will be referenced in other places in Tasktop.

Authentication

Provide authentication credentials so that Tasktop can access the artifacts in the above repository. Multiple authentication styles are available, but some may not apply to your organization.

Username:

Password:

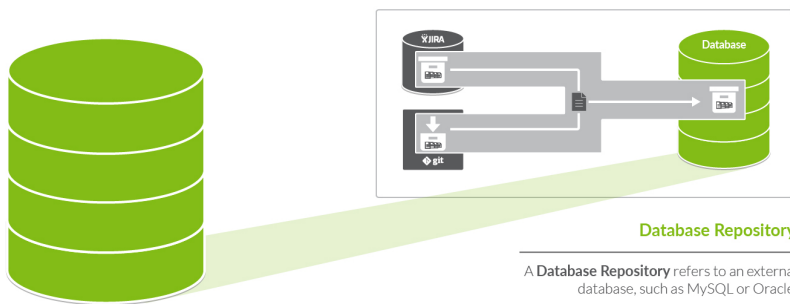


Database Repository Connection

Tasktop: 17.4 Release

What is a Database Repository Connection?

Database Connections are only available in Editions that contain the Enterprise Data Stream add-on. See [Tasktop Editions table](#) to determine if your edition contains this functionality.



A database repository, is a tool such as MySQL or Oracle, which allows you to flow data to a central database. Database repositories are used as part of the Enterprise Data Stream add-on.

In order to configure an Enterprise Data Stream Integration, you must first connect to the database that will be used by that integration. Creating a new database connection is similar to creating a [standard](#)

- What is a Database Repository Connection?
- Before You Begin
- Step 1: Download the SQL Driver
- Step 2: Upload the JDBC driver
- Step 3: Connect to your Database

[repository connection](#), with a few extra considerations. To create a new database connection, follow the steps below.

Before You Begin

- When you start up Tasktop, you will be prompted to log in. Please review the [User Management](#) section for instructions on how to log in and manage your user accounts.
- Next, you will be prompted to set a [Master Password](#), which will be used to encrypt your repository credentials.
- Before connecting to your repository, make sure that you have applied your license on the Settings screen. You can learn how to apply your license [here](#).

Step 1: Download the SQL Driver

MS SQL Server

The JDBC driver for MS SQL Server can be downloaded from the [Microsoft support site](#). The SQL Driver Location should reference the directory containing the sqljdbc42.jar file. This file should be the only .jar file in that directory, or you may end up with errors upon configuring your collection.

MySQL

The MySQL Connector/J driver can be downloaded from the [MySQL download site](#). The SQL Driver Location should reference the directory containing the mysql-connector-java-<version>-bin.jar file.

Oracle

The JDBC driver for Oracle can be downloaded from the [Oracle support site](#). Note that it is best if the Oracle JDBC driver that is used matches the version of the Oracle server that you are connecting to. Additionally, the ojdbc6.jar file is the only file that should be in the directory that is used for the SQL Driver Location or you may end up with errors upon configuring your collection.

Step 2: Upload the JDBC driver

The SQL driver files must be put on the file system of the same server where Tasktop is installed. When setting up a connection to your database with the SQL connector, the SQL Driver Location field should reference the location of the SQL driver files on the server.

MS SQL Server

The SQL Driver Location should reference the directory containing the sqljdbc42.jar file. This file should be the only .jar file in that directory, or you may end up with errors upon configuring your collection.


MySQL

The SQL Driver Location should reference the directory containing the mysql-connector-java-<version>-bin.jar file.

Oracle

The SQL Driver Location should reference the directory containing the ojdbc6.jar file. The ojdbc6.jar file should be the only file in that directory, or you may end up with errors upon configuring your collection. Note that it is best if the Oracle JDBC driver that is used matches the version of the Oracle server that you are connecting to.

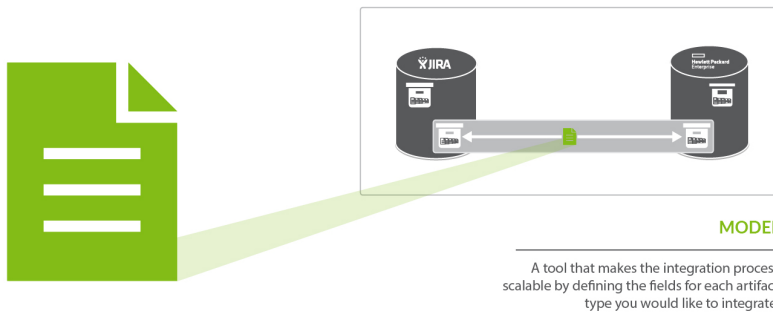
Step 3: Connect to your Database

1. In Tasktop, click 'Repositories' at the top of the screen, and click 'New Repository Connection'
2. Select 'Tasktop SQL' as the Repository type
3. Enter a label for your connection. This is how it will be referenced through the Tasktop application
4. Enter the URL of your database. The protocol should be "jdbc:sqlserver://" for a MS SQL database, "jdbc:mysql://" for a MySQL database or "jdbc:oracle://" for an Oracle database
5. Select the appropriate JDBC driver (SQL Server, MySQL or Oracle)
6. Enter the SQL driver location, which is the location of the SQL driver files on the Tasktop server. See steps 1 and 2 above for more information on the SQL driver files.
7. Enter a username and password for your database
8. If you'd like, you can test your connection by clicking the 'Test Connection' button in the upper right corner
9. In general, we recommend leaving the Concurrency Limit field blank. However, in cases where there is concern regarding high Tasktop load on a repository, a value can be set to limit how much work Tasktop can do in parallel on the repository.
 Caution should be used when setting this value. Setting the value too low when there is a large number of projects configured in collections and a low "Change Detection Polling Interval" setting can potentially cause Tasktop to be unable to process artifact changes. Please consult with Tasktop Support before setting a value here.
10. Click 'Save' and then 'Done' to save the connection

Step 2: Create or Reuse a Model

Tasktop: 17.4 Release

What is a Model?



A model is a tool that makes the integration process scalable by defining the fields for each artifact type you would like to integrate. By mapping collections to the same model, you will be able to easily add new repositories and new projects within those repositories to your integration landscape. You can learn more models in the [Key Concepts](#).

To access your models, click on the 'Models' button at the top of the screen:

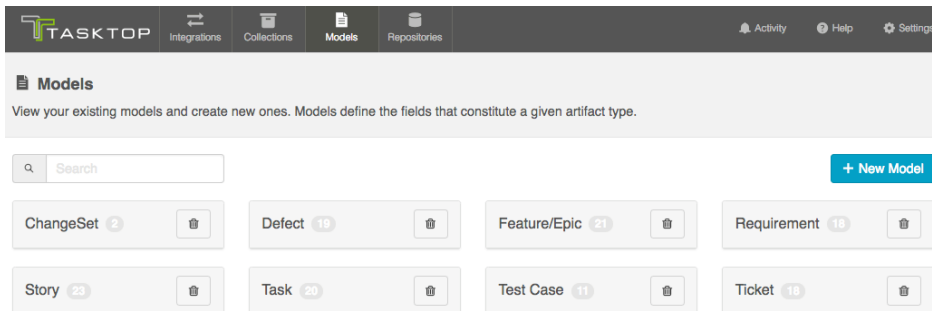
- What is a Model?
- Out of the Box Models
- Custom Models
- Add Fields to Your Model
- Smart Field Designation
- Field Label
- Field Type
 - Best Practices for Selecting a Model Field
 - Glossary of Field Types
 - Fields that Require Additional Configuration
 - Single-Select and



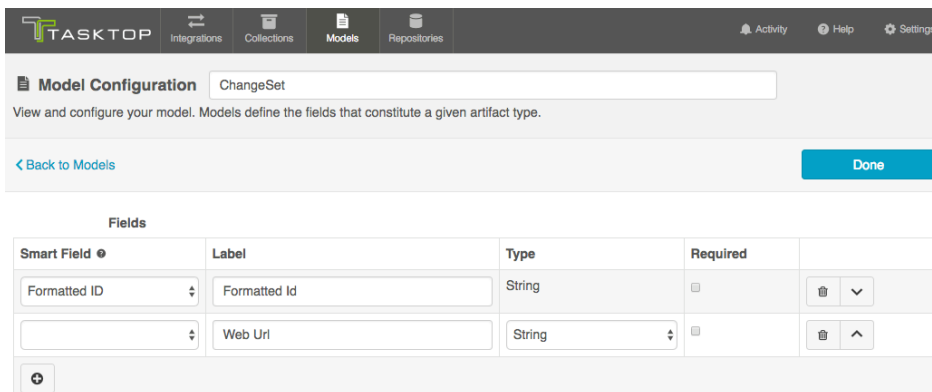
Out of the Box Models

Tasktop comes pre-packaged with several out-of-the-box models that are ready for you to use!

On the Models screen, you will see the name of each Model, with a number identifying how many fields are included in that model:




To view a model, simply click on its title. You will be brought to the Model Configuration screen, which will show the fields included in that model:

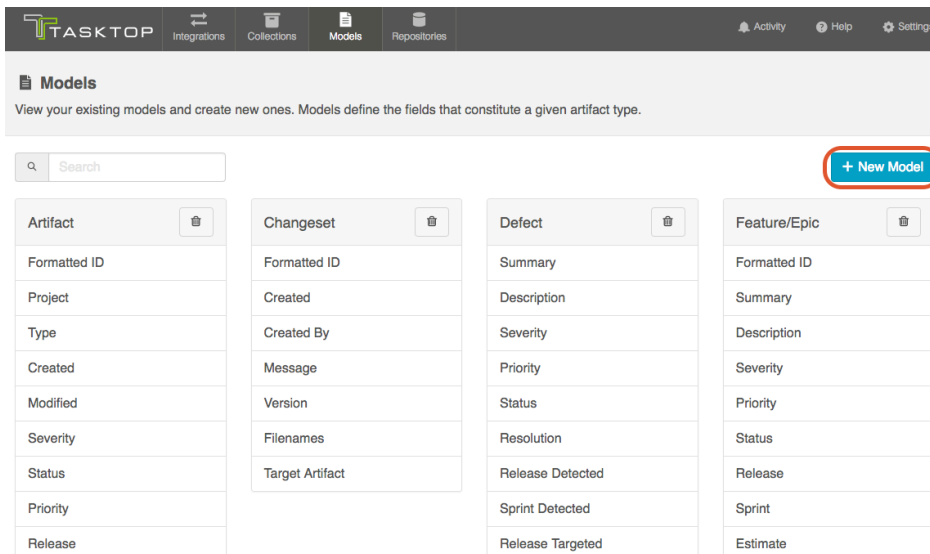


Custom Models

Check out the video below to learn how to create a new custom model:

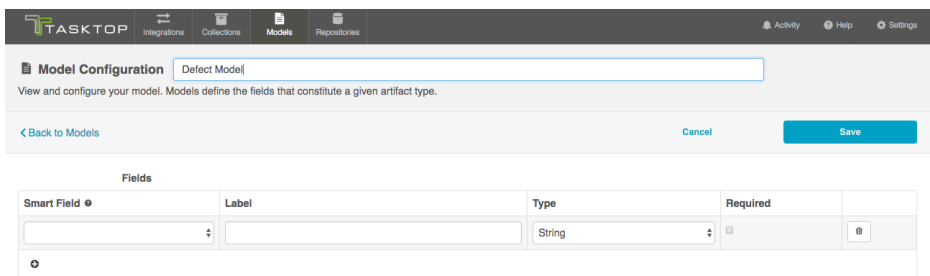
 Unknown macro: 'html'

To create a new custom model, click the '+ New Model' button at the top of the screen.



Add Fields to Your Model

You can start configuring your first model field immediately – just name it and start entering metadata into the first line. To add additional fields to your model, simply click on the plus sign at the bottom left of the model box.



Smart Field Designation

For each field you add to your model, you have the option of identifying its corresponding smart field type. *Smart fields* are a set of fields commonly available in the connectors for all of the repositories Tasktop

i-S
ele
ct

- Fields that Do Not Require Additional Configuration

- Boolean
- Date
- Date Time
- Double
- Duration
- Location
- Long
- Person and Person(s)
- Relationships and

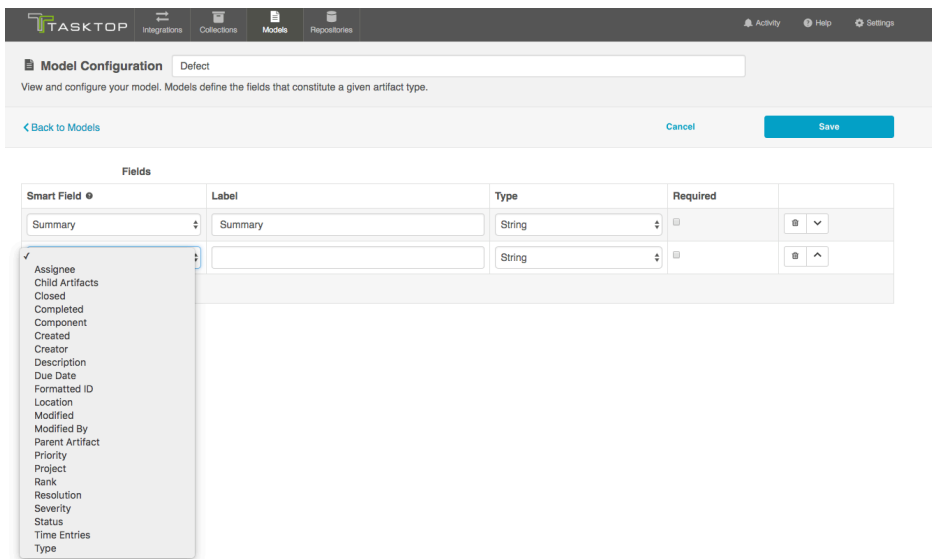
connects to. By designating a smart field to your model field, Tasktop will be able to more easily match fields from your repositories to your models while you are creating and editing collections.

Selecting a Smart Field will also give Tasktop the power to suggest the proper field type for your model field.

You do not have to select a smart field for all model fields. If you cannot find a smart field that corresponds to a model field, just leave the smart field drop down empty for that field.

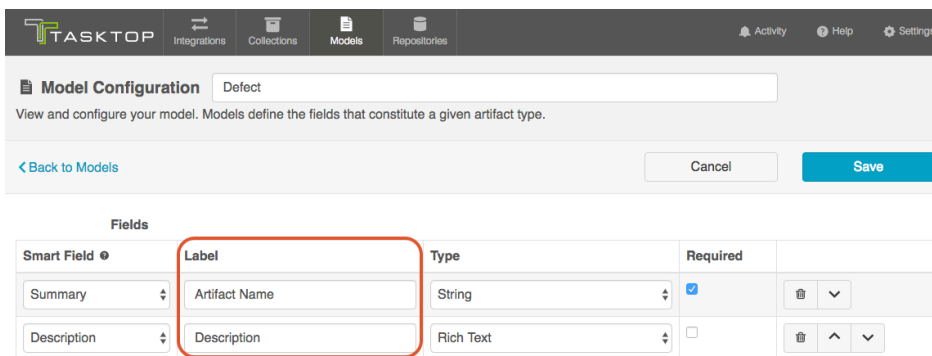
Some examples of smart fields are:

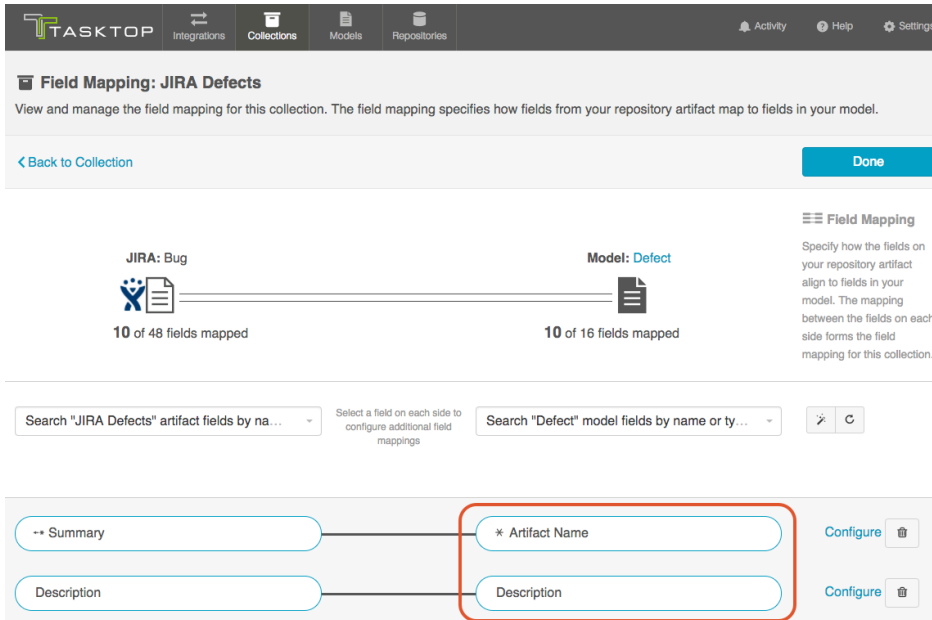
- Formatted ID: the human-readable ID of an artifact
- Location: the field that holds the URL of an artifact
- Modified: a date-time field showing when changes were last made to an artifact



Field Label

The *label* is the name of the field in your model that you will see throughout the Tasktop application, from the collection-to-model field mapping screen to the field flow screen in an integration.





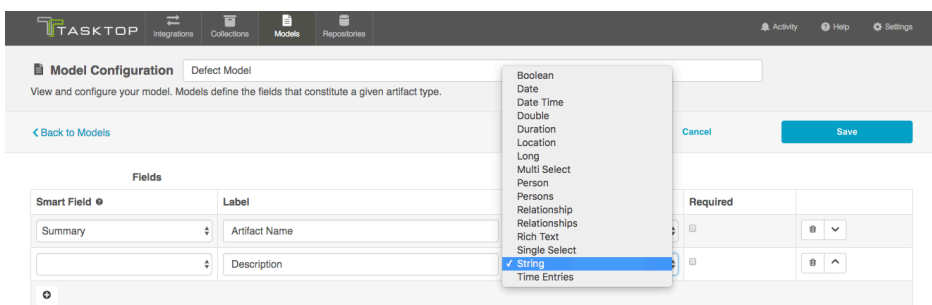
- hi p(s)
- Rich Text
- String
- Time Entries
- Web Links

- Required Designation

Field Type

Tasktop supports a number of field types, such as *string*, *multi-select*, *relationship*, and more, for use in your model. Identify the field type that most closely aligns with the type of information you expect to flow through this model field.

Review the sections below for best practices and additional configuration steps for each field type.



Best Practices for Selecting a Model Field

- The model field, by definition, sits in the middle of two fields: one from each repository you are integrating. Those two fields in your end systems may have different levels of detail, but by definition, they must map to the same Model field. We recommend that your Model field match the 'richer' of your two fields. This will ensure you preserve as much information as possible for as long as possible in your integrations. This allows your Model to be more reusable and to support more scenarios.

For example, when mapping between text fields, it's often good practice to use a Rich Text field in your Model. That way, you preserve the Rich Text from the source. If you map a Rich Text

field to a Text (string) field in the Model, you'll lose the formatting information immediately.

- If you are mapping a single- or multi-select field in your repository that contains a large look-up list (i.e. which has hundreds or thousands of possible values):
 - If the list of values match between your source and target repositories, make the model field a string field. This will allow the values to flow between the repositories without the need to maintain a field mapping.
 - If you only need to map a small sub-set of the values, make the model field a single- or multi-select field, and check 'Allow unmapped values to flow.'
- Whenever possible, utilize the smart fields available. For example, if you would like to add a 'Status' field to your model, use the 'Status' smart field, rather than entering 'Status' as the field label, and selecting a field type manually. This will enable Tasktop to auto-map the model field to the appropriate fields within each repository.
- If you would like to use a field for [artifact filtering](#), make sure to include that field in your model.

Glossary of Field Types

Fields that Require Additional Configuration

Single-Select and Multi-Select

Single-selects and multi-selects fields refer to fields in which the user selects one or many options from a list of values. These fields could refer to drop down menus, checkboxes, or radio buttons within the end repository, to name a few examples.

When utilizing single-select and multi-select fields in your model, there are a couple of additional configuration steps to be aware of.

First, click the 'Field Values' link to add values to your model. These will be the available field values that you will then map to fields within each end repository. If you'd like to add additional field values to your model, you can use the '+' button to do so.

Second, decide whether or not you'd like to allow unmapped values to flow.

If you do not allow unmapped values to flow (the default setting), the server will reject any value that is not specified in the model. In general, this is the recommended approach. If you select this approach, you will need to map all possible values for the repository field to the specific values for the model field during the [Collection-to-Model field mapping](#) step of the process.

If you do allow unmapped values to flow, field values not specified in the model will be able to flow while the integration is running. This can

make sense in a few specific scenarios, such as an Enterprise Data Stream integration or in single select to string transforms, where there are many options available and you don't desire any normalization of the data flowing through. In most cases, however, you will not want to allow unmapped values to flow.

The screenshot shows the 'Model Configuration' page for a 'Defect' model. It features a table with columns for 'Smart Field', 'Label', 'Type', 'Required', and actions. The 'Priority' field is highlighted with a red border. Below the table, the configuration for the 'Priority' field is shown, including a 'Field Values...' section with three entries: 'High', 'Medium', and 'Low', each with a trash icon and up/down arrows. At the bottom of this configuration, there is an unchecked checkbox labeled 'Allow unmapped values to flow'.

Smart Field	Label	Type	Required	
Summary	Artifact Name	String	<input checked="" type="checkbox"/>	
Description	Description	Rich Text	<input type="checkbox"/>	
Severity	Severity	Single Select	<input type="checkbox"/>	
Priority	Priority	Single Select	<input type="checkbox"/>	

Field Values...
Sev 1, Sev 2, Sev 3, ...
Allow unmapped values to flow: No

Field Values...
High
Medium
Low

Allow unmapped values to flow

In the image above, you have added 3 specific values for the field "Priority" but have not allowed unmapped values to flow, meaning that any field values sent from the collection will need to be mapped to these 3 model values in order for your artifact to flow successfully.

Fields that Do Not Require Additional Configuration

Boolean

Boolean fields are typically represented by checkboxes in the end repository. These fields are often useful for filtering integrations. As an example, you could create a custom boolean field titled "Participate in Tasktop Integration". If you filter by that field (on the [Artifact Filtering screen](#) of your integration), only artifacts that your users have checked will participate in the integration.

Date

These identify a specific date.

Date Time

These are fields that identify something more specific than a date. For example, January 1, 2017 9:35am. A 'Created' field is often a Date Time field.

Double

Use this field for number fields - either integers or decimals. For example, a double could include both values "2" and "2.5." The *Long* field

type can also be used for integers.

Duration

This field holds a length of time. This is typically used for worklogs and time estimations on tasks.

Location

This model field holds a URL.

There is also a Smart Field called Location which is specifically for the URL of a given artifact. The Location Smart Field is often used when you want to [synchronize a URL reference field to your target artifact](#). This allows for bi-directional traceability. It can also be used to report the location of an artifact in an [Enterprise Data Stream integration](#).

The 'Location' *model field type*, on the other hand, can be for any URL.

In addition to 'Location,' you will also see that there is a 'Web Links' field type available. The 'Web Links' field type includes the URL as well as additional information such as label, creator, and time of creation (depending on what the repository supports), while 'Location' includes only the URL.


Long

This field is for integer or whole numbers, only. An example of a *Long* field value is "2," but *not* "2.5." The *Double* field type can be used if you will also need to cover decimal values. Story points are a good example of a *Long* field.

Person and Person(s)

You'll notice that you are able to create both 'person' and 'persons' field types in your model. 'Person' refers to fields that contain one, and only one, Person object. Examples of this type of field are: Assignee, Owner, Reviewer, etc. Person objects contain more information than just the display name of the person. For example, they may also utilize the user's e-mail address or username in order to reconcile 'persons' between different repositories. You can learn more about person reconciliation strategies [here](#).

The Person(s) field type refers to fields that contain more than one Person. A 'Watchers' field is a good example. There can be one or more Persons in a single Watchers field.

 In general, we recommend using the 'persons' field type in your model, rather than 'person,' especially in cases where you may want to map a 'person' field in one repository to a 'persons' field in your other repository.

Relationship and Relationship(s)

You'll notice that you are able to create both 'relationship' and 'relationships' field types in your model. 'Relationship' refers to scenarios where your artifact can be related to one, and only, one artifact. An

example of a 'relationship,' is 'parent,' as oftentimes an artifact can only have one parent artifact. 'Relationships' refers to scenarios where your artifact can be related to many artifacts. An example of 'relationships' is 'child,' as one parent-artifact can often have many child artifacts.

🟡 In general, we recommend using the 'relationships' field type in your model, rather than 'relationship,' especially in cases where you may want to map a 'relationship' field in one repository to a 'relationships' field in your other repository.

Rich Text

This is for fields that can contain rich text. These are fields that can contain html and/or wiki markup, such as bold, italics, or colored fonts. These are often Description fields.

String

String fields are used for text input. These model fields will not transmit rich text information.

Time Entries

These fields are often used when reporting time worked on an artifact.

Web Links

Web Links fields are intended to point to URLs outside of a given tool. They can contain information in addition to the URL, such as label, time of creation, and creator (depending on what the repository supports). They could also be considered a hyperlink field.

In addition to 'Web Links,' you will also see that there is a 'Location' field type available. The 'Web Links' field type includes the URL as well as additional information such as label, creator, and time of creation (depending on what the repository supports), while 'Location' includes only the URL.

Required Designation

For each field, you can configure whether or not that field requires a value.

Model Configuration Defect

View and configure your model. Models define the fields that constitute a given artifact type.

< Back to Models Cancel Save

Smart Field	Label	Type	Required
Summary	Artifact Name	String	<input checked="" type="checkbox"/>
Description	Description	Rich Text	<input type="checkbox"/>
Severity	Severity	Single Select Field Values... Sev 1, Sev 2, Sev 3, ... Allow unmapped values to flow: No	<input type="checkbox"/>

Marking a field as required has implications for all collection types:

- For repository collections, any required model field will be shown with a red asterisk in the collection to model mapping:

Field Mapping: JIRA Defects

View and manage the field mapping for this collection. The field mapping specifies how fields from your repository artifact map to fields in your model.

< Back to Collection Cancel Save

JIRA: Bug 6 of 48 fields mapped Model: Defect 6 of 16 fields mapped

Field Mapping

Specify how the fields on your repository artifact align to fields in your model. The mapping between the fields on each side forms the field mapping for this collection.

Select a field on each side to configure additional field mappings

Search "JIRA Defects" artifact fields by na... Search "Defect" model fields by name or type

- * Artifact Name (String) Not Mapped
- Modified By (Person) Not Mapped
- Priority (Single Select) Not Mapped
- Release Detected (Single Select) Not Mapped
- Release Targeted (Single Select) Not Mapped

- For gateway collections, you will need to pass in a value in the payload for any required field in order for Tasktop to accept the payload.
- For database collections, the suggested DDL will mark the field as required ("not null"); this means that if you use that suggested DDL to create your database tables, the field will be required by your database table to create a new record about an artifact:

Data Description Language Generator

Database

Model

Suggested
DDL

```
CREATE TABLE DEFECT (  
  ID BIGINT (19) AUTO_INCREMENT,  
  FORMATTED_ID VARCHAR (1000),  
  ARTIFACT_NAME VARCHAR (1000) NOT NULL,  
  DESCRIPTION VARCHAR (1000),  
  SEVERITY VARCHAR (255),  
  PRIORITY VARCHAR (255),  
  STATUS VARCHAR (255),  
  RESOLUTION VARCHAR (255),  
  RELEASE_DETECTED VARCHAR (255),  
  SPRINT_DETECTED VARCHAR (255),  
  RELEASE_TARGETED VARCHAR (255),  
  SPRINT_TARGETED VARCHAR (255),  
  CREATED_BY VARCHAR (64),  
  MODIFIED_BY VARCHAR (64),  
  OWNER VARCHAR (64)
```

Execute the DDL and Close to refresh the list of tables.

Close


Step 3: Create Your Collection(s)




Tasktop: 17.4 Release


Types of Collections

Your collections define which artifacts are eligible to flow as part of your integration.

You can create three types of collections:

 Unknown macro: 'html'

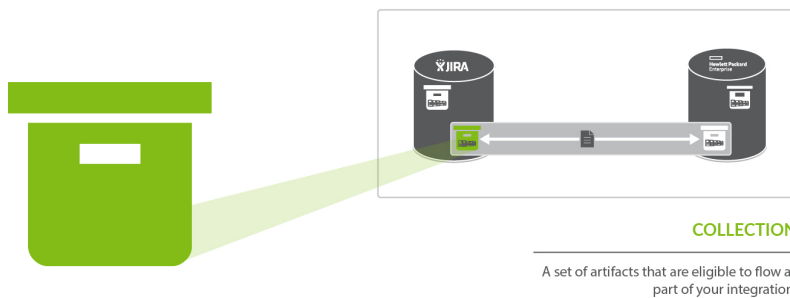
 <p>Standard Repository Collection</p>	 <p>Database Repository Collection</p>	 <p>Gateway Collection</p>
<p><i>Standard Repository Collections are available in all Editions.</i></p>	<p><i>Database Repository Collections are only available in Editions that contain the Enterprise Data Stream add-on. See Tasktop Editions table to determine if your edition contains this functionality.</i></p>	<p><i>Gateway Repository Collections are only available in Editions that contain the Gateway add-on. See Tasktop Editions table to determine if your edition contains this functionality.</i></p>
<p>A standard repository collection contains artifacts, such as defects or requirements, from repositories, such as JIRA or HPE ALM.</p>	<p>A database repository collection connects to a database repository, such as MySQL or Oracle.</p>	<p>A gateway collection contains artifacts sent via an in-bound webhook, from an external tool.</p>
<p>Learn More</p>	<p>Learn More</p>	<p>Learn More</p>

 Unknown macro: 'html'

Repository Collection (Standard)

Tasktop: 17.4 Release

What is a Collection?

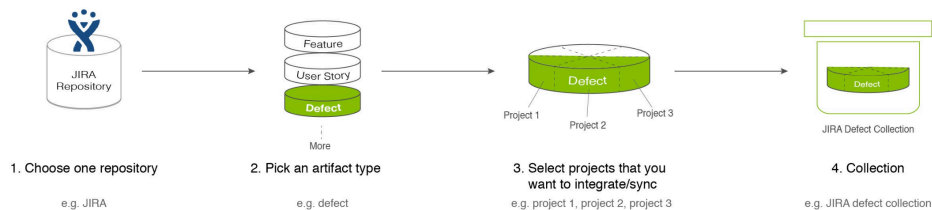


You can think of a *collection* as the set of artifacts that are eligible to flow as part of your integration. The process of creating a collection consists of a few steps which whittle down your repository into a smaller subset of artifacts. To create your collection, you will specify:

- What is a Collection?
- Types of Repository Collections
- Video Tutorial
- How to Create a Standard Repository Collection
 - Map Fields
 - Field Mapping Icons
 - Constant Value Mapping
 - Transforms
 - Configure Relationships
 - Filtered Transform

1. The repository the artifacts live in
 - a. Each collection can only come from *one* repository
2. The artifact type (i.e. defect, requirement, test case, etc)
 - a. Each collection can only contain *one* artifact type
3. The projects within the repository those artifacts live in
 - a. Each collection can contain one or multiple projects
4. The model you would like your collection to be mapped to (not pictured)
 - a. Each collection can be mapped to one and only one model

- m
- Person Reconciliation
- Set a State Transition
- Optional: Set a Repository Query



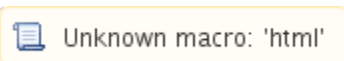
You can learn more about collections in the [Key Concepts](#).

Types of Repository Collections

There are two types of Repository Collections: Standard Repository Collections, which connect to repositories like *JIRA*, *HPE ALM*, and *ServiceNow*, and Database Repository Collections, which connect to databases, such as *MySQL*. On this page, we will be teaching you how to configure a standard repository collection.

Video Tutorial

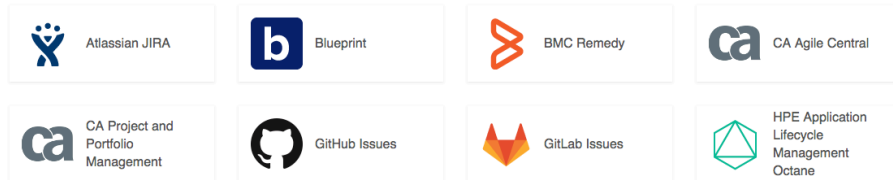
Check out the video below to learn how to create a new repository collection:



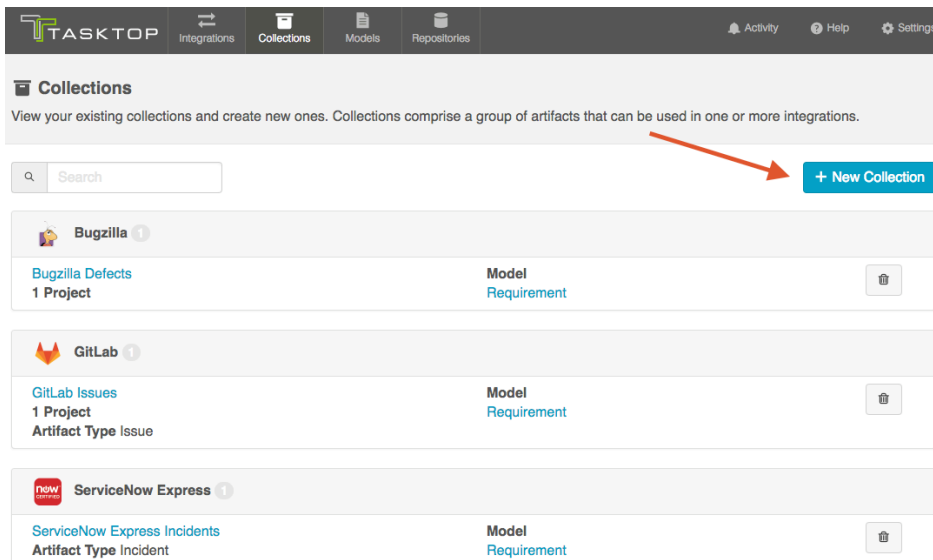
How to Create a Standard Repository Collection

To create a standard repository collection, follow the steps below:

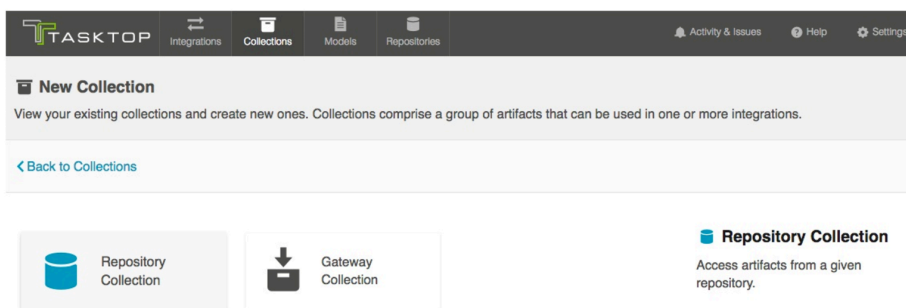
Select 'Collections' at the top of the screen:



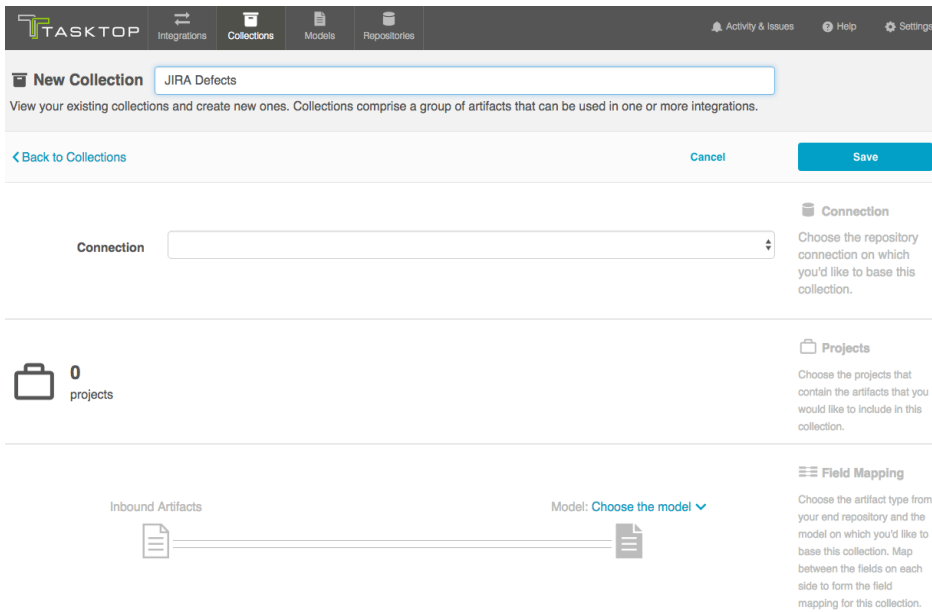
Click 'New Collection':



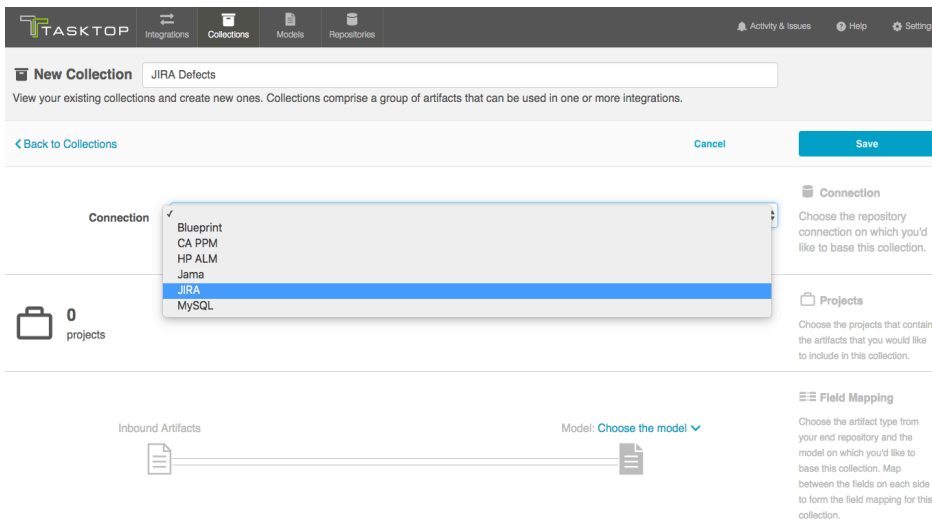
Select "Repository Collection" as the collection type.



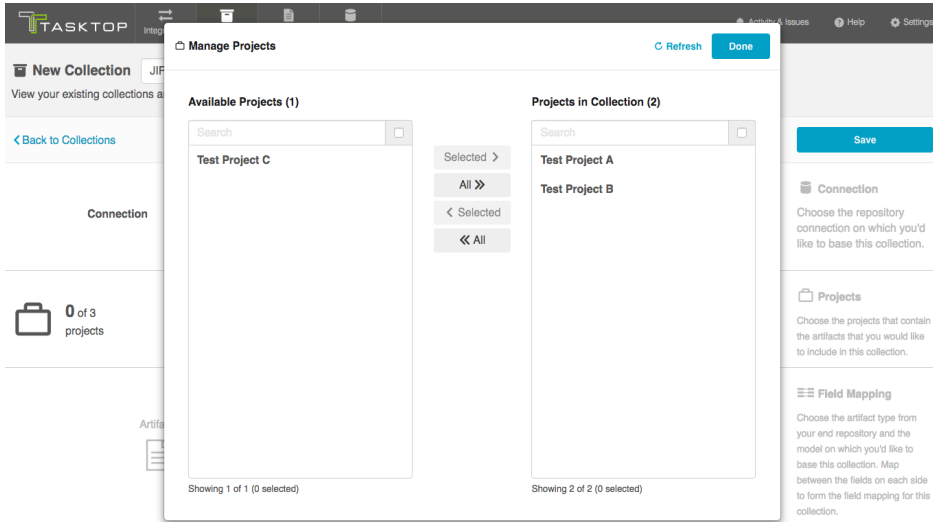
Enter a name for your collection



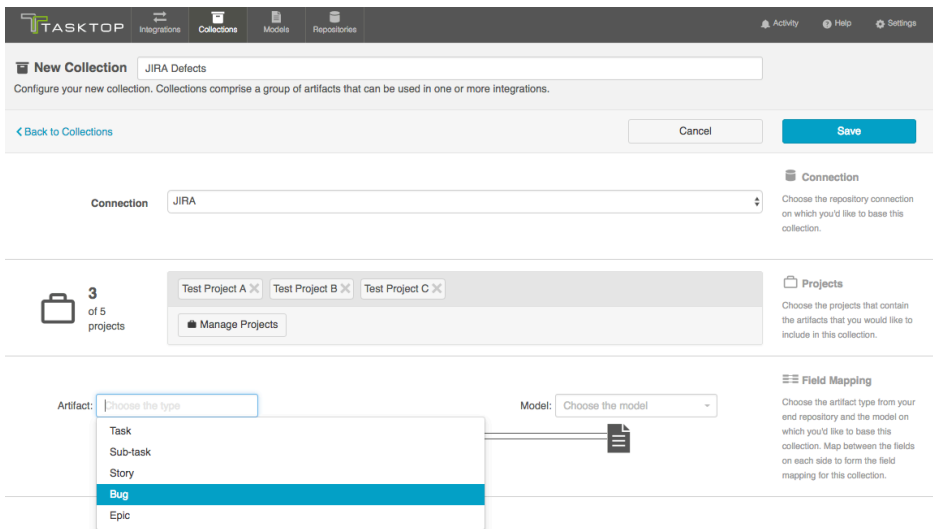
Select the Repository Connection on which you'd like to base this collection. The collection will include artifacts from the repository collection you have selected.



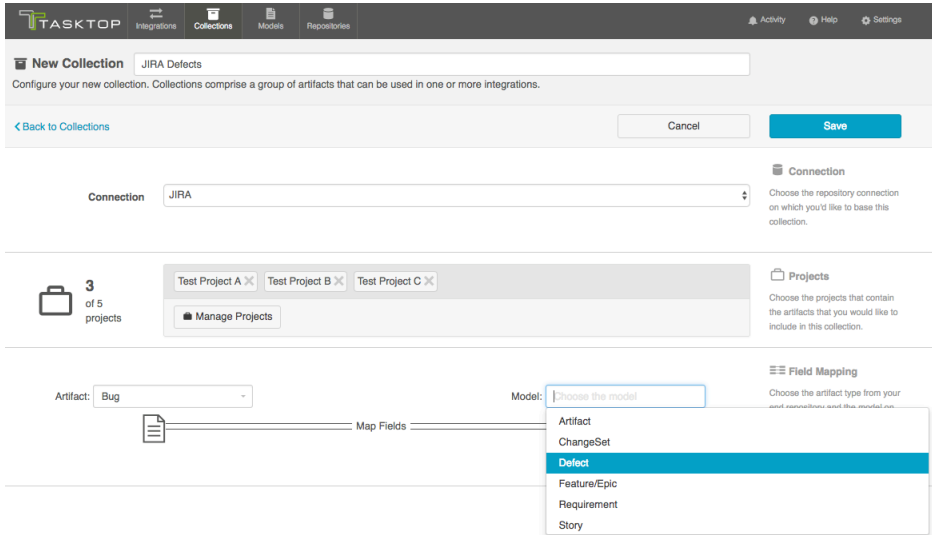
Add projects to your collection by selecting 'Manage Projects'. These are the projects from which Tasktop will be able to create, retrieve, and update artifacts.



Select the artifact type from the repository that you would like to include in this collection. Remember, a single collection can only contain artifacts of a single type.



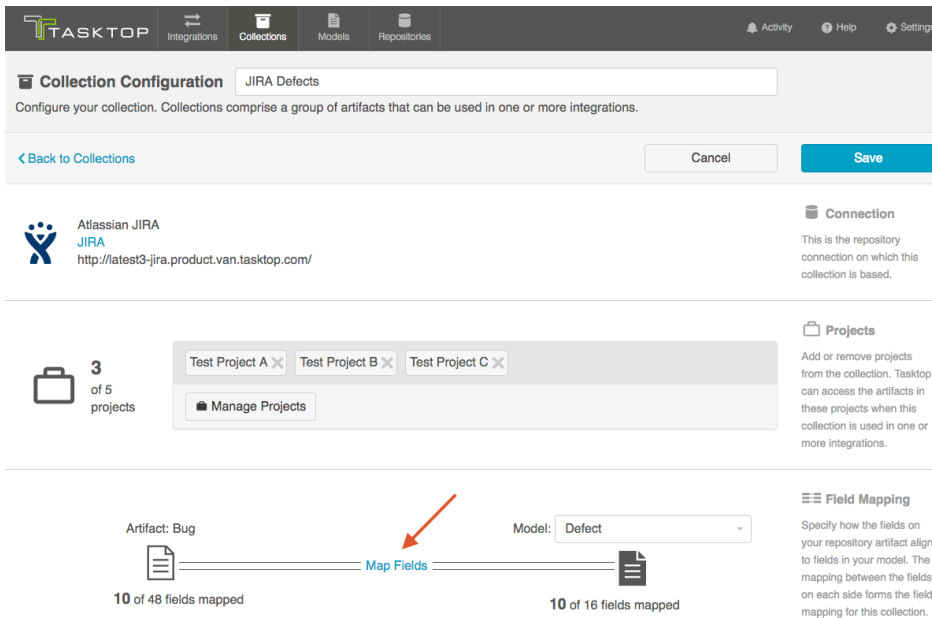
Select the Model on which you'd like to use for this collection.



⚠ Note that the projects included in your collection must contain at least one artifact of the type selected. For example, in the image above, there must be at least one bug in Test Project A in JIRA in order for your collection to save.

Map Fields

Now that you have identified the collection artifact type and model, you can complete the collection to model field mapping by clicking the 'Map Fields' link. The link will become active once you save the collection.



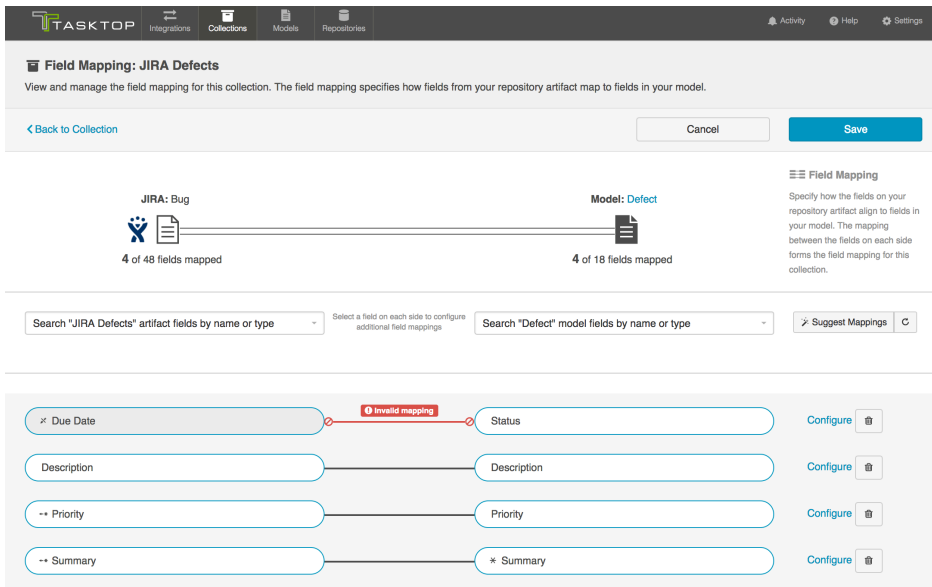
Doing so will take you to a drill in page where you can specify how the fields in your model will map to the fields available on your artifacts in the repository. Tasktop will auto-map fields when possible, based on the names of fields and the smart field designations that have been set in a given model.

💡 Tip: If you need to refresh the fields available for the collection, use

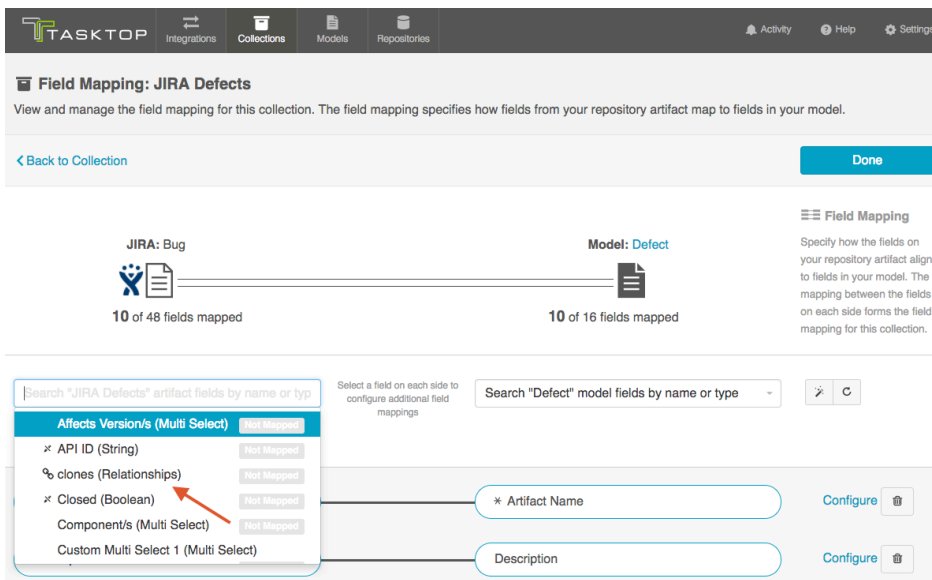
the refresh button to the right of 'Suggest Mappings,' rather than your browser's 'refresh' button.

You can map additional fields by using the two drop down boxes:

⚠️ Note: If you attempt to map fields that do not have a valid transform between one another (example: if you map 'due date,' a date field, to 'status,' a single-select field), you will get an 'invalid mapping' warning, and the mapping will not be saved.








To help troubleshoot, you can view the field type when selecting each value from the drop down menu. This will enable you to ensure that the transforms between the two field types will make sense.



Field Mapping Icons

On the Collection-to-Model Field Mapping screen, you will see a number of icons which will help you understand any special properties or requirements for each field. If you hover your mouse over an icon, you will see a pop-up explaining what the icon means. You can also review their meanings in the legend below:

Icon	Meaning
------	---------

	<p>A constant value will be sent. Note that:</p> <ul style="list-style-type: none"> • If the icon is on the side of the collection, this means that a constant value will be sent to your model. This means that any time this collection is integrated with another collection, the <i>other</i> collection will receive this constant value for the field in question. • If the icon is on the side of the model, this means a constant value will be sent to your collection. This means that any time this collection is integrated with another collection, that <i>this</i> collection will receive this constant value for the field in question.
	<p>Collection field is read-only and cannot receive data.</p>
	<p>To create artifacts in your collection, this field must be mapped to your model.</p>
	<p>This is a required field in your model; it must be mapped to your collection.</p>
	<p>This field will not be updated as part of your integration because the mapping would be invalid. You do not have the option of changing this.</p>

Constant Value Mapping

In some scenarios, either the collection artifact or the model might require that a value be provided for a given field. This value is usually provided by mapping it to the equivalent field. However, sometimes your collection artifact has a field that needs a value that doesn't align with any fields in your model, and sometimes your model might have a required field that doesn't have an equivalent field from the collection artifact. In these cases, you can set a constant value. By doing so, you'll specify the value that you would like to provide for that field.

Constant values can be set for the following field types:

- Boolean
- Date/DateTime

- Double
- Location
- Long
- Multi-Select
- Person
- Rich Text
- Single-Select
- String

Scenario 1: If your repository requires a field for artifact creation, but that field is not a part of your model:

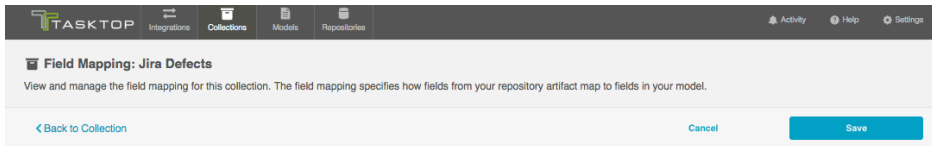
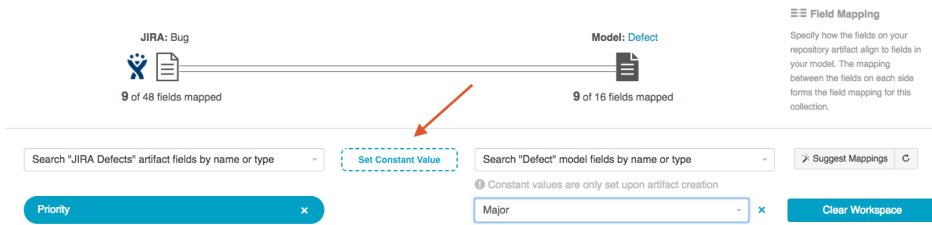
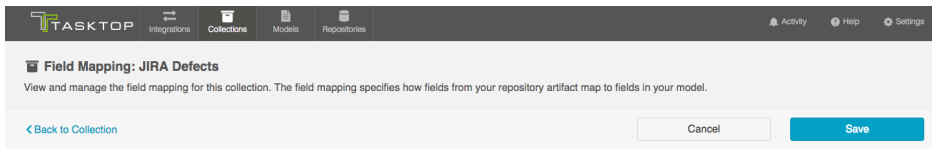
Solution: Set a constant value on the side of the model, to send to your collection.

To set a constant value for a field, select 'Constant Value' from the drop down menu on the model side. Enter the value, and then click the 'Set Constant Value' box.

The image displays two screenshots of the Tasktop Field Mapping interface for 'Jira Defects'.

Top Screenshot: Shows the initial state where the 'Priority' field from the 'JIRA: Bug' repository is mapped to the 'Modified By (Person)' field in the 'Model: Defect'. A dropdown menu is open on the model side, showing options: 'Constant Value', 'Constant Value Based on Projects', 'Modified By (Person)', 'Priority (Single Select)', 'Severity (Single Select)', and 'Release Detected (Single Select)'. The 'Constant Value' option is selected.

Bottom Screenshot: Shows the next step where the 'Set Constant Value' button is active. A dropdown menu is open on the model side, showing options: 'Blocker', 'Critical', 'Major', 'Minor', and 'Trivial'. The 'Major' option is selected. A note above the dropdown states: 'Constant values are only set upon artifact creation'.



Once the constant value is set, you will notice a few things:

- The pill will be rectangular and grey: this denotes that a constant value has been set.
- The Constant Value icon will be displayed inside the pill.
- The 'prohibited' icon will appear next to the pill. This indicates that no values can be sent to the Constant Value field. The constant value is essentially a dead end, and cannot be linked to a repository or model on the other side.

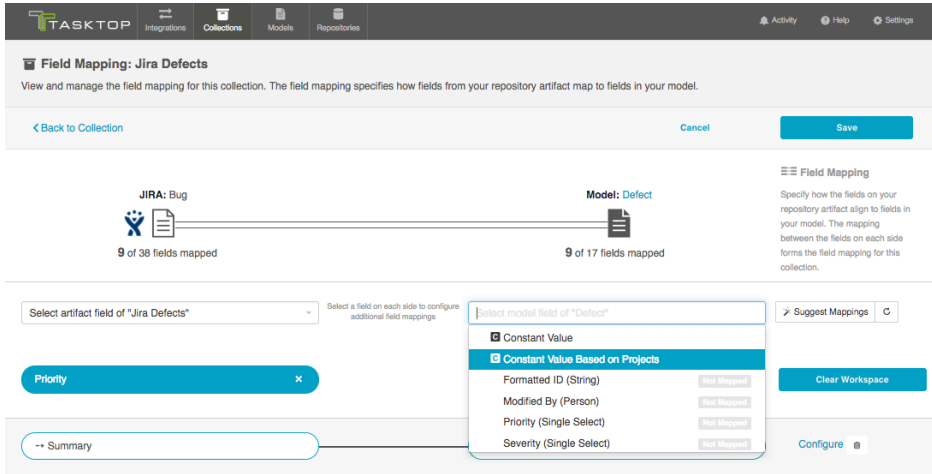
In the scenario above, any time a new defect is created in JIRA, the priority will be set to 'Major.' JIRA will not send 'priority' data to any other collections, as 'priority' does not exist in the model.

If desired, you can also set constant values per project:

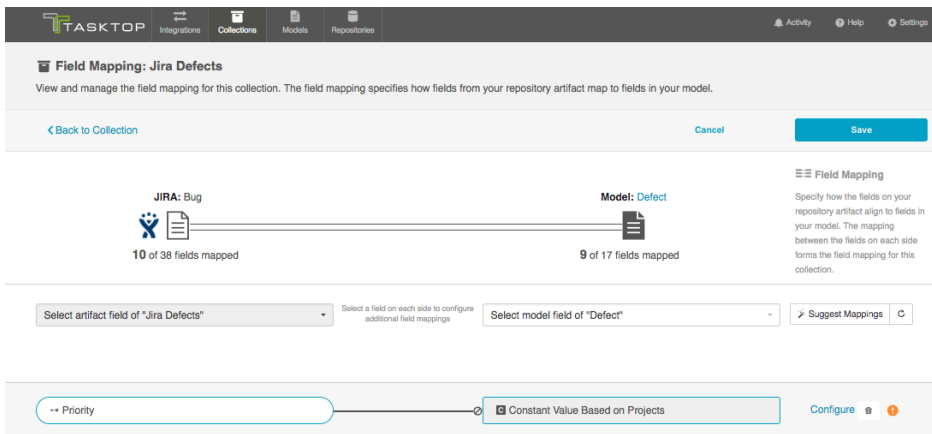
You may wish to set a constant value based on project in the following scenarios:

- In order to set a unique value for a specific field, such as release or iteration, depending on the project
- If the values for a single-select field vary across projects

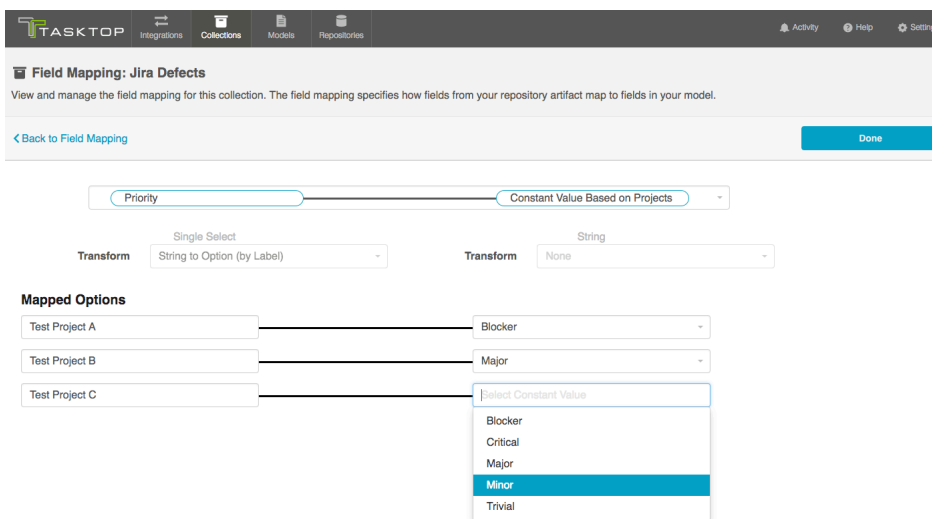
To do this, select 'Constant Value Based on Projects':



Once selected, you will see an orange exclamation point appear next to the 'Configure' link:

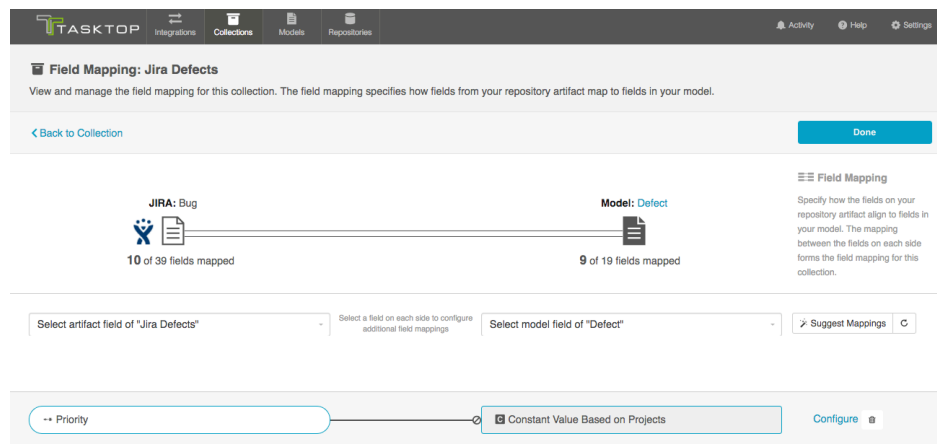


Click "Configure" to get to the Configuration Screen. On this screen, you will be able to set a distinct constant value for each project in your collection:



In the screenshot above, an artifact created in Test Project A would be assigned the value of "Blocker" for the Priority field, while an artifact created in Test Project B would be assigned the value of "Major" for the

Priority field. Finally, an artifact created in Test Project C would be assigned the value "Minor" for the Priority field.



Once the constant value is set, you will notice a few things:

- The pill will be rectangular and grey: this denotes that a constant value has been set.
- The Constant Value icon will be displayed inside the pill.
- The 'prohibited' icon will appear next to the pill. This indicates that no values can be sent to the Constant Value field. The constant value is essentially a dead end, and cannot be linked to a repository or model on the other side.

Note: Sometimes, a single-select field in your collection will not return any values that you can select in the UI. In cases when this is true, and when the artifact will accept new values for that field, you will see a text input in which you can configure a constant value (instead of the traditional drop-down list for a single-select).

Scenario 2: If your model requires a field, but the end repository utilized in your collection does not have that field:

Solution: Set a constant value on the collection side to send to your model. This means that any time this collection creates a corresponding artifact in another collection, the 'severity' field (in the example below) will be set to the constant value, in that other repository.

To set a constant value for a field, select 'Constant Value' from the drop down menu on the collection side. Enter the value, and then click the 'Set Constant Value' box.

Field Mapping: JIRA Defects
View and manage the field mapping for this collection. The field mapping specifies how fields from your repository artifact map to fields in your model.

[Back to Collection](#) Cancel Save

JIRA: Bug 3 of 48 fields mapped Model: Defect 4 of 18 fields mapped

Field Mapping
Specify how the fields on your repository artifact align to fields in your model. The mapping between the fields on each side forms the field mapping for this collection.

Search "JIRA Defects" artifact fields by name or type Select a field on each side to configure additional field mappings Search "Defect" model fields by name or type Suggest Mappings C

Sev 3	Severity	Configure
Description	Description	Configure
Priority	Priority	Configure
Summary	* Summary	Configure

Once the constant value is set, you will notice a couple of things:

- The pill will be rectangular and grey: this denotes that a constant value has been set.
- The 'prohibited' icon will appear next to the pill. This indicates that no values can be sent to the Constant Value field. This makes sense, because in this example your repository did not have a 'severity' field to begin with.

Transforms

When you map a collection field to a model field, it's necessary to transform the data from the source field to the target field. Depending on the field types, that transform may or may not be possible within Tasktop Integration Hub.

You can see a table of the available transforms by appending '/transforms' to the URL of your instance's Tasktop home page. For example, `<http://localhost:8080>/#/transforms`

This will lead you to the Field Value Transformations screen:

Field Value Transformations

See the transformations available between different field types within Tasktop Integration Hub

Filter table by collection

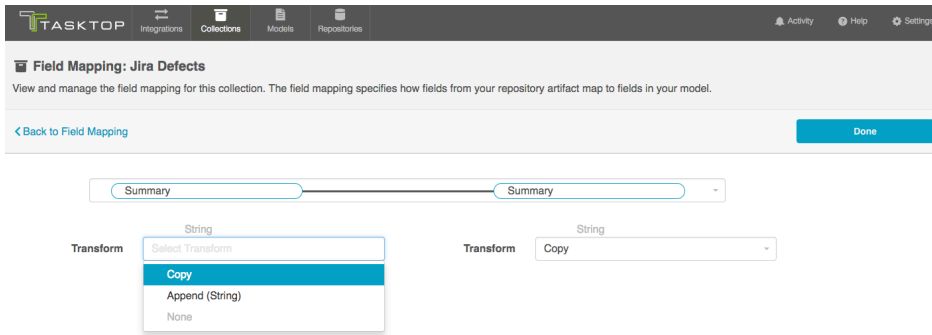
Clear

Displaying transformations for field types from repository collection Jira Defects

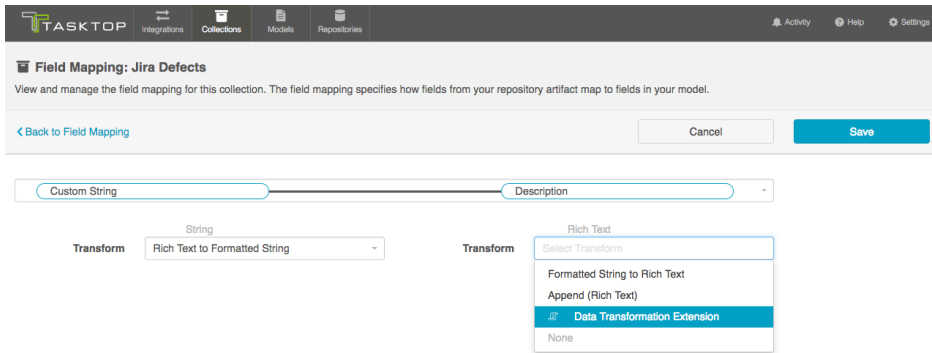
		Supported Model Field Types							
		Field Labels	Date Time	Location	Person	Relationships	Rich Text	Single Select	String
Supported Collection Field Types	Attachments	<ul style="list-style-type: none"> Attachment 							
	Boolean	<ul style="list-style-type: none"> Closed 						●	●
	Comments	<ul style="list-style-type: none"> Comment 							
	Date	<ul style="list-style-type: none"> Due Date 	●						●
	Date Time	<ul style="list-style-type: none"> Resolved Created Updated 	●						●
	Duration	<ul style="list-style-type: none"> Time Spent Original Estimate Remaining Estimate 							●
	Location	<ul style="list-style-type: none"> URL 		●			●		●
	Multi Select	<ul style="list-style-type: none"> Flagged Epic/Theme Component/s <p>Show More (5)</p>				●		●	●
	Person	<ul style="list-style-type: none"> Reporter Assignee 			●				●
	Persons	<ul style="list-style-type: none"> Watchers 			●				●
	Relationship	<ul style="list-style-type: none"> Epic Link 		●		●	●	●	●
	Relationships	<ul style="list-style-type: none"> Sub-Tasks is blocked by blocks <p>Show More (6)</p>				●	●	●	●
	Rich Text	<ul style="list-style-type: none"> Description Environment 					●		●
	Single Select	<ul style="list-style-type: none"> Project Issue Type Resolution <p>Show More (5)</p>				●		●	●
	String	<ul style="list-style-type: none"> Rank Issue ID API ID <p>Show More (3)</p>	●	●	●		●	●	●
	Time Entries	<ul style="list-style-type: none"> Log Work 							
Web Links	<ul style="list-style-type: none"> Web Links 							●	

Once you've confirmed that your field type transformation is possible, you can click 'Configure' next to the mapping pair in order to see the available data transformations. Similar fields in different repositories often come in different formats, resulting in the need for values to be transformed to the proper format for a given repository. This screen allows you to configure how different types of fields will translate from

one to the other (for example, 'copy,' 'formatted string to rich text,' 'append,' or 'none'). In most scenarios, the default setting will be appropriate, and you will not need to modify anything here.

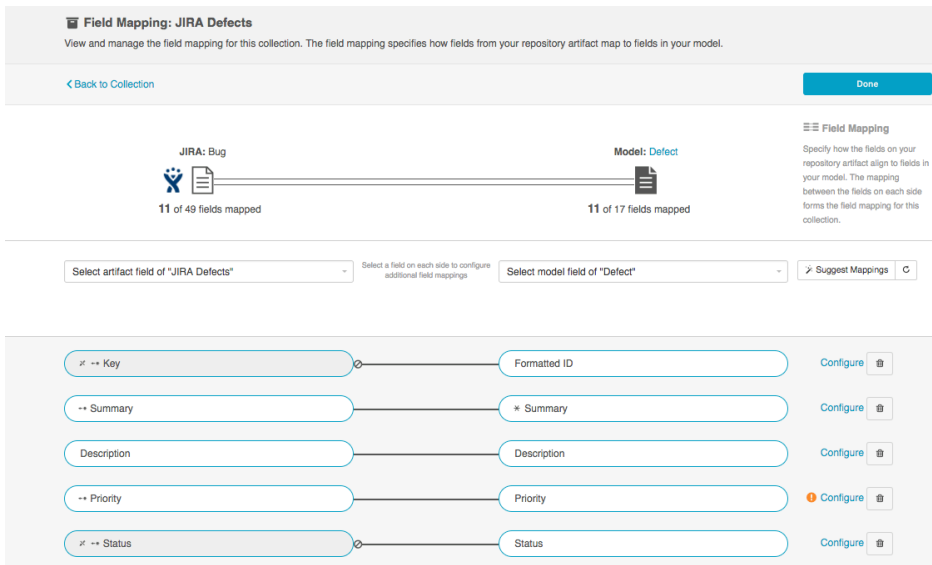


If you have configured a Custom Data Transformation extension, you can also apply it here:

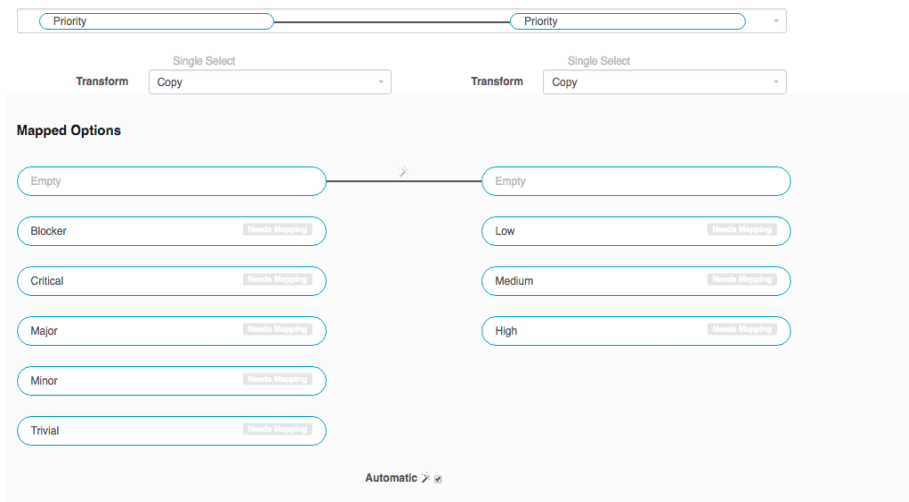


Single-Select Fields

When flowing single-select fields, it is important to click the 'configure' button in order to map your field options to the model. You will see an alert next to any mappings that require additional configuration.



Once you click 'Configure,' you will be lead to the Field Mapping Screen:



Transforms

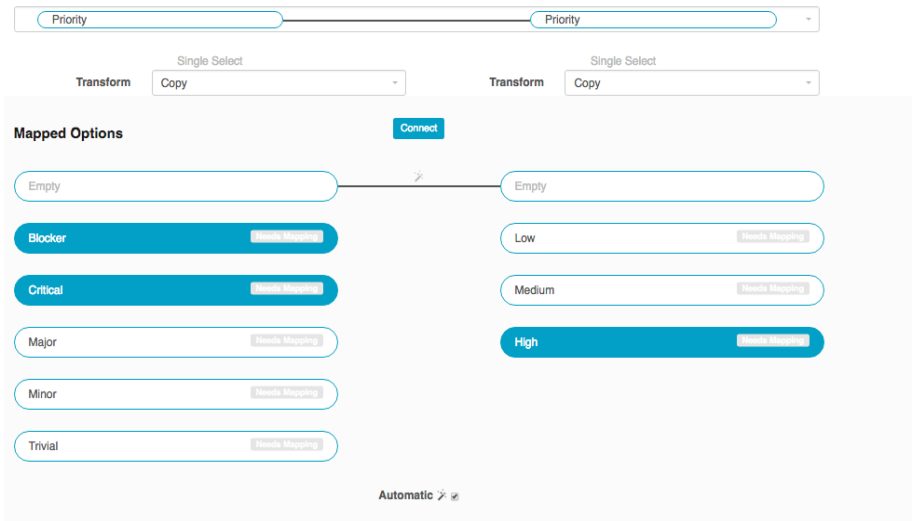
In most scenarios, you will configure your transform as 'copy' on both the collection and on the model side. This means that the model will pass an identical copy of its value to the collection, and vice versa. This should be the default setting.

If you are using a state transition extension, however, the transform on the left will be 'none' and the transform on the right will be 'copy.'

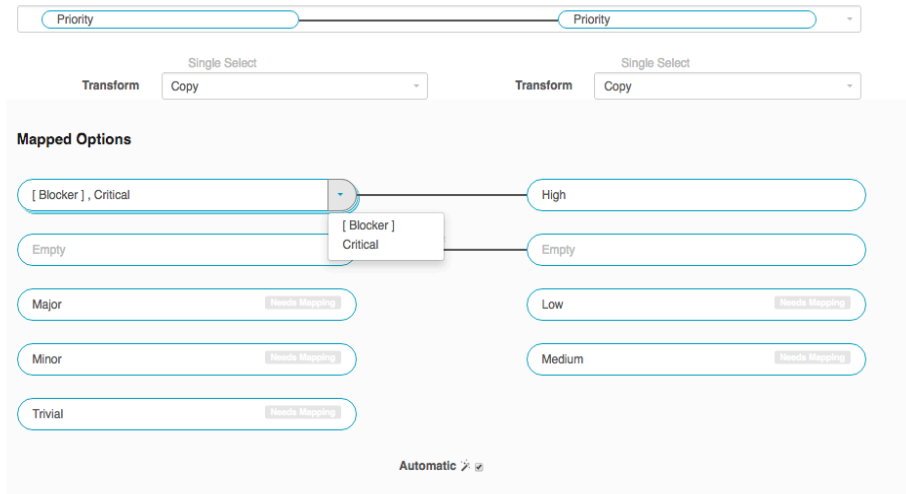
Field Mapping

If the 'Automatic' magic wand box is checked, it means that Tasktop will use its built-in smarts to pre-map some of the fields for you, based on their labels. If you'd like, you can click the trash can icon next to each mapping to remove the mapping, and then manually re-map it.

To complete the field mapping, select the values in the collection and in the model that you would like to map to one another, and then click 'connect.' This process enables to the model to act as a 'translator' between two different collections which may have different sets of options for a single-select field.



💡 When you map multiple collection values to a single model value, you will find that one value on the collection side is listed in brackets. This indicates which value will be set when the mapped model value is passed in. In the scenario below, if the model passes a 'high' priority value to your collection, that artifact will default to a priority status of 'blocker,' rather than 'critical.' You can modify the default value by clicking the arrow icon on the collection field pill.



You can also map many model fields to a single collection field. The brackets on the model side similarly indicate which value will be set in the model when either of the mapped collection values are passed in.

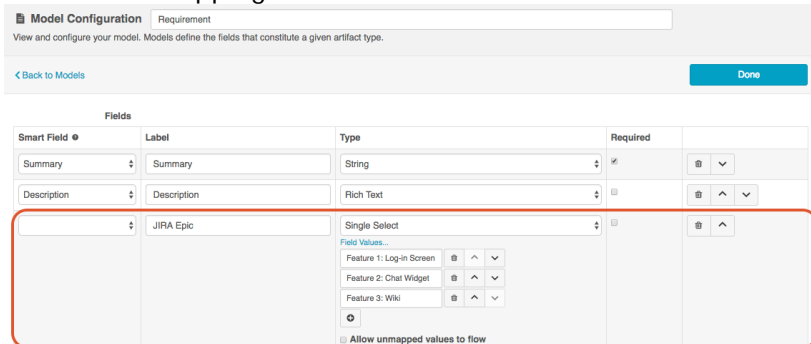
Relationship to Single-Select Transform

If desired, you can map a relationship on your source artifact to a single-select field on your target artifact. For example, you may wish to write the JIRA Epic-link (relationship) to a custom single-select field in QASymphony qTest Manager. In order to do that, you will need to map

a relationship field in your source collection to a single-select field in your model.

Here's how to configure this scenario in Tasktop Integration Hub...

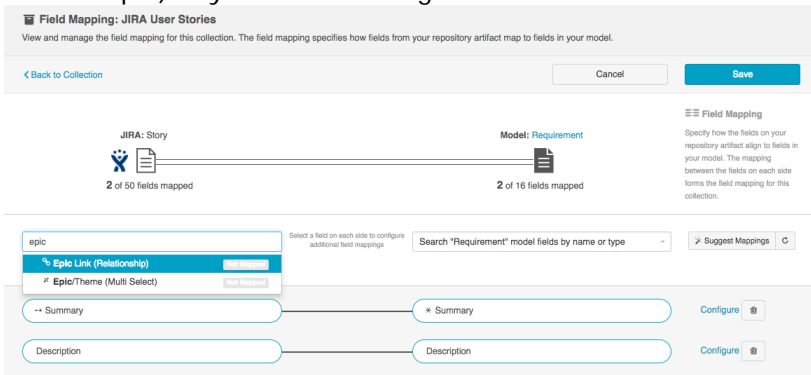
1. Ensure that your model includes a corresponding single-select field for the mapping



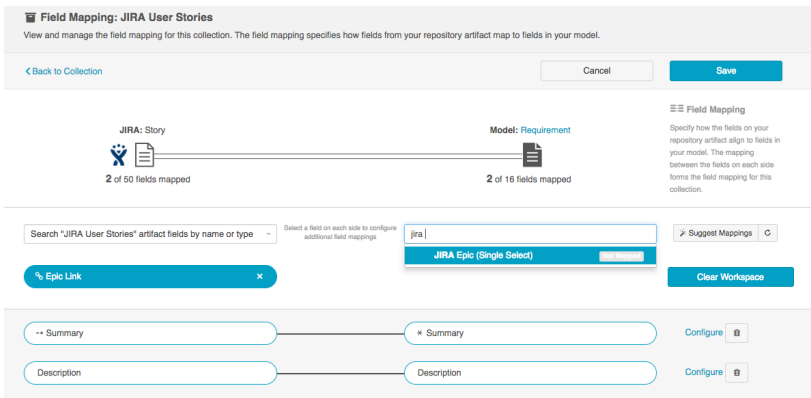
The screenshot shows the 'Model Configuration' interface for a 'Requirement' model. It features a table with columns for 'Smart Field', 'Label', 'Type', and 'Required'. The 'JIRA Epic' row is highlighted with a red border. Below the table, there is a 'Field Values...' section with three options: 'Feature 1: Log-in Screen', 'Feature 2: Chat Widget', and 'Feature 3: Wiki'. A 'Done' button is located at the top right.

Smart Field	Label	Type	Required
Summary	Summary	String	<input checked="" type="checkbox"/>
Description	Description	Rich Text	<input type="checkbox"/>
	JIRA Epic	Single Select	<input type="checkbox"/>

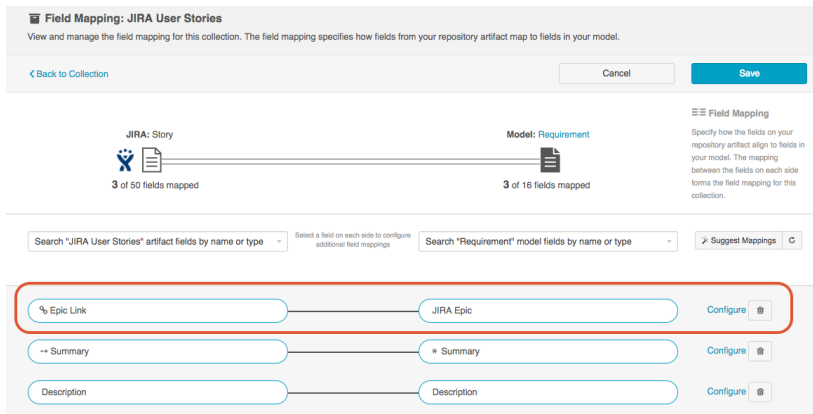
2. In the source collection, click on 'Map Fields,' and create a mapping from the collection's relationship field (Epic-Link in this example) to your model's single-select field.



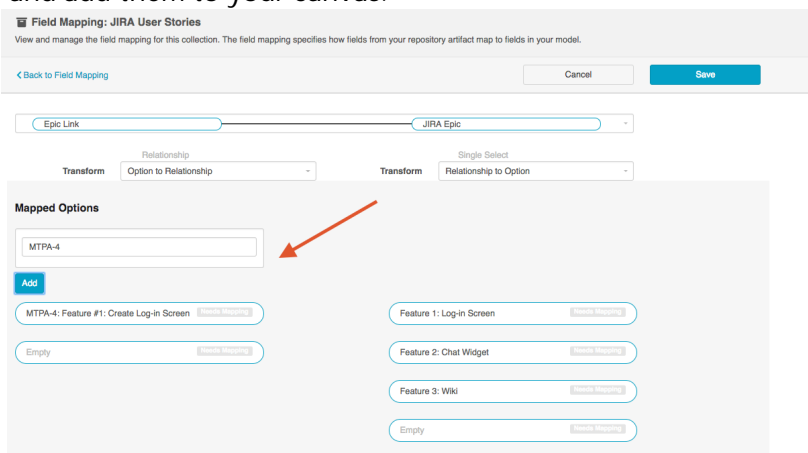
The screenshot shows the 'Field Mapping: JIRA User Stories' interface. It displays a diagram with 'JIRA: Story' (2 of 50 fields mapped) and 'Model: Requirement' (2 of 16 fields mapped). Below the diagram, there are search boxes for 'epic' and 'Requirement' model fields. A list of source fields includes 'Epic Link (Relationship)' and 'Epic/Theme (Multi Select)'. Two mappings are shown: 'Summary' to 'Summary' and 'Description' to 'Description', each with a 'Configure' button.



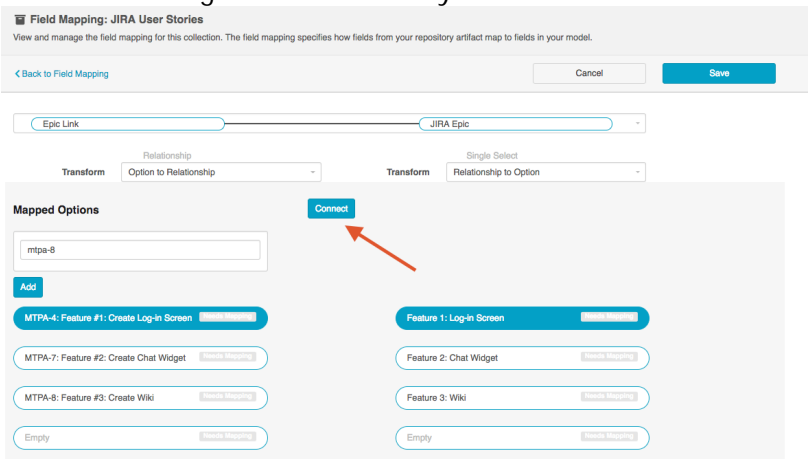
The screenshot shows the 'Field Mapping: JIRA User Stories' interface. It displays a diagram with 'JIRA: Story' (2 of 50 fields mapped) and 'Model: Requirement' (2 of 16 fields mapped). Below the diagram, there are search boxes for 'JIRA User Stories' artifact fields and 'jira'. A list of source fields includes 'Epic Link'. A mapping is shown between 'Summary' and 'Summary' with a 'Configure' button.



3. Once the fields are mapped, click the 'Configure' link on the right side
4. Here you can search for the related Epics by their formatted ID , and add them to your canvas.



5. Once the related Epics are added to the canvas, map them to the available single-select fields in your model.



Field Mapping: JIRA User Stories
View and manage the field mapping for this collection. The field mapping specifies how fields from your repository artifact map to fields in your model.

[Back to Field Mapping](#) Cancel Save

Epic Link — JIRA Epic

Relationship: Option to Relationship | Single Select: Relationship to Option

Mapped Options

mpa-8

Add

- MTPA-4: Feature #1: Create Log-in Screen — Feature 1: Log-in Screen
- MTPA-7: Feature #2: Create Chat Widget — Feature 2: Chat Widget
- MTPA-8: Feature #3: Create Wiki — Feature 3: Wiki
- Empty — Empty

6. Click 'Save' and 'Done.'
7. Navigate to your target collection
8. Map the target collection field to the single-select field in your model. Click configure to map the field options.

Field Mapping: QASymphony Requirements
View and manage the field mapping for this collection. The field mapping specifies how fields from your repository artifact map to fields in your model.

[Back to Collection](#) Cancel Save

QASymphony qTest Manager: Requirement — Model: Requirement

2 of 36 fields mapped — 2 of 3 fields mapped

Field Mapping: Specify how the fields on your repository artifact align to fields in your model. The mapping between the fields on each side forms the field mapping for this collection.

Search "Requirement" model fields by name or type

JIRA Epic (Single Select)

- Name — Summary
- Description — Description

Field Mapping: QASymphony Requirements
View and manage the field mapping for this collection. The field mapping specifies how fields from your repository artifact map to fields in your model.

[Back to Collection](#) Cancel Save

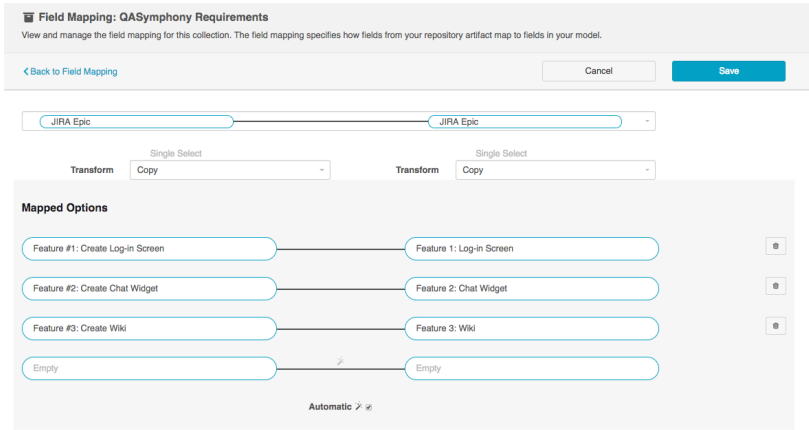
QASymphony qTest Manager: Requirement — Model: Requirement

3 of 36 fields mapped — 3 of 3 fields mapped

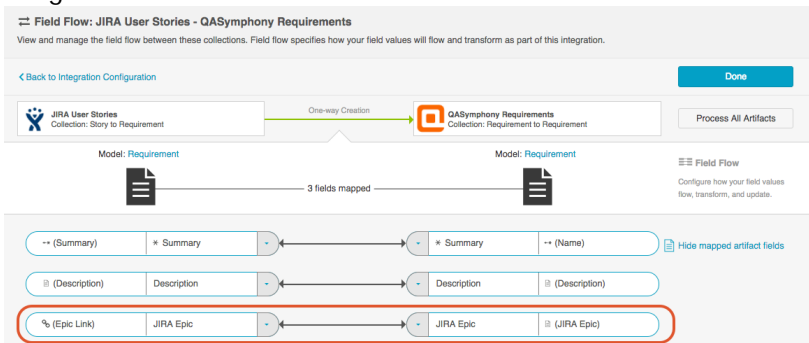
Field Mapping: Specify how the fields on your repository artifact align to fields in your model. The mapping between the fields on each side forms the field mapping for this collection.

Search "QASymphony Requirements" artifact fields by ... | Search "Requirement" model fields by name or type

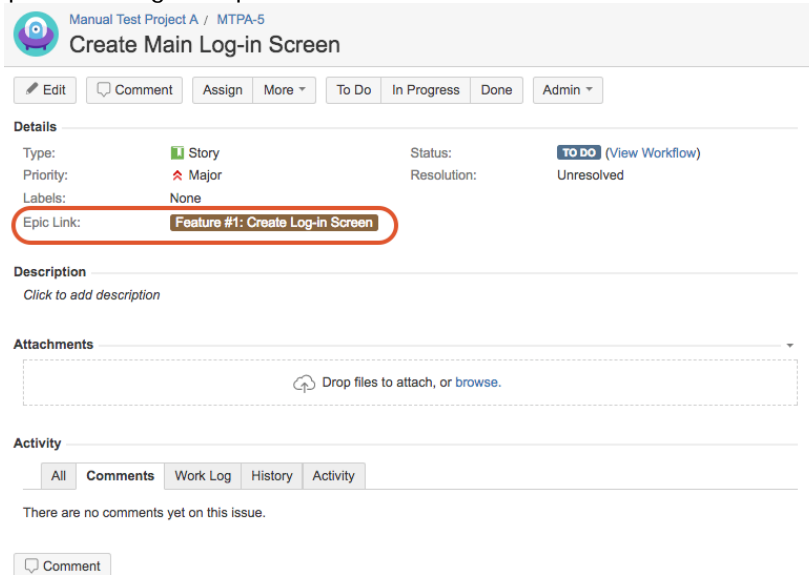
- JIRA Epic — JIRA Epic Configure
- Name — Summary Configure
- Description — Description Configure

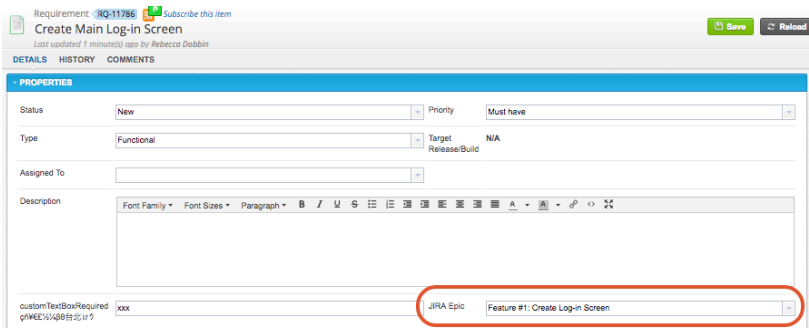


9. Once you've configured your integration, your completed Integration Field Flow will look like this:



10. When you run your integration, the single-select in your target repository will be updated based on the epic link (relationship) in your source repository.
11. Here's the original user story in JIRA. You can see that its Epic Link (a relationship to an associated Epic artifact) has flowed to the 'JIRA Epic' field (a single-select field) on the QASymphony qTest Manager requirement:

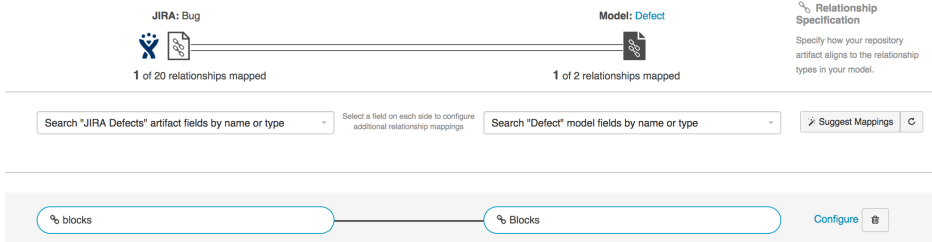
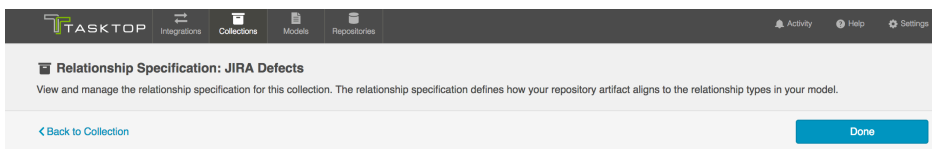
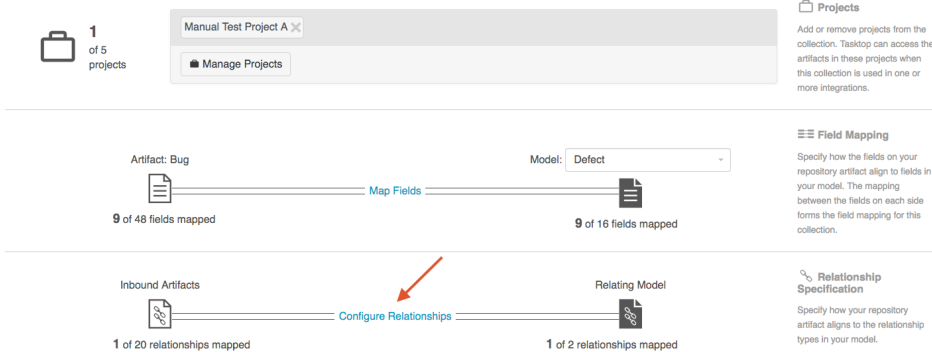
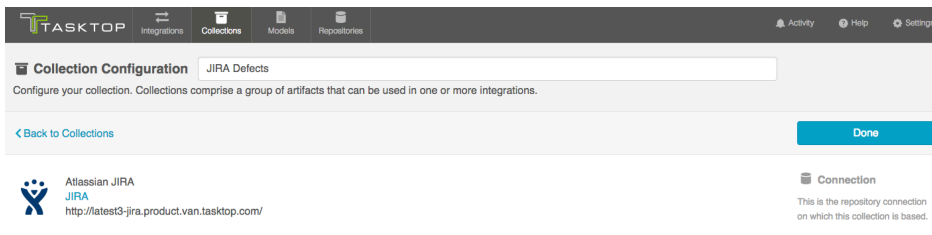




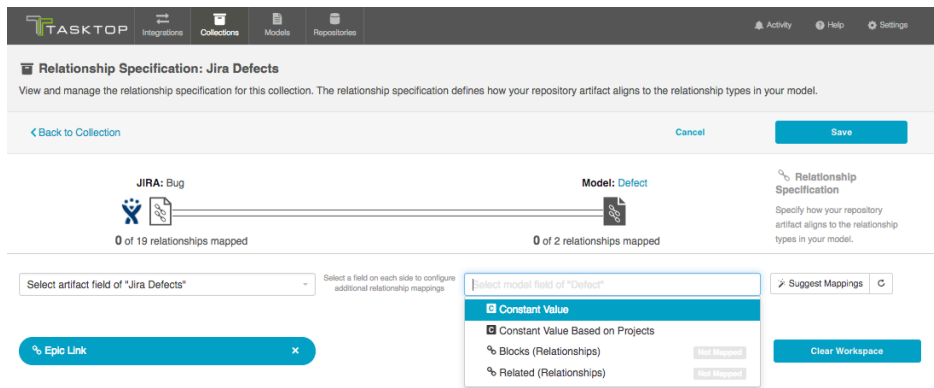
Configure Relationships

If you have any relationship(s) fields in your model, you can map those by clicking the "Configure Relationships" link.

💡 Note that any relationship(s) types you'd like to flow as part of your integration must be mapped to each collection involved in the integration.



For 'relationship' type fields, you have the option of configuring constant values. To learn how to configure constant values, please reference the [constant value section](#) above.



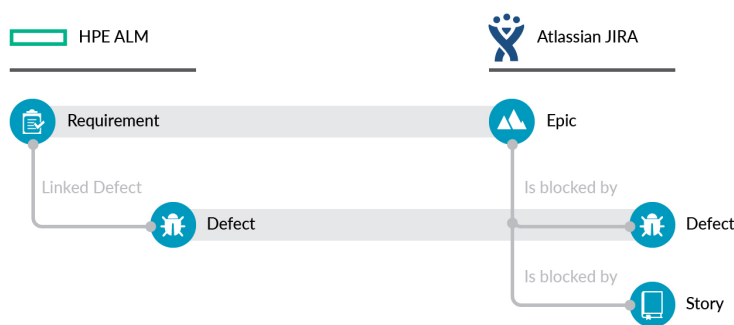
You can learn more about Artifact Relationship Management (ARM) [here](#).

Filtered Transform

Consider this example scenario: In JIRA, the relationship type, 'is blocked by' can create a relationship from a JIRA Epic to both JIRA Defects and JIRA Stories. However, let's imagine that you would only like to flow JIRA Epics and JIRA Defects in your integration, and not JIRA Stories.

Under normal circumstances, if your JIRA artifact were blocked by a JIRA Story, your integration may error out, due to the fact that Tasktop would be unable to locate the related Story in your integration (since you have not configured your integration to flow Stories).

To avoid receiving this error, use the 'Copy (Filtered)' transform for your relationship mapping on the collection side.

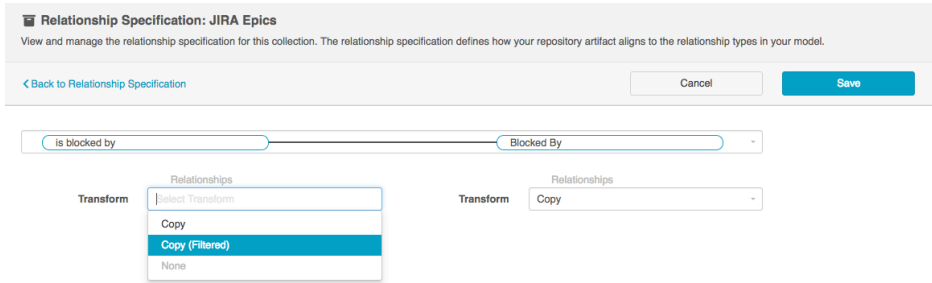


To re-cap, the 'Copy (Filtered)' transform should be used in the following scenario:

- You are flowing *Artifact Type A* via your Tasktop integration
- You have mapped *Relationship Type B* to your collection
- *Relationship Type B* has the potential to create a relationship

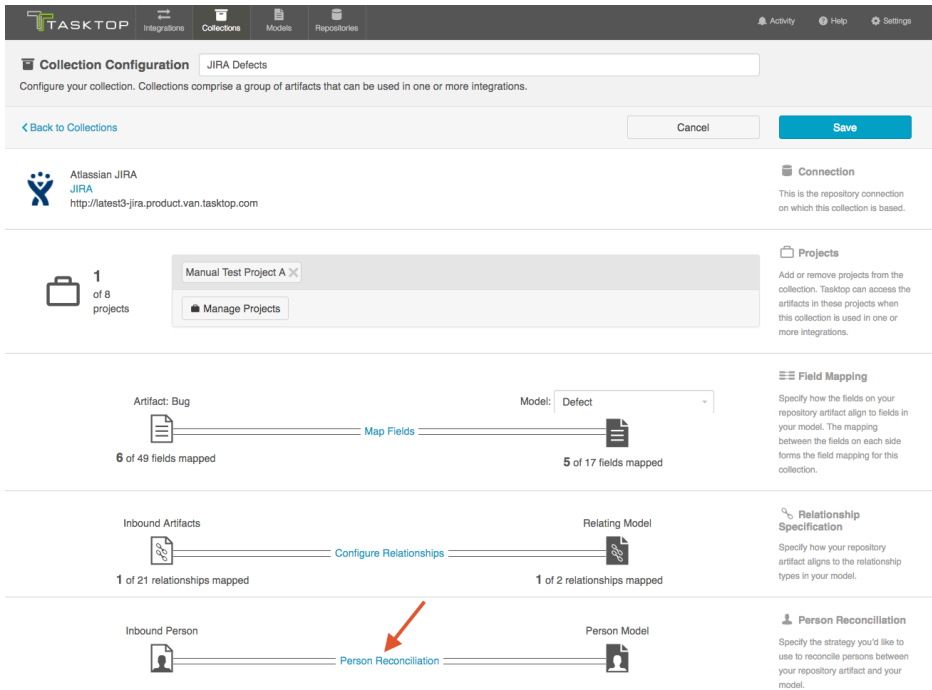
between *Artifact Type A* and *Artifact Type C*, or between *Artifact A* and *Artifact D* in your source repository

- *Artifact C* is flowing in your integration, BUT
- *Artifact Type D* is not flowing in your integration

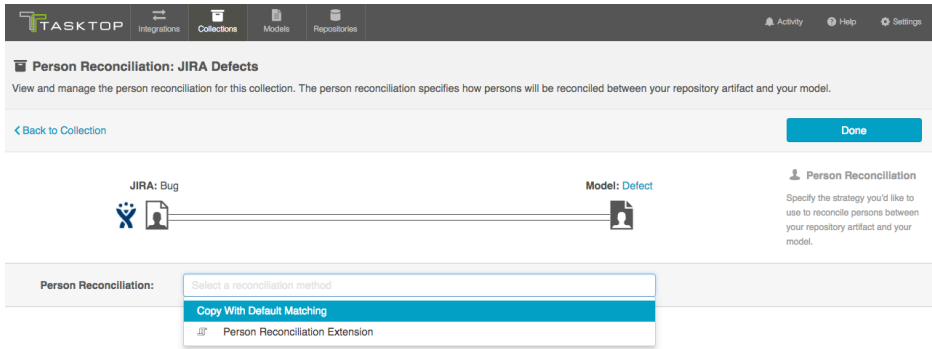


Person Reconciliation

To configure Person Reconciliation, click the 'Person Reconciliation' link.

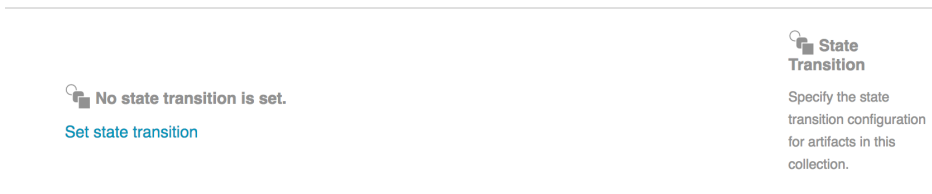


If you have configured a [Person Reconciliation extension](#) on the settings page, you will be able to select that extension here, or to choose our default person reconciliation strategy ("Copy with Default Matching"). Our default algorithm will match based on name, ID, and/or e-mail.

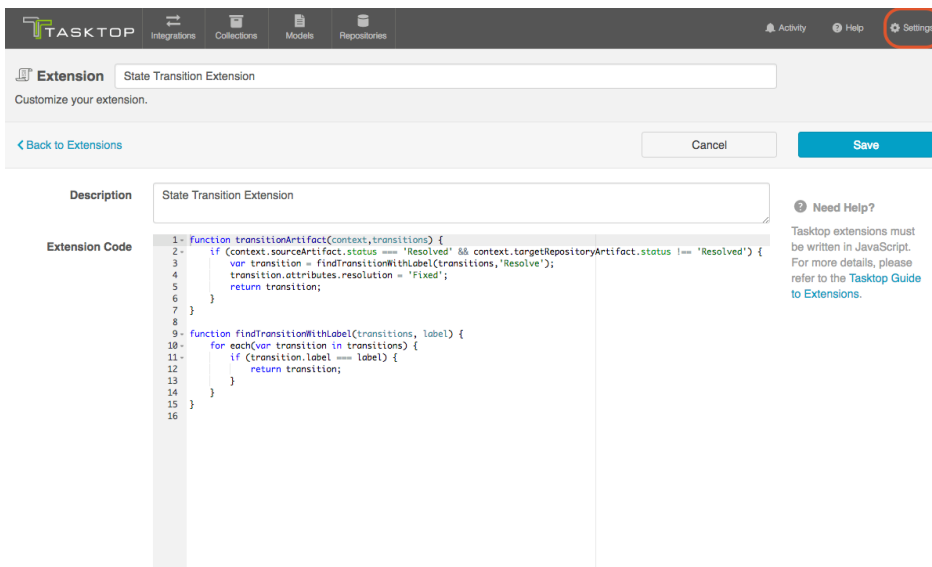


Set a State Transition

Some repositories require pre-defined state transitions for certain fields (for example, where an artifact must move from a status of *New* to *In Progress* to *Closed*). If your collection utilizes a repository that makes use of state transitions, you will see a State Transition sash at the bottom of the Collection Configuration screen:

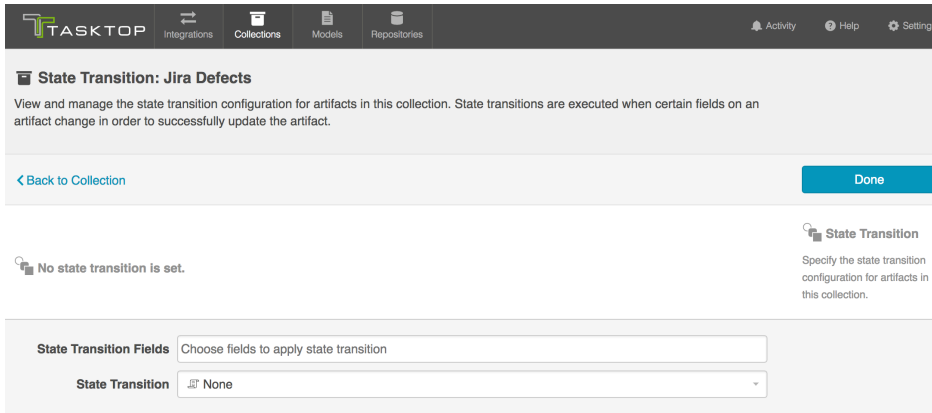


In order to successfully flow field values for fields that require state transitions, a state transition extension must be set. First, create and save the extension itself from the [Settings](#) screen. If you need help creating the extension, you can find more information in the [Extensions](#) section.

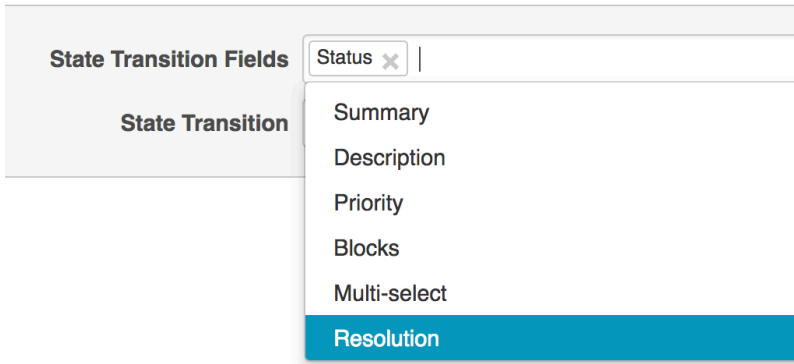


Once the extension is configured, click 'Set state transition' on the State Transition Sash on the Collection page.

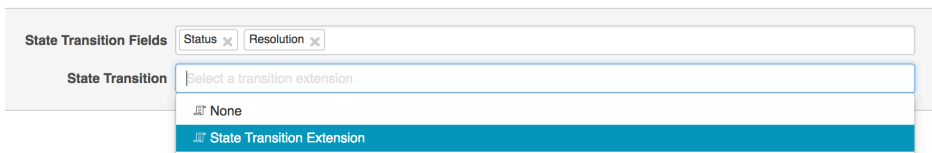
This will lead you to the State Transition screen:



Enter any fields that you would like to apply your state transition extension to in the 'State Transition Fields' box:

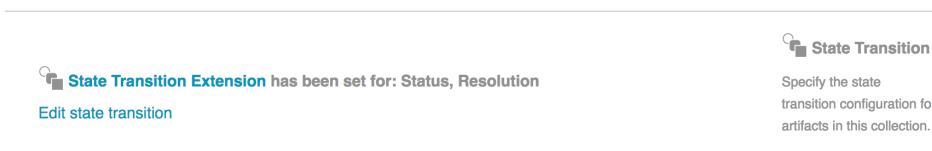



Next, select the State Transition Extension that you would like to use:



Click 'Save,' and then 'Done.'

You will now see your extension outlined on the Collection Configuration screen:



 Note that the extension will only impact how data flows *from* the model to the repository (Jira in this case). If you would like impact how data flows from the repository to the model (and then to whichever target collection is connected on the other side), you will need to [configu](#)

re the field appropriately. If you would like to use a state transition extension on the other side, you must configure that on the corresponding repository collection's State Transition screen.

Optional: Set a Repository Query

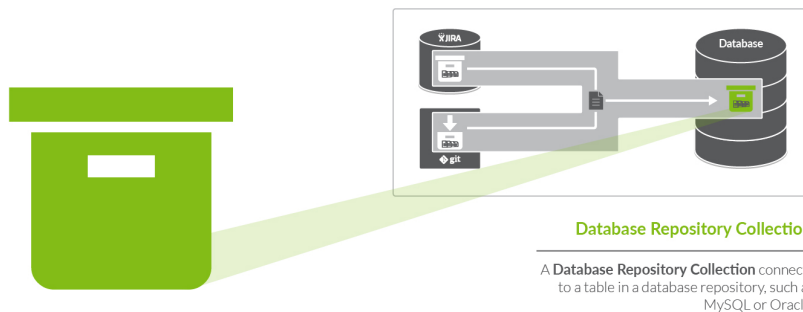
You can learn more about Repository Queries [here](#).

Repository Collection (Database)

Tasktop: 17.4 Release

Database Collections are only available in Editions that contain the Enterprise Data Stream add-on. See [Tasktop Editions table](#) to determine if your edition contains this functionality.

What is a Database Repository Collection?



- What is a Database Repository Collection?
- Video Tutorial
- How to Create a Database Collection
 - Map Fields
 - Constant Value Mapping

There are two types of Repository Collections: Standard Repository Collections, which connect to repositories like *JIRA* or *HPE ALM* and Database Repository Collections, which connect to databases, such as *MySQL*. On this page, we will be teaching you how to configure a database repository collection.

A Database Repository Collection connects to a table in a database repository, such as *MySQL* or *Oracle*. Once your Database Repository Collection is configured, you can flow information from artifacts in your source collections (either Repository or Gateway Collections) to that table, via an Enterprise Data Stream Integration.

You can learn more about collections in the [Key Concepts](#).

Video Tutorial

Check out the video below to learn how to create a new collection for your database repository:

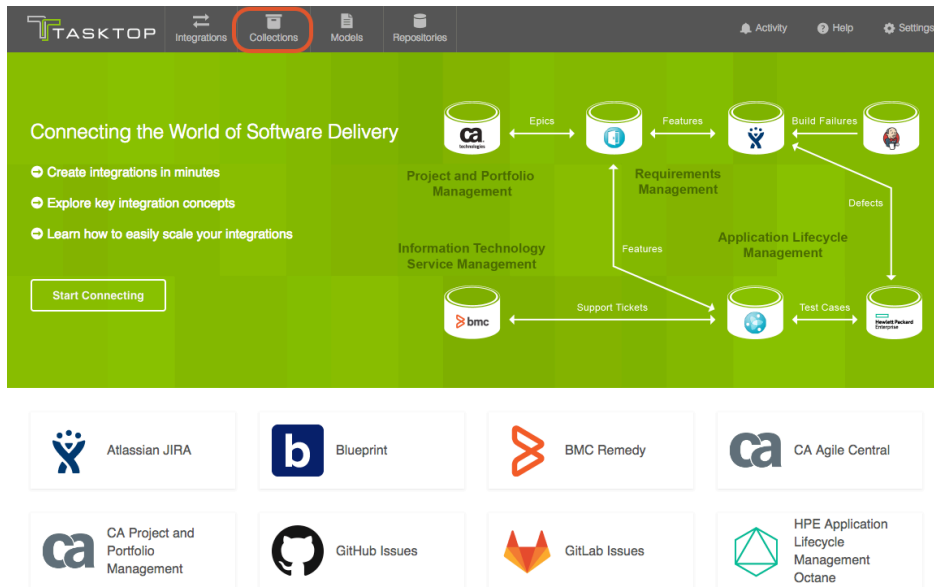
📌 Remember that a Database Collection is a type of Repository Collection

📄 Unknown macro: 'html'

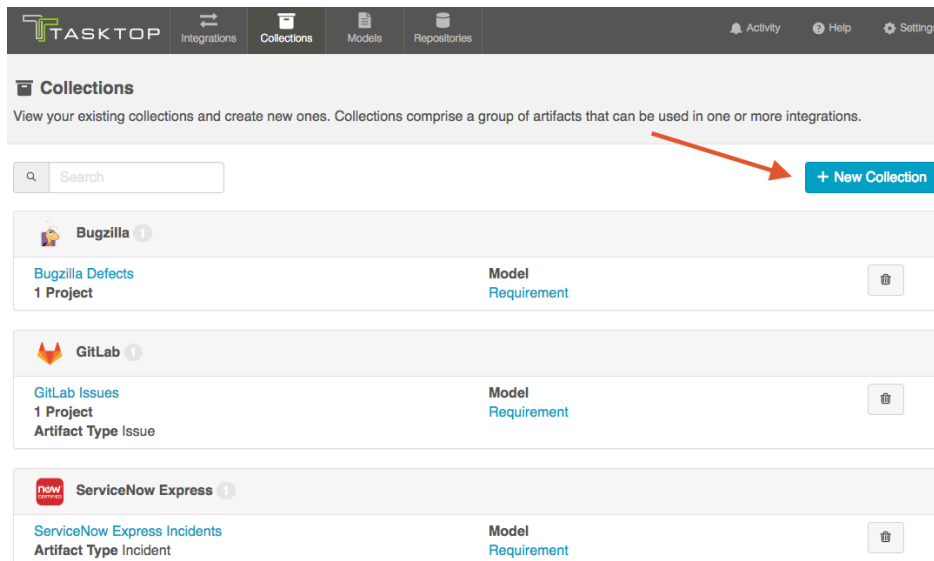
How to Create a Database Collection

To create a database repository collection, follow the steps below:

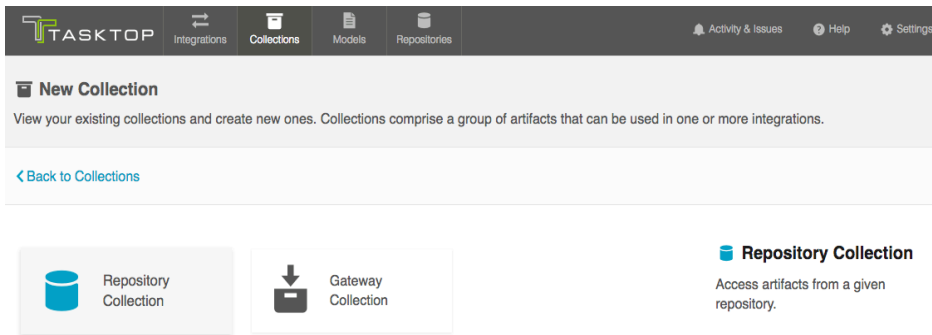
Select 'Collections' at the top of the screen:



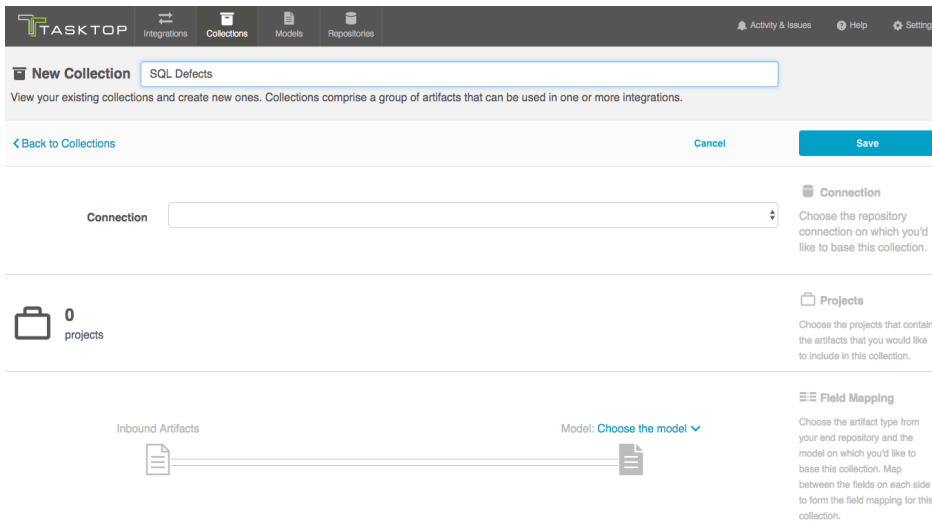
Click 'New Collection':



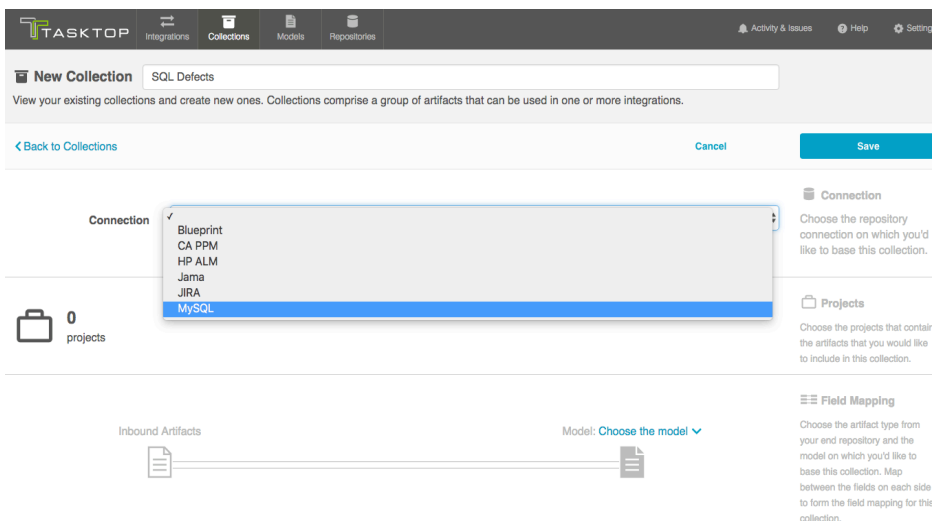
Select "Repository Collection" as the collection type:



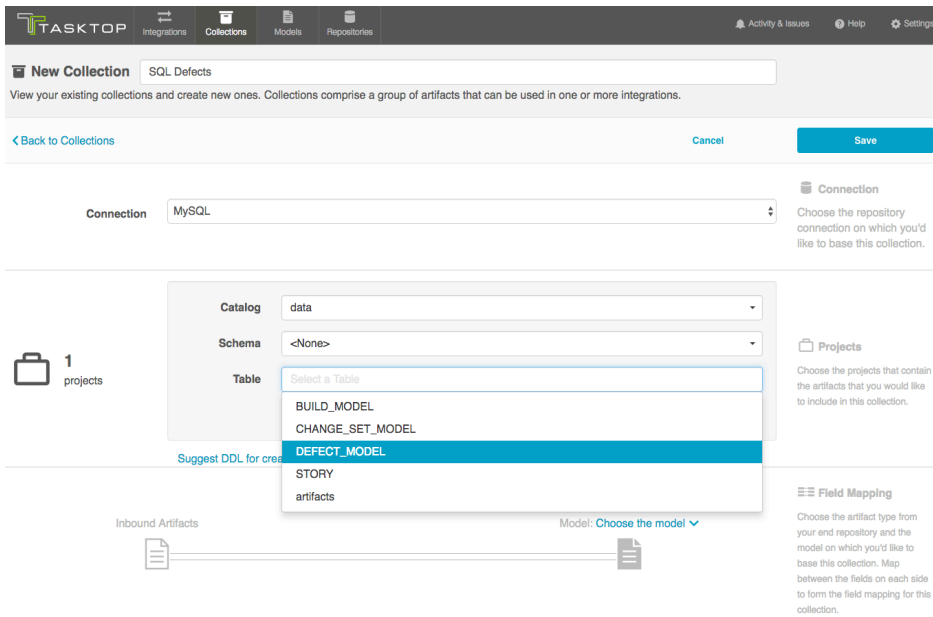
Enter a name for your collection:




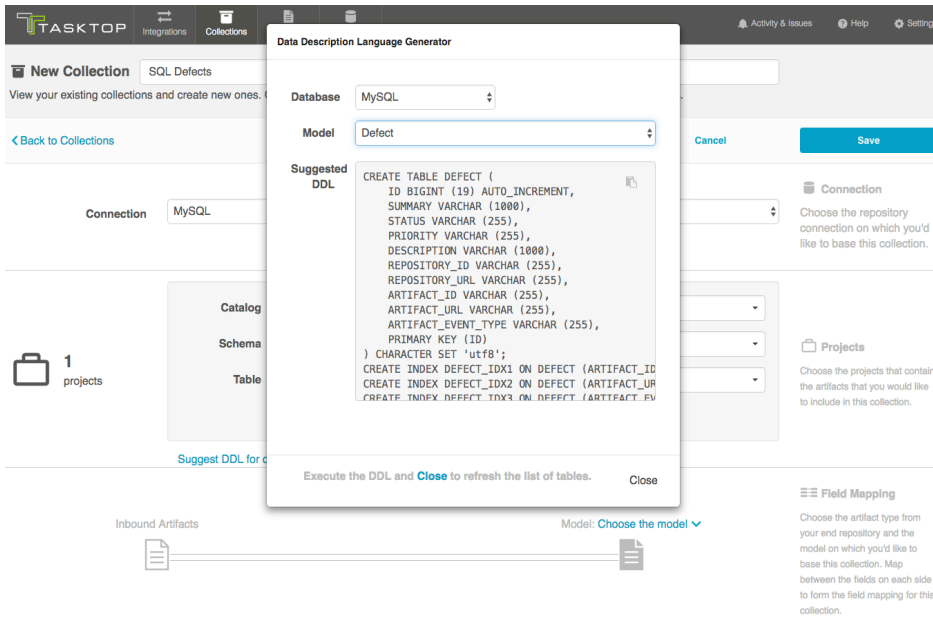
Select the Connection on which you'd like to base this collection. In our example, we are selecting MySQL, which is the 'Tasktop SQL' repository connection we have configured.



Select the database table that will receive artifacts that flow to this collection.



 Note: if your table is not listed, you can use the "Suggest DDL" tool to generate a SQL command that can help you create a table that aligns with the model on which you'd like to base this collection.



Select the Model on which you'd like to base this collection.

Map Fields

Now that you have identified the model, you can complete the collection-to-model field mapping by going into the "Map Fields" link.

Note: If you used the Suggest DDL tool to create your database table, the mapping will be done automatically.

Constant Value Mapping

In some scenarios, the database might require that some of its

columns/fields always have a value. This value is usually provided by mapping it to the equivalent model field. When there is no equivalent field in the model that can provide a value, you can set a constant value into your end-database column/field. The value you configure will then always get written out.

To set a constant value for a field, select the 'Constant Value' option from the drop down menu on the model side. This will tell the integration to *always* flow that value to the database collection. Enter the value, and then click the 'Set Constant Value' box.

Note: Constant values can be set for the following fields types:

- Boolean
- Date/DateTime
- Double
- Location
- Long
- Multi Select
- Person
- Rich Text
- Single Select
- String

Only some of these types are relevant for your database collection, however, given the field types that can be configured in the database itself.

The screenshot shows the TASKTOP interface for field mapping. At the top, there are navigation tabs for Integrations, Collections, Models, and Repositories. The main heading is 'Field Mapping: Defects from Database'. Below this, there are two main sections: 'MySQL: Defects from Database' (2 of 13 fields mapped) and 'Model: Defect' (2 of 4 fields mapped). A 'Field Mapping' section on the right explains that mapping to a common model provides a standard view of the collection to enable integrations with other collections. The interface shows a mapping between 'project (String)' in the MySQL collection and 'project' in the Model. A dropdown menu is open for the 'project' field, showing options: 'Constant Value', 'Description (String)', 'Summary (String)', 'Priority (Single Select)', and 'Severity (Single Select)'. The 'Constant Value' option is selected. Below the dropdown, there are two more mappings: 'priority' in the MySQL collection mapped to 'Priority' in the Model, and 'severity' in the MySQL collection mapped to 'Severity' in the Model. Each mapping has a 'Configure' button next to it. At the bottom right, there are buttons for 'Suggest Mappings', 'Refresh', and 'Clear Workspace'.

Field Mapping: Defects from Database
View your existing collections and create new ones. Collections comprise a group of artifacts that can be used in one or more integrations.

MySQL: Defects from Database (2 of 13 fields mapped) | Model: Defect (2 of 4 fields mapped)

project (String) | Set Constant Value | Constant Value | Suggest Mappings | Refresh

project x | Constant Value: Project Name | Clear Workspace

priority | Priority | Configure

severity | Severity | Configure

Configure Relationships

If you have any relationship(s) fields in your model, you can map those on the "Configure Relationship Types" screen of a given collection.

Note: if you used the Suggested DDL tool to create your database table, the mapping should be done generally.

Collection Configuration MySQL Defects
Configure your collection. Collections comprise a group of artifacts that can be used in one or more integrations.

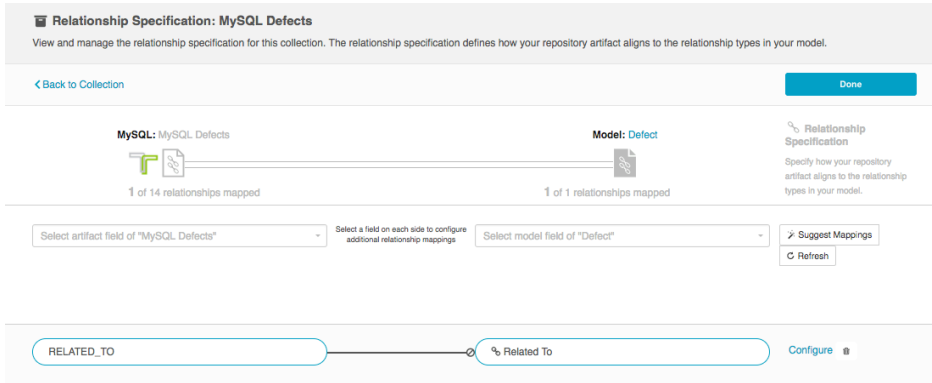
Tasktop SQL Reporting Database
jdbc:mysql://latest5-platform.product.van.tasktop.com:3413

1 projects | Add Project | Suggest DDL for creating a table...

Inbound Artifacts (7 of 13 fields mapped) | Map Fields | Model: Requirement | Relating Model (7 of 15 fields mapped)

Inbound Artifacts (0 of 11 relationships mapped) | Configure Relationships | Relating Model (0 of 5 relationships mapped)

Inbound Person | Person Resolution Strategy | Person Model



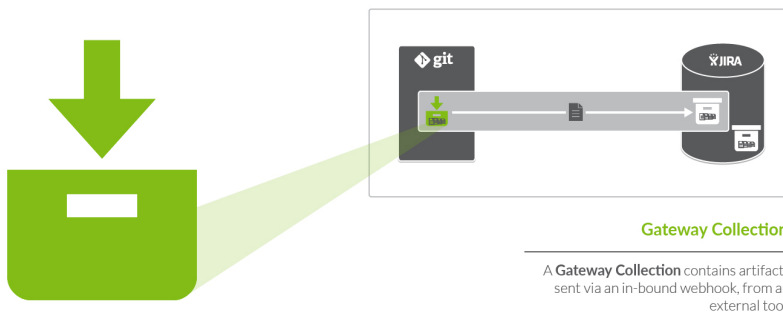
Gateway Collection

Tasktop: 17.4 Release

What is a Gateway Collection?

Gateway Collections are only available in Editions that contain the Gateway add-on. See [Tasktop Editions table](#) to determine if your edition contains this functionality.

- What is a Gateway Collection?
- Video Tutorial
- How to Create a Gateway Collection

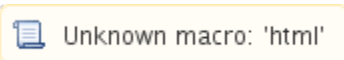


You can think of a *collection* as the set of artifacts that are eligible to flow as part of your integration. A Gateway collection contains artifacts sent via an in-bound webhook, from a DevOps tool.

You can learn more about collections in the [Key Concepts](#).

Video Tutorial

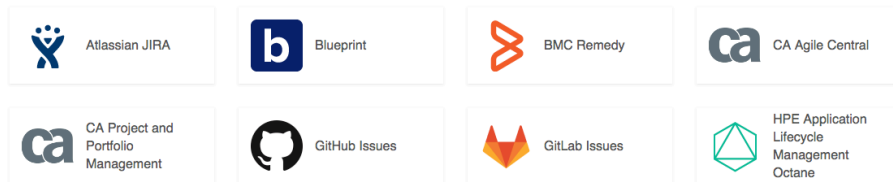
Check out the video below to learn how to create a new gateway collection:



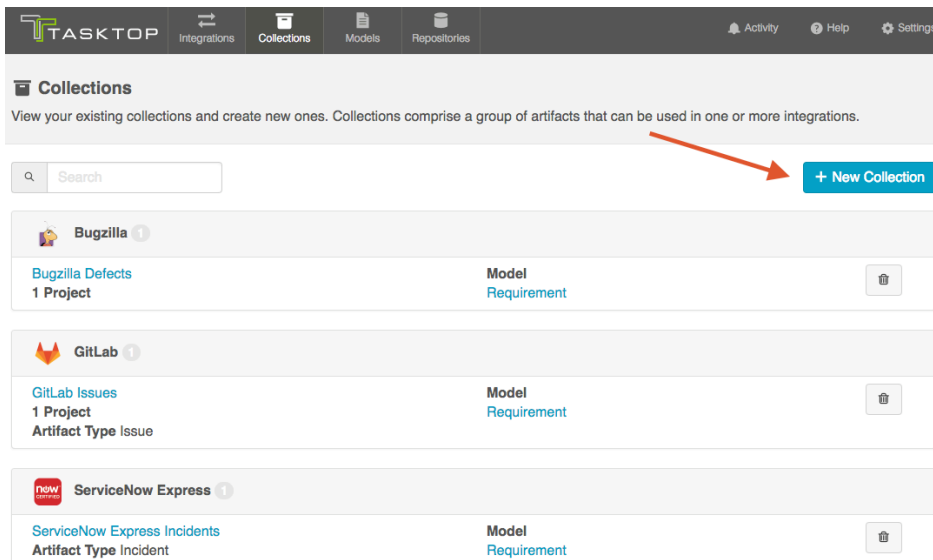
How to Create a Gateway Collection

To create a gateway collection, follow the steps below:

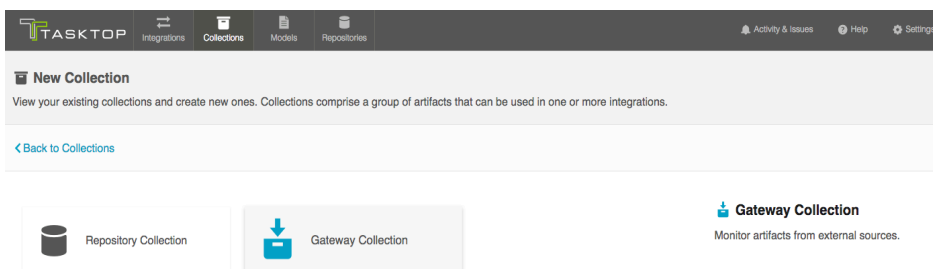
Select 'Collections' at the top of the screen:



Click 'New Collection':



Select "Gateway Collection" as the collection type.



Enter a name for your collection.

Next, specify the *path* for your collection. These characters will form the REST endpoint to which you can send artifacts to Tasktop via this gateway collection.

💡 Upon first creating your Gateway collection, Tasktop will populate path with the name that you have given to your collection. You can change this if desired.

To secure your gateway collection, Tasktop automatically appends a token (a universally unique identifier) to the path of a gateway collection. This token will be incorporated into your gateway URL and help ensure that only users that know the full path with its token can access your gateway collection.

You can remove the token by clicking the trash can icon to the right, and refresh it by hitting the magic wand icon that appears in its place. Once refreshed, click 'save,' and the URL will be updated.

Select the Model on which you'd like to base this collection.

NEW COLLECTION Gateway Defects

View your existing collections and create new ones. Collections comprise a group of artifacts that can be used in one or more integrations.

< Back to Collections Cancel **Save**

Path:

Token:

Model:

- Build Model
- ChangeSet Model
- Defect**
- Story
- Time Entry

If you have configured a [payload transformation extension](#) for your Gateway collection on the Settings screen, you can select it here.

NEW COLLECTION Gateway Defects

Select your collection type, then start configuring your new collection. Collections comprise a group of artifacts that can be used in one or more integrations.

< Back to Collections Cancel **Save**

Path:

Token:

Model:

Payload Transformation:

- Payload Transformation Extension**
- None

If you have any relationship(s) fields in your model, you'll need to identify a target repository for each. This will ensure that enough information is being sent in via the Gateway to uniquely locate the artifact you'd like to relate to.

COLLECTION Gateway Changesets

View your existing collections and create new ones. Collections comprise a group of artifacts that can be used in one or more integrations.

< Back to Collections **Done**

Path:

Token:

Model:

Relationship Field Configuration

Related Stories:

- Select a repository
- Blueprint** b
- CA PPM ca
- HP ALM =
- Jama j

Access Details

Url: CA PPM
CA Project and Portfolio Management

Method: HP ALM
HPE QC / ALM

Content-Type: Jama

Example Payload:

```
{
  "summary": "String",
  "priority": "Trivial",
  "web_url": "String",
  "committer": "userId",
  "related_stories": []
}
```

Example Script:

```
curl -H 'Content-Type: application/json' --data-binary '{"summary":"String","priority":"Trivial","web_url":"S'
```

Once you've saved your collection, you will be able to observe the access details given for this gateway collection:


The screenshot shows the Tasktop interface for a collection named 'Defects'. The top navigation bar includes 'Integrations', 'Collections', 'Models', and 'Repositories'. The main content area shows the collection name 'Defects' and a description: 'View your existing collections and create new ones. Collections comprise a group of artifacts that can be used in one or more integrations.' Below this, there are fields for 'Path' (http://latest2-platform.product.van.tasktop.com/api/v1/artifacts/defects), 'Token' (bbb56062-1afa-4812-897e-9f8c2b5a8bcd), and 'Model' (Defect). Under the 'Access Details' section, the 'Uri' is the same as the path, the 'Method' is POST, and the 'Content-Type' is application/json. An 'Example Payload' is shown as a JSON object: {"summary": "String", "status": "In Progress", "priority": "High", "description": "String"}. An 'Example Script' is provided: curl -H 'Content-Type: application/json' --data-binary '{"summary":"String","status":"In Progress","priority":'.





Step 4: Configure your Integration

Tasktop: 17.4 Release

Types of Integration Templates

Tasktop offers a range of Integration Templates to enable you to achieve a diverse set of goals:

 Unknown macro: 'html'

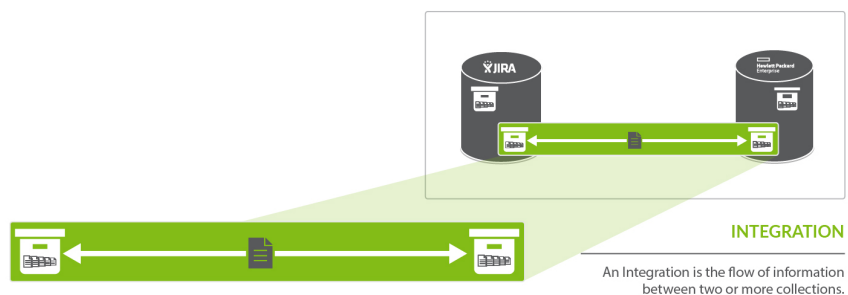
 Synchronize Integration Template	 Create via Gateway	 Modify via Gateway	 Enterprise Data Stream
--	---	---	---

<p><i>The Synchronize Integration Template is available in all Editions.</i></p>	<p><i>The Create via Gateway Template is only available in Editions that contain the Gateway add-on. See Tasktop Editions table to determine if your edition contains this functionality.</i></p>	<p><i>The Modify via Gateway Template is only available in Editions that contain the Gateway add-on. See Tasktop Editions table to determine if your edition contains this functionality.</i></p>	<p><i>The Enterprise Data Stream Template is only available in Editions that contain the Enterprise Data Stream add-on. See Tasktop Editions table to determine if your edition contains this functionality.</i></p>
<p>This integration connects teams working in different tools as they fulfill their roles in the software development lifecycle. As part of this integration, artifacts will flow between separate repository collections.</p>	<p>This integration creates traceability between artifacts across the software development lifecycle. New artifacts will be created in a repository collection when artifacts are sent to Tasktop via a Gateway collection, through an inbound webhook.</p>	<p>This integration creates traceability between artifacts across the software development lifecycle. Already existing artifacts in a repository collection will be located and modified in a specified way when artifacts are sent to Tasktop via a Gateway collection, through an inbound webhook.</p>	<p>This integration simplifies enterprise reporting by unlocking software lifecycle data from its application tool silos and providing a rich data repository for near real-time analytics. Records will be created in a single database when artifacts from one or more collections are created or changed.</p>
<p>Learn More</p>	<p>Learn More</p>	<p>Learn More</p>	<p>Learn More</p>

Synchronize Integration Template

Tasktop: 17.4 Release

[What is an Integration?](#)



An *integration* is quite simply the flow of information between two or more collections. When you configure your integration, you can customize the field flow, artifact routing, artifact filtering, as well as enable or disable comment flow or attachment flow.

[Video Tutorial](#)

- [What is an Integration?](#)
- [Video Tutorial](#)
- [Use Case and Business Value](#)
- [Template Affordances](#)
- [How to Configure a Synchronize Integration](#)
 - [Getting Started](#)
 - [Artifact Creation Flow](#)
 - [Field Flow](#)
 - [Field Flow Direction and Frequency](#)
 - [Field Flow](#)

Check out the video below to learn how to configure a Synchronize Integration.

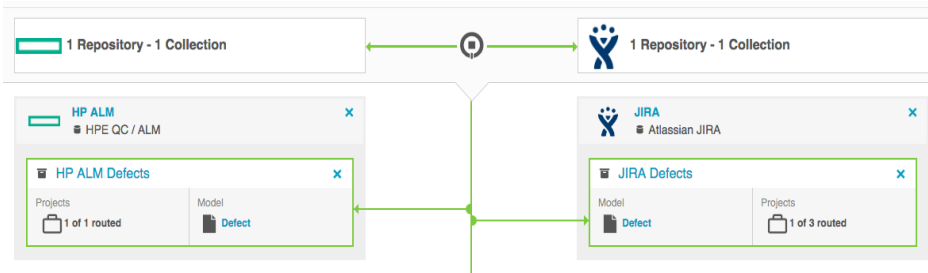
⚠ This video assumes that you have already configured your repositories, models, and collections as outlined in the [Quick Start Guide](#).



Use Case and Business Value

The Synchronize Integration Template connects teams working in different tools as they fulfill their roles in the software development lifecycle.

As part of this integration, artifacts will flow between disparate repository collections. You can choose to have either one-way or two-way artifact creation. Artifacts created in repository collection A can create corresponding artifacts in repository collection B, artifacts in repository B can create corresponding artifacts in repository collection A, or both can occur in the same integration. You'll also configure the direction in which each field on those artifacts should be updated.



Template Affordances

The Synchronize Integration Template allows you to flow artifacts between two repository collections.



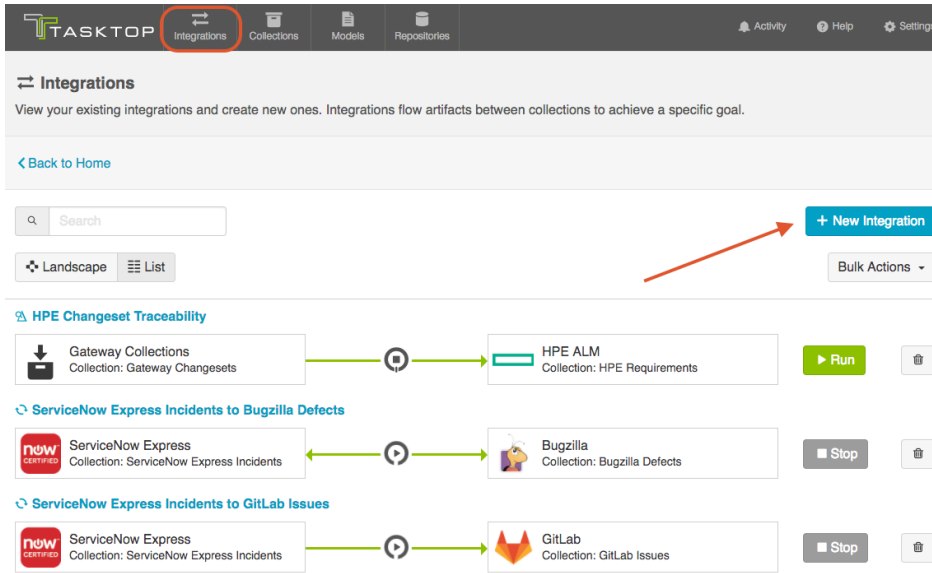
How to Configure a Synchronize Integration

Getting Started

Now that you have all of your base components set up, you can configure integrations to connect the artifacts in your collections.

To configure your integration, select 'Integrations' at the top of the screen, then click 'New Integration.'

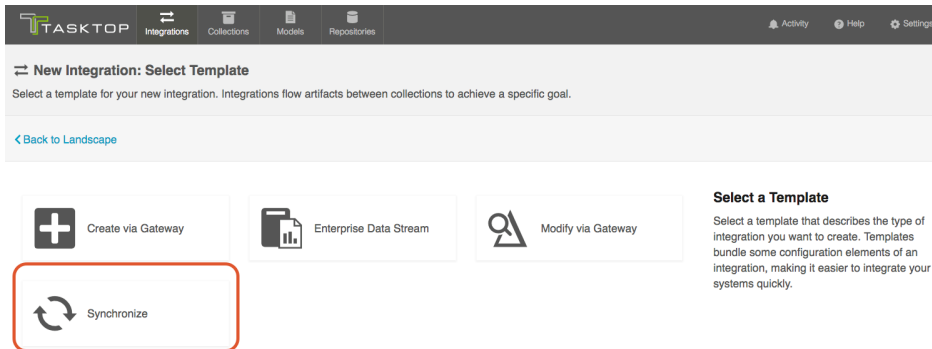
- Icons
- Process All Artifacts
- Artifact Routing
 - Static Artifact Routing
 - Conditional Artifact Routing
- Artifact Filtering
 - Artifact Creation vs. Artifact Update
 - Create Artifact Filter Statements
 - Viewing Artifact Filter Statements
 - Filtering via Repository Queries
- Comment Flow
 - Comment Impersonation
- Attachment Flow
- Conflict Resolution Strategy
- Running your Integration
 - From the Integration Configuration Screen



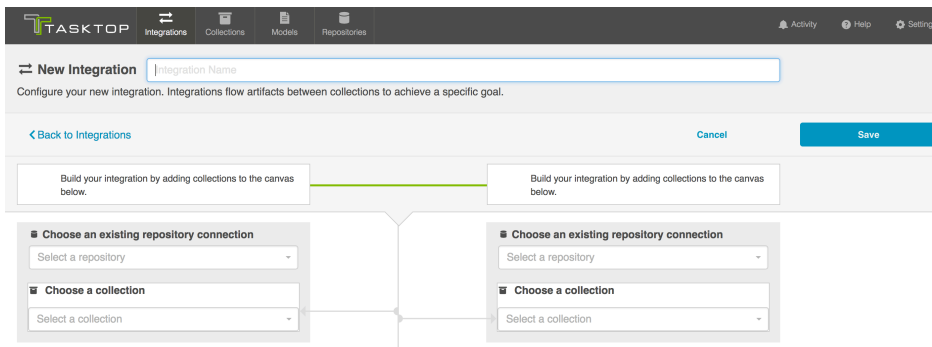
- From the Integrations List Page
- Viewing Your Integrations
 - Landscape View
 - List View
- Tips and Tricks
 - Synchronizing Internal Relationships
 - Synchronizing an Artifact ID or URL Reference

Select your desired integration template from the options available.

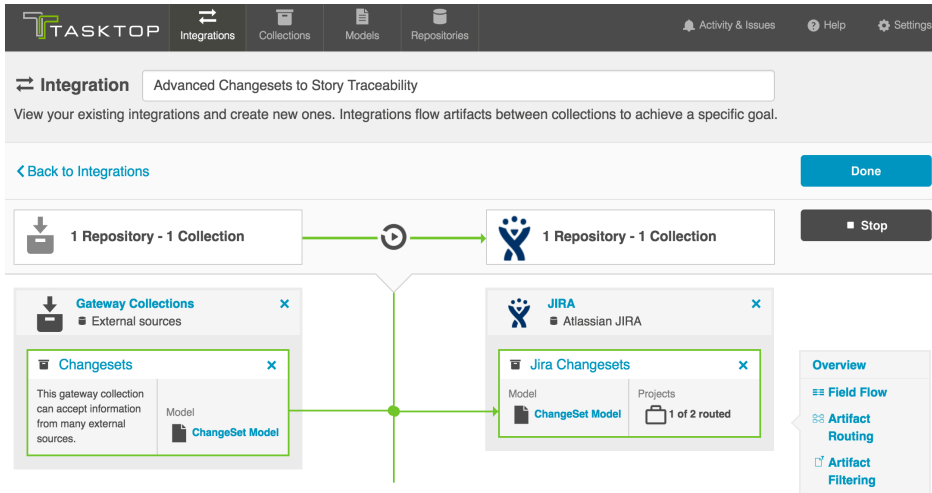
💡 Depending on the edition of Tasktop you are utilizing, you may not have all options available.



This will bring you to the New Integration Screen:

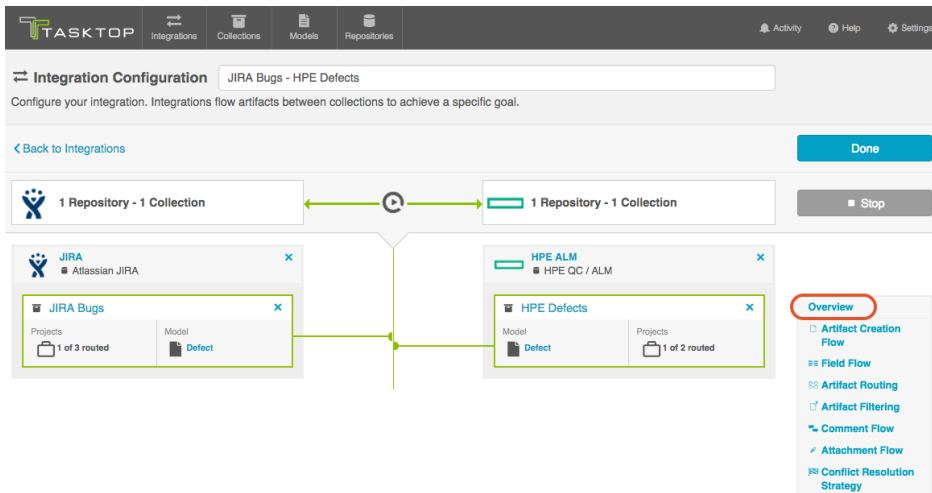


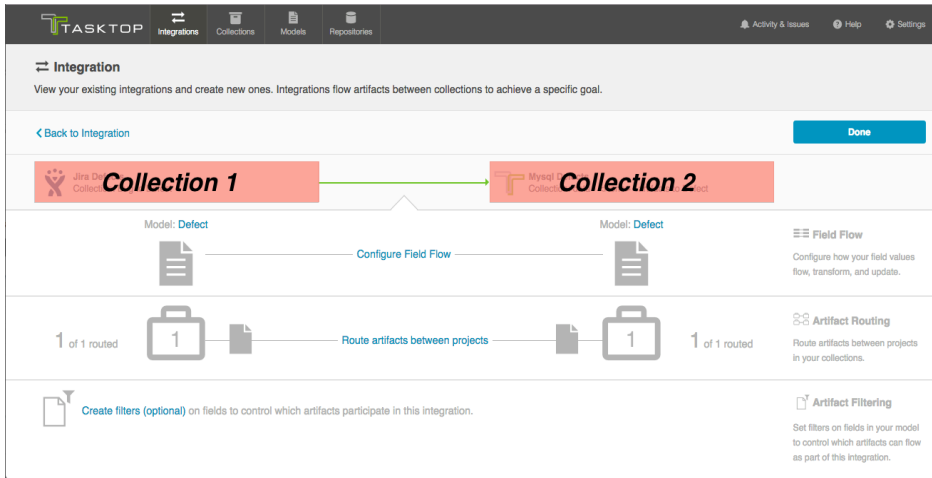
Name your integration and select your repositories and collections:



While each template might have some special steps and affordances (which are detailed in the help section for a given integration template), the general configuration components of an integration are described below.

You can click the 'Overview' link on the right side of the Integration Page to get to the main display page (shown in the second screen shot).



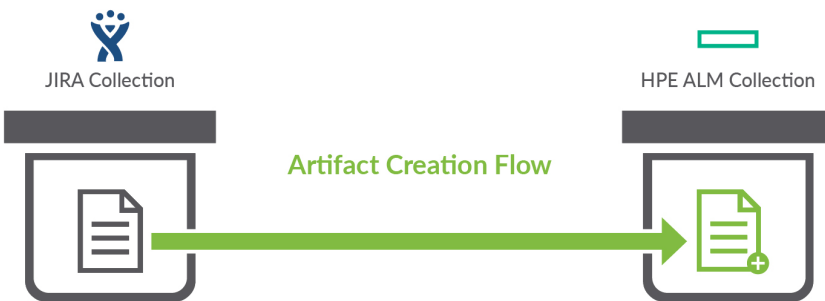


Artifact Creation Flow

The Synchronize Integration is unique in that it allows the user to determine whether to create artifacts in both repositories, or to create artifacts in just one of the two repositories.

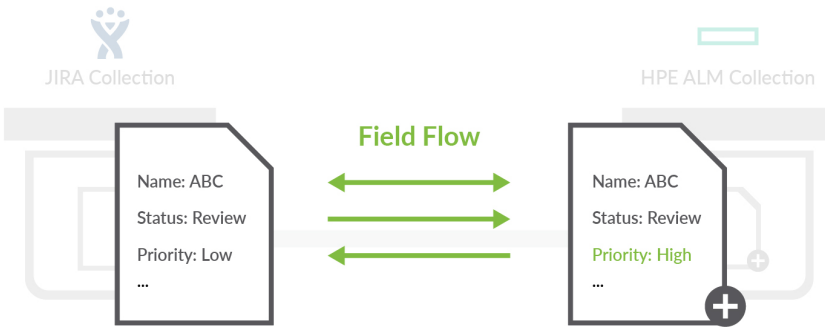
Note that this setting relates only to the creation of artifacts (as opposed to the modification of fields on those artifacts). So for example, if I chose to set up one-way *artifact creation flow* from JIRA to HPE ALM, this means that when the integration is run, new or existing artifacts from JIRA will create new artifacts in HPE ALM, but new or existing artifacts from HPE ALM will not create new artifacts in JIRA.

However, once a JIRA artifact creates a target artifact in HPE ALM, if any modifications are made to that artifact in HPE ALM, that modification could flow back over to JIRA, based on the integration's *field flow* configuration. So while the integration is not creating new artifacts in JIRA, it can modify existing artifacts based on corresponding changes made in HPE ALM.



Artifact Creation Flow

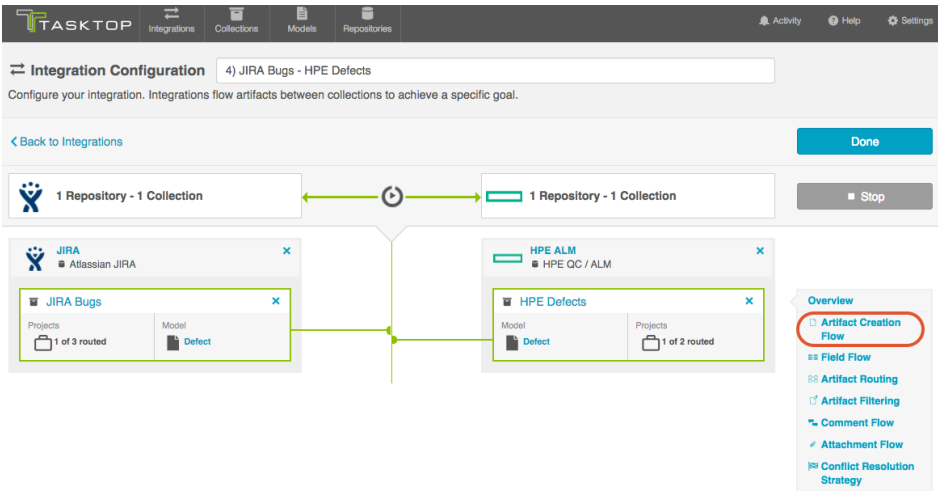
Based on the **Artifact Creation Flow** configured above, artifacts in JIRA will create new artifacts in HPE ALM, but artifacts in HPE ALM will **not** create new artifacts in JIRA.



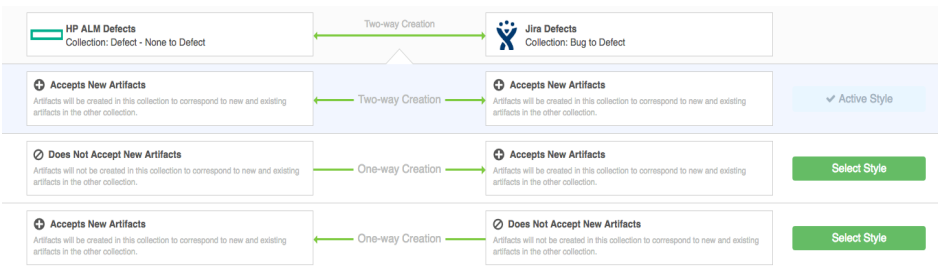
Field Flow

Note that **Field Flow** is set independently for each field pair, and does not need to match the configuration for **Artifact Creation Flow**. In the example above, if the priority on our HPE ALM artifact is changed from Low to High, that updated field value will flow back to JIRA.

Here's how to configure the Artifact Creation Flow:
 From the Integration page, select 'Artifact Creation Flow'



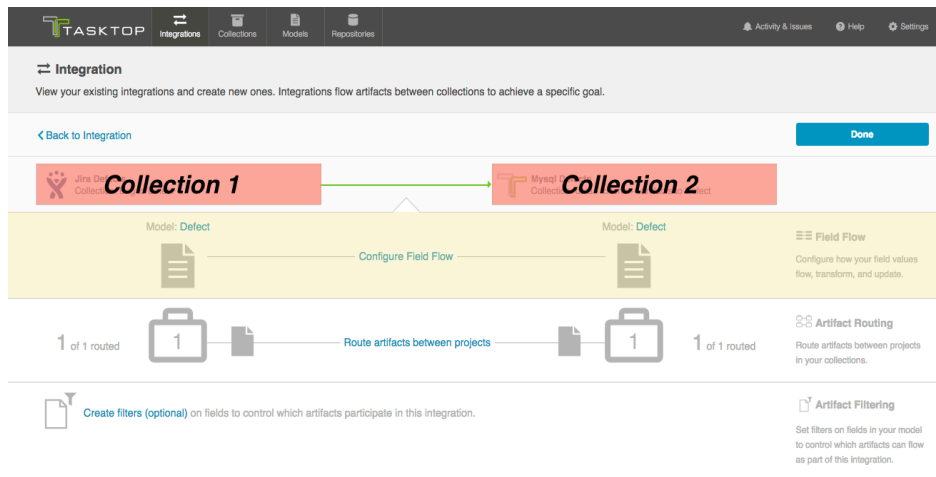
This will lead you to the Artifact Creation Flow page, where you will be able to select Two-way Creation (artifacts will be created in both collections to correspond to new and existing artifacts in the other collection), or One-way Creation (only one of the two repositories will have new artifacts created to correspond to new and existing artifacts in the other collection).



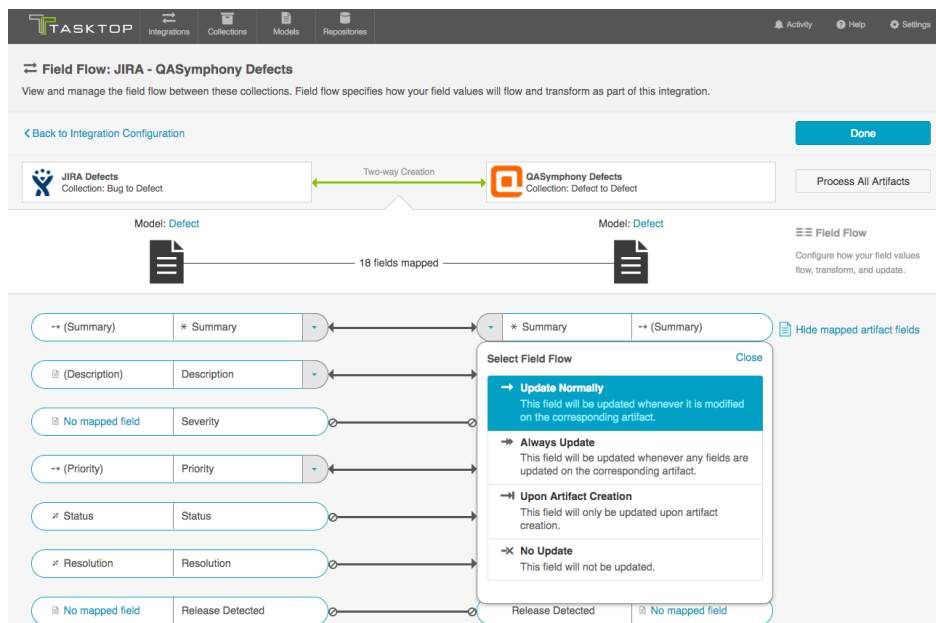
On the Field Flow screen, you can configure the following:

- the direction fields flow in
- the frequency with which they flow (i.e. only upon creation vs. always updating)

To get to the Field Flow screen, click 'Configure Field Flow':

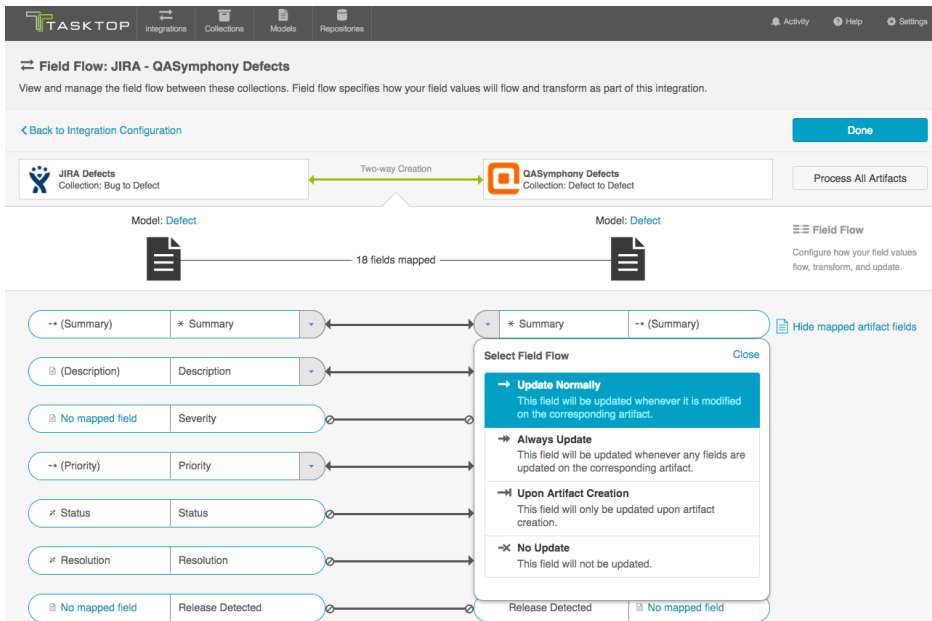


You will be directed to the Field Flow screen:



You can see the names of the mapped artifact fields for each collection on the far left and far right, with the model fields displayed in the middle. To hide the mapped artifact fields, select 'Hide mapped artifact fields' on the right.

Field Flow Direction and Frequency









Here, you can specify the direction fields flow in, as well as their frequency:





Icon	Meaning
→	Update Normally: This field will be updated whenever it is modified on the corresponding artifact
⇒	Always Update: This field will be updated whenever any fields are updated on the corresponding artifact
→	Upon Artifact Creation: This field will only be updated upon artifact creation
→X	No Update: This field will not be updated

⚠ Note: The field flow settings behave a bit differently for Constant Values. This is because constant values exist as part of your Tasktop configuration, and not on the artifact itself. Therefore, changes in constant values are not detected in the same way that updates made on the actual artifact are detected. If you change the constant value that is linked to your model, your integration will not automatically detect this update and sync it over. The value will only update if another field on that artifact is updated. Because of this, for constant values, "update normally" and "always update" will behave identically: meaning that the constant value will update whenever any other field is updated on that artifact.

Field Flow Icons

On the Integration Field Flow page, you will see a number of icons, which will help you understand any special properties or requirements for each field. If you hover your mouse over an icon, you will see a pop-up explaining what the icon means. You can also review their meanings in the legend below:

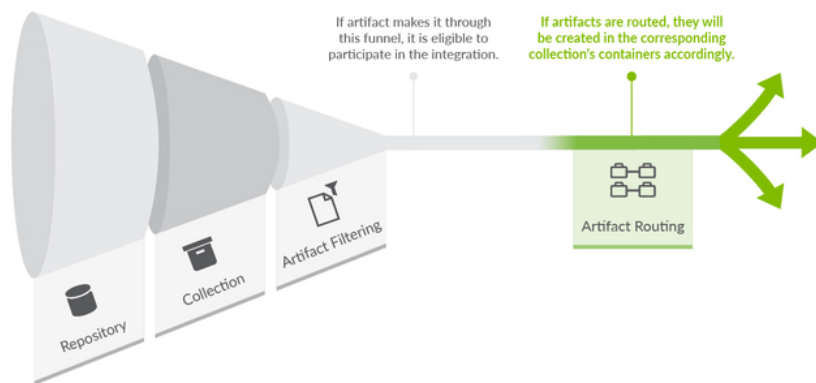
Icon	Meaning
	<p>A constant value will be sent.</p> <p>Note that:</p> <ul style="list-style-type: none">• If the icon is on the side of the collection, this means that a constant value will be sent to your model. This means that any time this collection is integrated with another collection, the <i>other</i> collection will receive this constant value for the field in question.• If the icon is on the side of the model, this means a constant value will be sent to your collection. This means that any time this collection is integrated with another collection, that <i>this</i> collection will receive this constant value for the field in question.
	Collection field is read-only and cannot receive data
 	To create artifacts in your collection, this field must be mapped to your model.
	This is a required field in your model; it must be mapped to your collection.
	This field will not be updated as part of your integration, due to how you have configured it. This field flow configuration can be changed if you'd like.

	<p>This field will not be updated as part of your integration because the mapping would be invalid. You do not have the option of changing this.</p>
	<p>This field will update normally as part of your synchronize integration; this means it will be updated whenever it is modified on the corresponding artifact.</p>
	<p>This field will always update as part of your synchronize integration; this means that it will be updated whenever <i>any</i> fields are modified on the corresponding artifact.</p>
	<p>This field will only be updated upon initial artifact creation.</p>

Process All Artifacts

The 'Process All Artifacts' button will prompt Tasktop to process all artifacts in the integration. Any changes or additions you've made to your collection-to-model mappings will be applied to all artifacts participating in the integration. This functionality can be useful when adding a new field to your field flow configuration. You can learn more about this process [here](#).

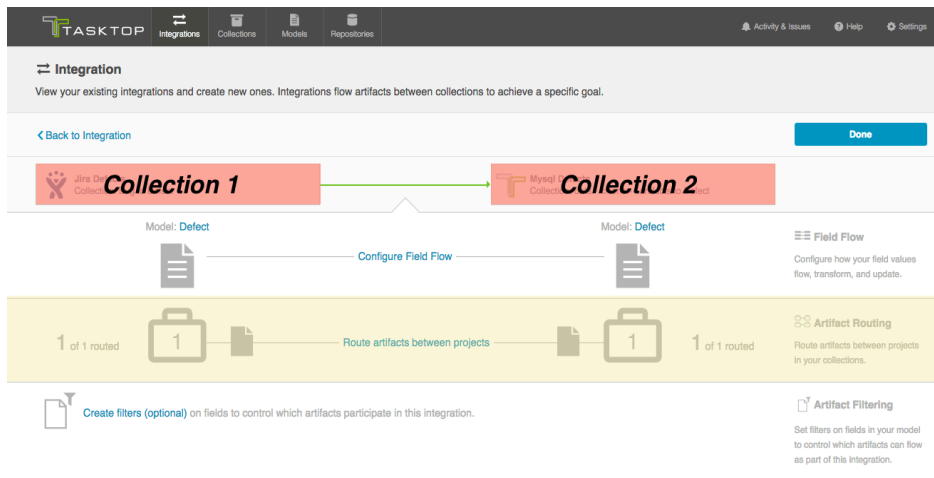
Artifact Routing



Artifact Routing is needed when artifacts are being created as part of an integration. In addition to knowing the repository in which artifacts should be created, Tasktop also needs to know which container (i.e. project, module, folder, etc) a given artifact should be created in. Specifying the artifact routing does this.

Initially, the artifact routing will determine where an artifact gets created. Over time, if an artifact on either side moves, we will move the artifact to the corresponding container of the new route, if this is allowed in your repository. If you are moving between lower-level containers, such as sets or folders, this is generally possible. However, we will not do so if the move on one side crosses the bounds of the top-level container (generally the high-level container, added at the collection level).

To configure Artifact Routing, select 'Route artifacts between projects' on the Integration Overview screen, or 'Artifact Routing' on the right pane of the Integration Configuration screen.

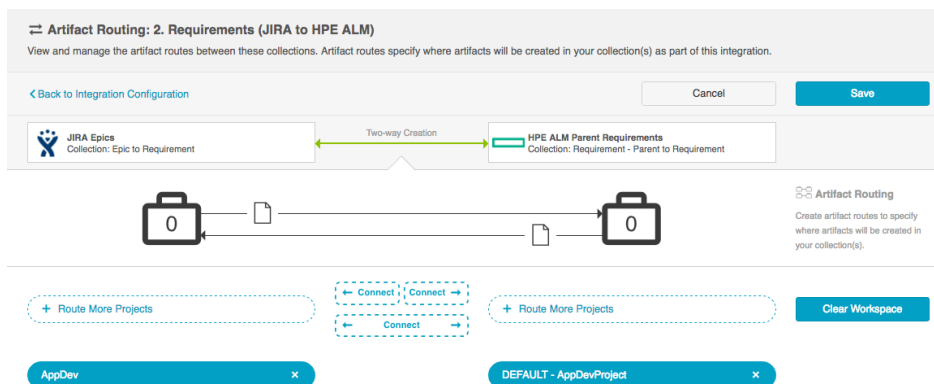


Static Artifact Routing

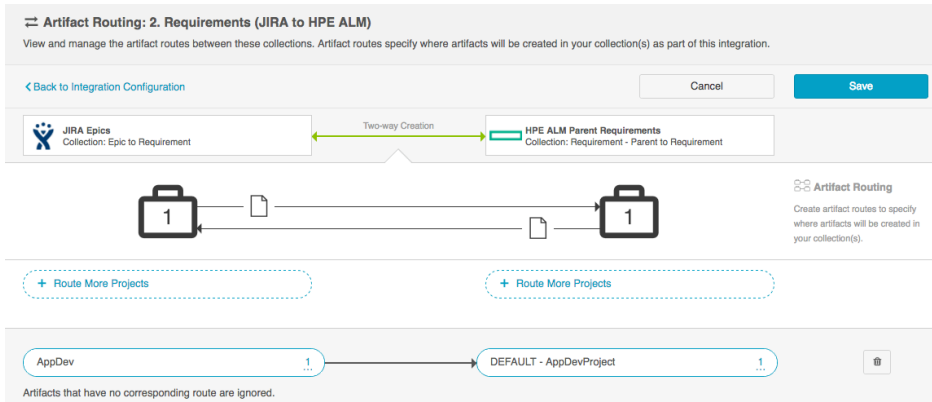
In some cases, the project an artifact is in on one side can sufficiently determine which project an artifact should be created in in the corresponding collection. In these instances, you can configure what is known as 'static artifact routing' (also known as 'explicit artifact routing').

Static artifact routes can have one or more source projects, but only a single target project.

To configure a static artifact route, use the "Route More Projects" buttons to add projects from your collections to your working space and connect them using the "Connect" button. The directionality on the connect button refers to artifact creation.

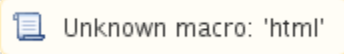


In the example shown below, artifacts from the JIRA AppDev project will be created in AppDevProject in HPE ALM.



Conditional Artifact Routing

Check out the video below to learn more about Conditional Artifact Routing:



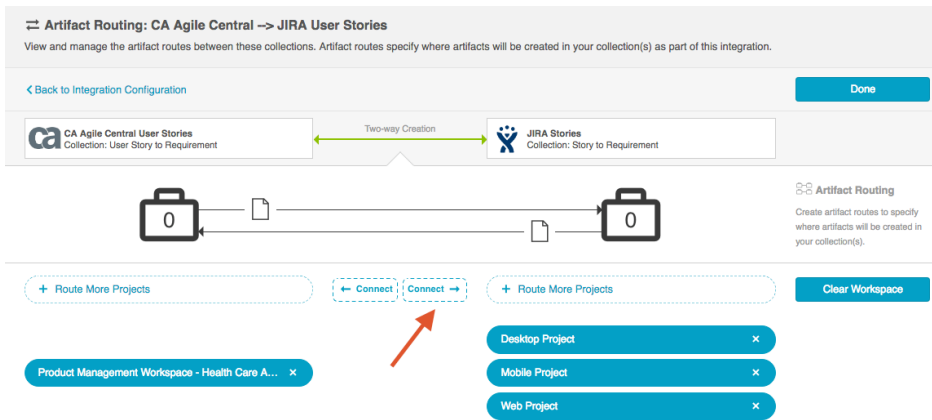
⚠ Note: The video above demonstrates Conditional Artifact Routing within the context of a Create via Gateway Integration. Create via Gateway Integrations are only available in editions that contain the Gateway add-on. See [Tasktop Editions table](#) to determine if your edition contains this functionality. Though the video is for a Gateway Integration, the core concepts outlined in the video can be applied to any integration template.

In some cases, the project an artifact is in within the source repository does not provide enough information to determine which project the artifact should be created in within its target repository. Oftentimes, in fact, some unique characteristic of an artifact, such as a specific field value, is the factor that should be used to determine which project an artifact should be created in within the target repository.

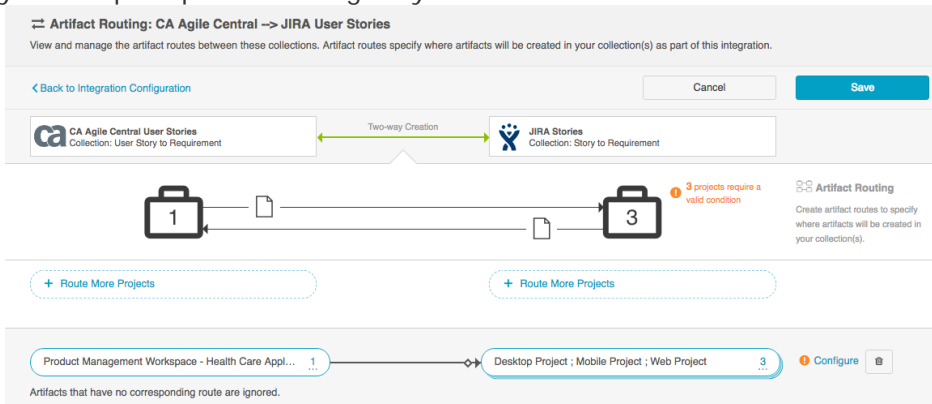
In these instances, you will configure what is known as conditional artifact routing to determine which project each artifact is created in within your target repository. Conditional artifact routing (also known as 'dynamic artifact routing') can be used to inspect a single-select field of an artifact and, depending on its value, to route that artifact to the appropriate project in the target collection.

Conditional artifact routes can have one or more source projects, and always have multiple target projects.


To create a conditional artifact route, use the "Route More Projects" buttons to add projects from your collections to your workspace and connect them using the "Connect" button.



Notice that after you've created your conditional artifact routing group, you'll be prompted to configure your route.



Click 'Save,' and then click 'Configure.' You'll be brought to the Conditional Artifact Routing screen. Here you'll start by selecting the model field on the artifact that you would like to use to determine your artifact route.

 Note: Conditional Artifact Routes can only be configured based on single-select fields in your model.

In the example below, the field "Application" contains the unique values that should determine the project an artifact will be created in in JIRA.

Conditional Artifact Routing:: CA Agile Central -> JIRA User Stories
View and manage the conditions that determine this artifact route. Artifact routes specify where artifacts will be created in your collection(s) as part of this integration.

[Back to Artifact Routing](#) Done

CA Agile Central User Stories
Collection: User Story to Requirement

Two-way Creation

JIRA Stories
Collection: Story to Requirement

0 conditions configured 3 projects require a valid condition

Product Management Workspace - Health Care Appl... → Desktop Project ; Mobile Project ; Web Project

Configure which condition the artifacts must match to flow to a project on the right.

Model field that contains the unique routing identifier:

Specify Model Field...

- Priority
- Release
- Application**

Target project to receive artifacts based on a condition:

- Desktop Project
- Mobile Project
- Web Project

Specify handling for artifacts not matched by conditions above:

- Error: Artifacts that do not meet any of the conditions specified will result in an error.
- Ignore: Artifacts that do not meet any of the conditions specified will be ignored.
- Default Route: Artifacts that do not meet any of the conditions specified will be routed to a particular project.

Route unidentified artifacts from this side → Select a Target Project

After you select the model field, you can identify one or more value to correspond to each target project. You can also use the 'Manage Values' link to select from a list of values.

Conditional Artifact Routing:: CA Agile Central -> JIRA User Stories
View and manage the conditions that determine this artifact route. Artifact routes specify where artifacts will be created in your collection(s) as part of this integration.

[Back to Artifact Routing](#) Cancel Save

CA Agile Central User Stories
Collection: User Story to Requirement

Two-way Creation

JIRA Stories
Collection: Story to Requirement

2 conditions configured 1 project requires a valid condition

Product Management Workspace - Health Care Appl... → Desktop Project ; Mobile Project ; Web Project

Configure which condition the artifacts must match to flow to a project on the right.

Model field that contains the unique routing identifier:

Application

Model field equals one or more unique values:

Desktop [Manage Values](#) → Desktop Project

Mobile [Manage Values](#) → Mobile Project

Specify Unique Values... **Web** → Web Project

Once you've done this, you'll see your full conditional artifact routing group:

Conditional Artifact Routing:: CA Agile Central --> JIRA User Stories
View and manage the conditions that determine this artifact route. Artifact routes specify where artifacts will be created in your collection(s) as part of this integration.

[Back to Artifact Routing](#) Cancel Save

CA Agile Central User Stories
Collection: User Story to Requirement

Two-way Creation

JIRA Stories
Collection: Story to Requirement

3 conditions configured All projects can receive artifacts.

Product Management Workspace - Health Care Appl... Desktop Project ; Mobile Project ; Web Project

Configure which condition the artifacts must match to flow to a project on the right.

Model field that contains the unique routing identifier:
Application

Model field equals one or more unique values:

Desktop x Manage Values Desktop Project

Mobile x Manage Values Mobile Project

Web x Manage Values Web Project

In this example:

- User Stories from CA Agile Central with a value of "Desktop" in the Application field will cause the creation of a corresponding Story in the Desktop Project in JIRA,
- User Stories from CA Agile Central with a value of "Mobile" in the Application field will cause the creation of a corresponding Story in the Mobile Project in JIRA, and
- User Stories from CA Agile Central with a value of "Web" in the Application field will cause the creation of a corresponding Story in the Web Project in JIRA.

You can specify how you'd like to handle User Stories from CA Agile Central that do not meet any of the conditions specified (for instance, if its value for "Application" is "Other",) by selecting one of the options provided at the bottom of the screen:

Specify handling for artifacts not matched by conditions above:

Error: Artifacts that do not meet any of the conditions specified will result in an error.

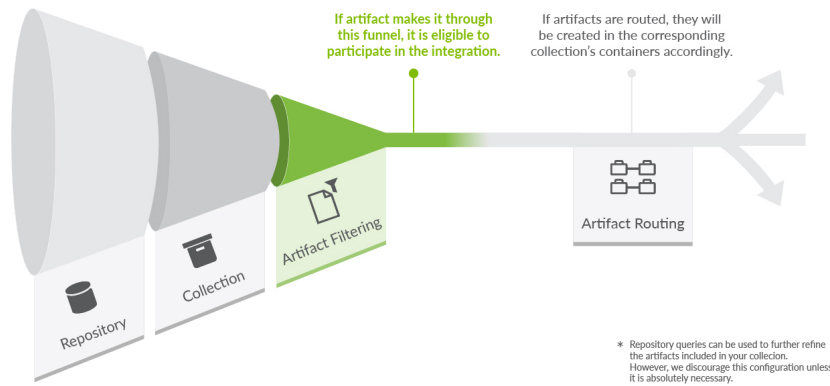
Ignore: Artifacts that do not meet any of the conditions specified will be ignored.

Default Route: Artifacts that do not meet any of the conditions specified will be routed to a particular project.

Route unidentified artifacts from this side Select a Target Project

Artifact Filtering

When configuring your integration, you have several options available to refine which artifacts are eligible to flow. The final mechanism available is *artifact filtering*, which is configured at the Integration level.



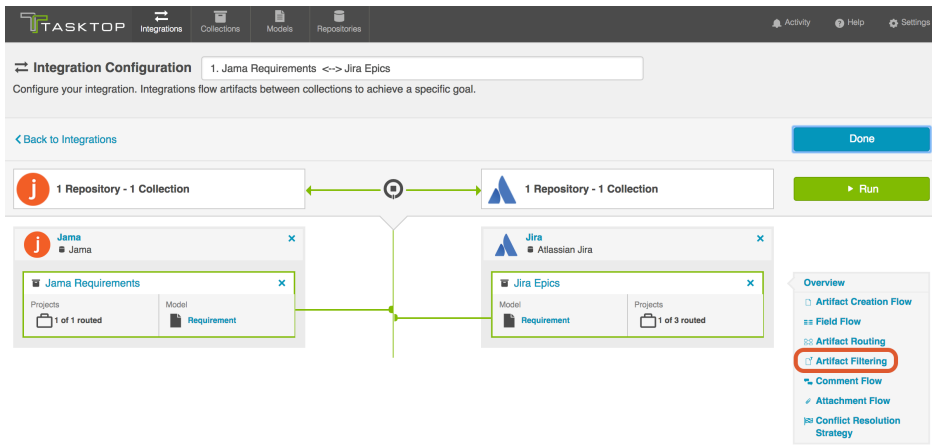
Artifact Filtering enables you to set filters on an integration in order to limit which artifacts are eligible to flow in your integration.

To use a field for artifact filtering, it must:

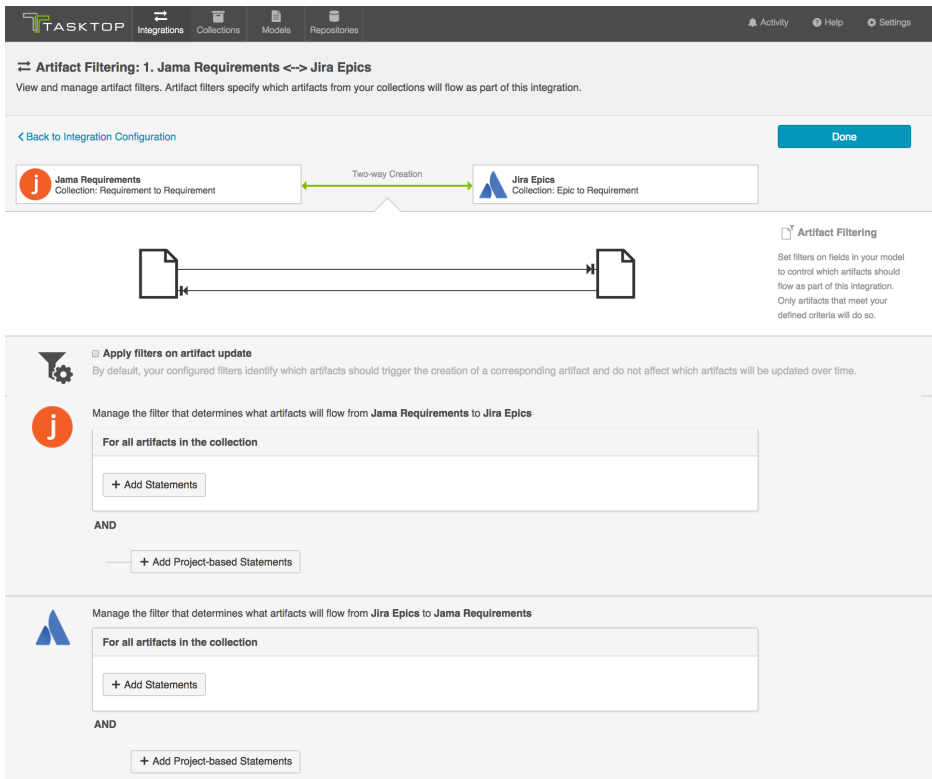
- Be a part of your model, and be mapped to the collection you are filtering from
- Be one of the following field types:
 - Single Select
 - Note that in cases where 'allow unmapped values to flow' is enabled in the model, only fields that are already a part of the model will be considered for artifact filtering
 - Multi-Select
 - Note that in cases where 'allow unmapped values to flow' is enabled in the model, only fields that are already a part of the model will be considered for artifact filtering
 - Date
 - Date/Time
 - Duration
 - String

💡 Note that you can utilize our transforms to filter based on an 'unsupported' collection field type, if that field is mapped to a supported field type in your model. For example, you could filter based on a Boolean field in your repository, if that boolean field is mapped to a single select field in your model.

To configure Artifact Filtering, click the 'Artifact Filtering' link:



This will lead you to the Artifact Filtering screen, where you can configure your artifact filters.



Artifact Creation vs. Artifact Update

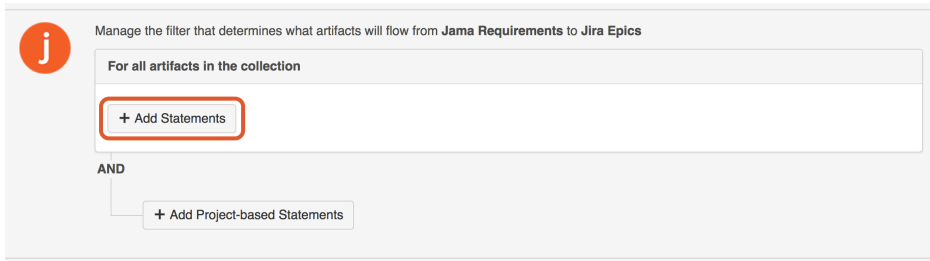
First, determine whether you'd like your filter to apply to artifact creation, or to artifact creation *and* artifact update. By default, your filter will apply only to artifact creation. This means that once artifacts are synchronized, updates will continue to flow between them, even if values are changed that make it such that they no longer meet your filtering criteria. In most scenarios, this will be the desired outcome, as it ensures that your source and target artifacts will stay in sync with one another.

Create Artifact Filter Statements

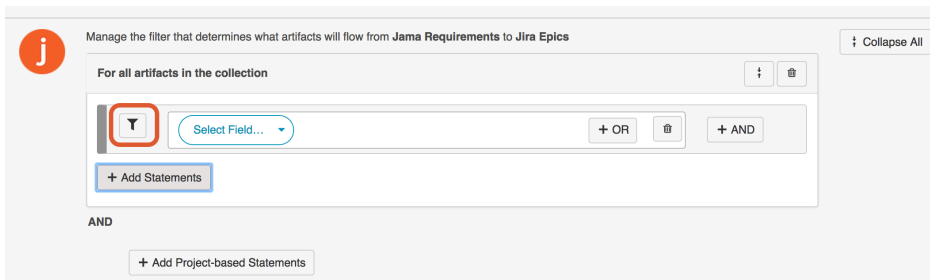
Next, you can begin configuring your artifact filtering statements. You can add statements for all artifacts in both collections, all artifacts in one collection, or to artifacts in specific projects within your collection.

Apply Filter to All Artifacts in Both Collections

To apply a filter to all artifacts in both collections, click '+Add Statements' for all artifacts in your first collection.



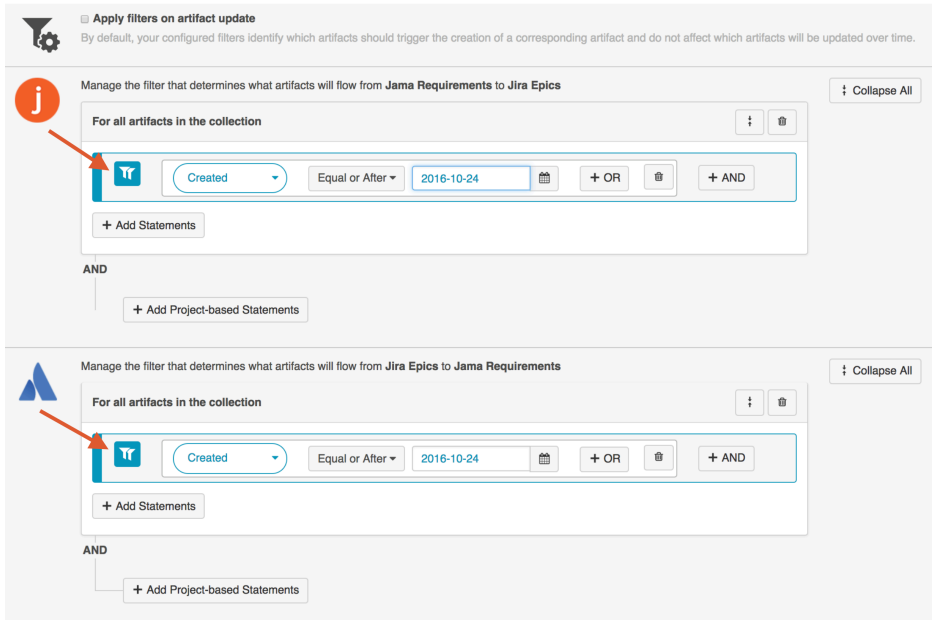
Then, click the filter button. This will apply your filter to the other collection participating in your integration.



You will notice that the button changes to show two filters, indicating that your filter will apply to both collections.

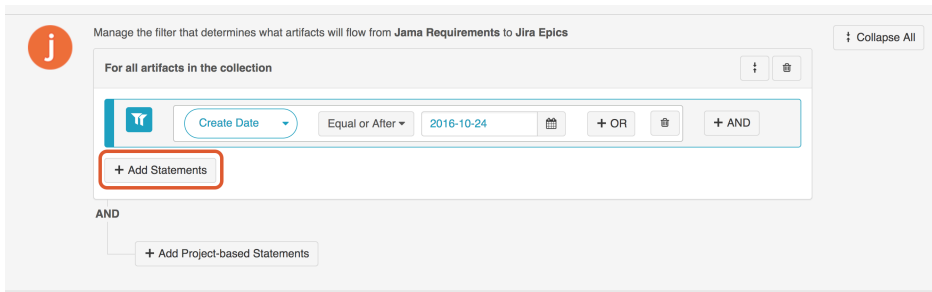
You'll also notice that any modifications you make to that filter statement will automatically be reflected in the other collection. If you'd like to disconnect the filter from both collections, simply click the double-filter button again, and you will be able to edit each filter individually.

Here we are filtering both collections to only create target artifacts that were created on or after October 24th, 2016.

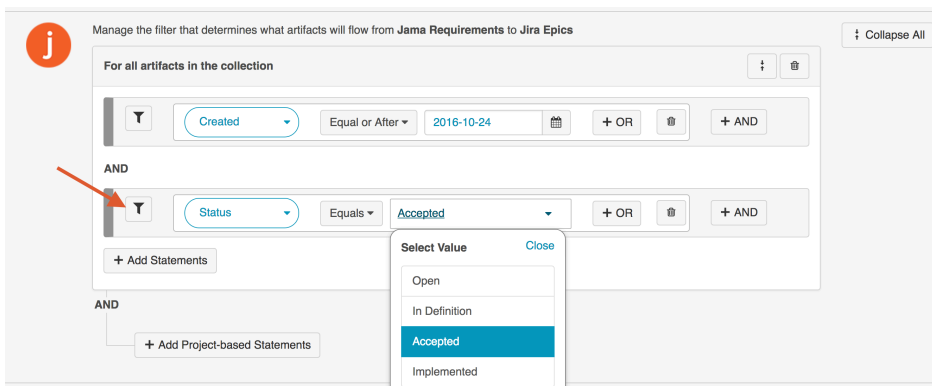


Apply Filter to All Artifacts in One Collection

To apply a filter to all artifacts in one collection, simply click the '+Add Statements' button in the desired collection:

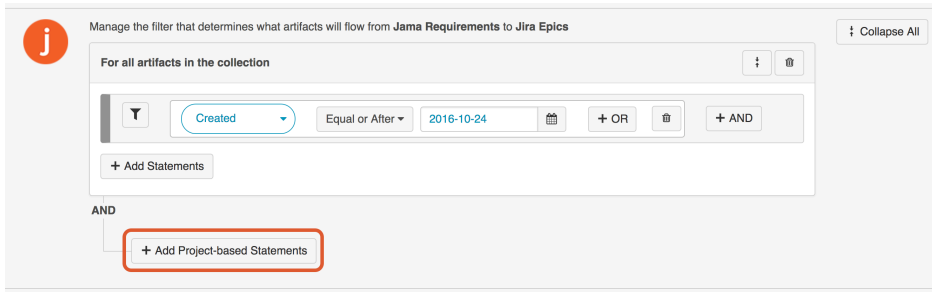


Select your artifact filtering fields and values. You'll see that there is only one filter displayed on the left, which tells you that this filter only applies to one collection in your integration.

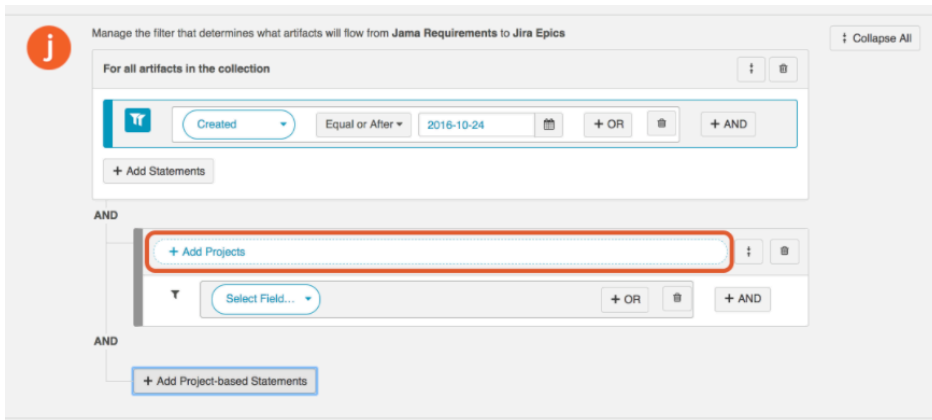


Apply Filter to Artifacts within Certain Projects in a Collection

To apply a filter to artifacts within certain projects in a collection, click '+Add Project Based Statements'

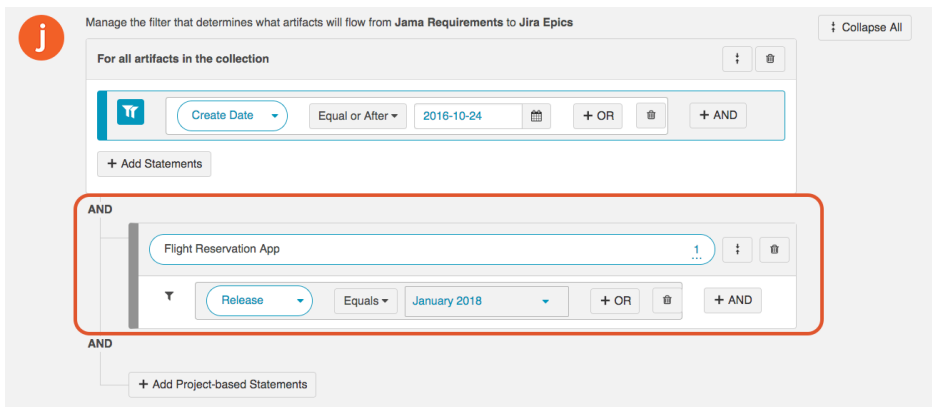


Click '+Add Projects' to select your project.



Select the project(s) you'd like your filter to apply to.

Then click 'Select Field...' to begin configuring your filtering statement.



Viewing Artifact Filter Statements

You can click the 'Collapse All' button to view an easier-to-read version of your artifact filtering statements.

Filtering via Repository Queries

In rare cases, you may find that the best option to restrict the artifacts eligible to flow is by setting a query within the repository itself.

⚠️ Repository Queries are advanced functionality, and should only be used when you are truly unable to filter as desired using the built-in Tasktop functionality of Repositories, Collections, and Artifact Filtering.

If you plan to utilize repository queries, check the box next to 'Enable collections to be refined by setting a repository query,' on the [Repository Connection](#) screen.

Repository Connection
View and configure your repository connection.

[Back to Integration Configuration](#) **Done**

Jama

Label

Location

Authentication

Username

Password

Proxy Server **Use proxy server**

Proxy Host Address

Username

Password

Additional Settings

Repository Query **Enable collections to be refined by setting a repository query**

Concurrency Limit

Once this is selected, you will be able to select a repository query at the [Collection level](#) for any collections utilizing this repository.

Collection Configuration
Configure your collection. Collections comprise a group of artifacts that can be used in one or more integrations.

[Back to Collections](#) **Done**

Jama
Jama
<http://pdt-jama188.van.tasktop.com:8080/contour>

1 of 29 projects
Agile Project

Artifact: Requirement **Model: Requirement**

13 of 51 fields mapped **13 of 15 fields mapped**

Inbound Artifacts **Relating Model**

2 of 27 relationships mapped **2 of 5 relationships mapped**

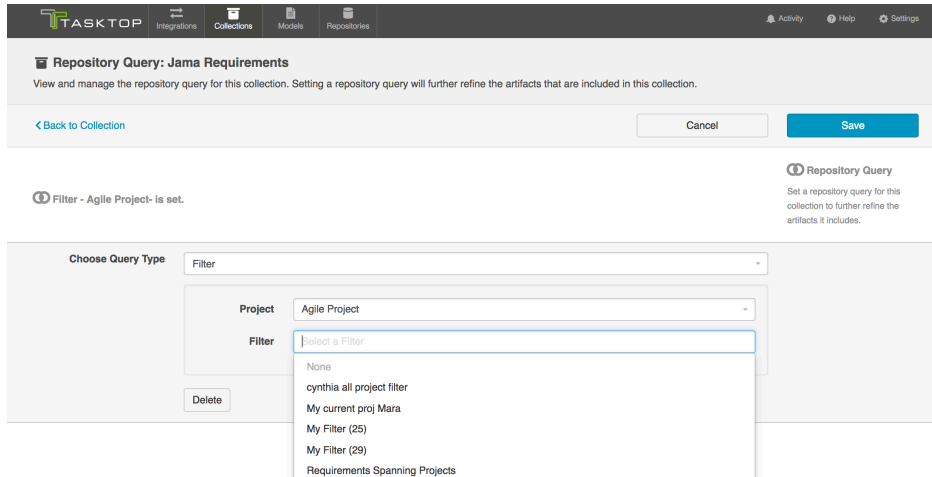
Inbound Person **Person Model**

Person Resolution Strategy

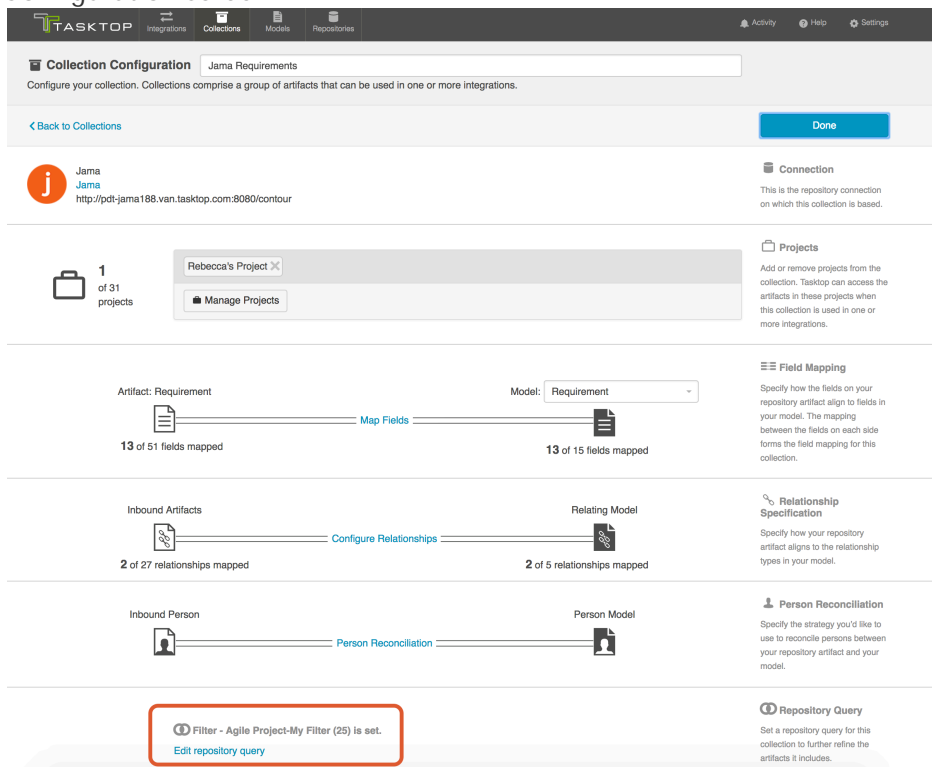
No repository queries are set.
[Set repository query](#)

On the drill-in page, you'll be able to search for your desired repository

query. Select the query you'd like to use, and click 'Save,' and then 'Done.'



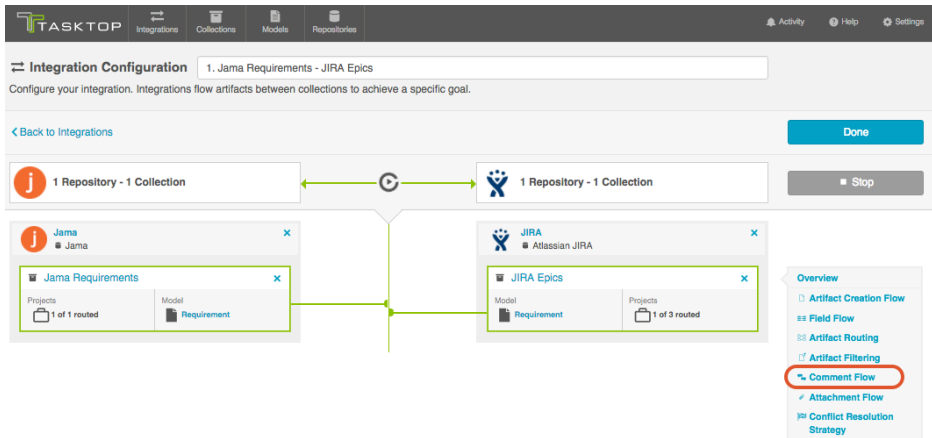
You will then see the selected repository query on the Collection Configuration screen:



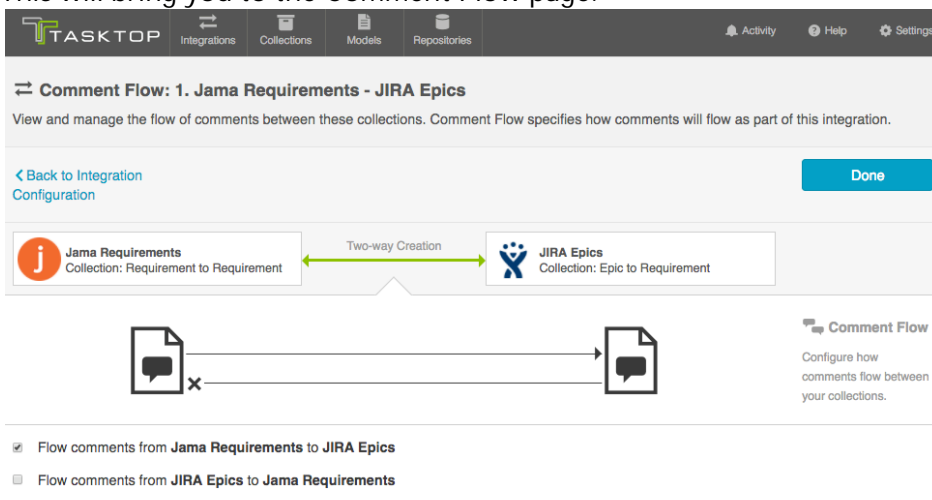
Remember, applying a repository query to a collection will only further refine the artifacts included in that collection. If you select a query that encompasses artifacts in projects not in your collection, these artifacts will not be added to the collection unless you also add those projects to your collection as you normally would.

Comment Flow

When configuring a synchronize integration, you have the option of deciding to flow comments between collections. To enable and configure Comment Flow, click 'Comment Flow' on the Integration Configuration screen.



This will bring you to the Comment Flow page:



If your collection enables comment flow, you will be able to use the check-boxes to flow, or not flow, comments as part of your integration. You can choose to flow comments bi-directionally or in a single direction.

Comment Impersonation


Comment Impersonation refers to Tasktop's ability to assign a specific user to a given artifact or artifact entity. You can learn if your repository supports impersonation by viewing our [Connector Documentation](#) here.

Depending on whether or not impersonation is supported, your comments may flow over to your target repository in one of two ways:

- When a given repository supports impersonation, Tasktop will assign the comment to the proper user if it is possible to locate the user with the information provided on the source artifact.

In cases like this, your comment will appear as though it were created by the corresponding user, as seen in the comment below:

Comments



 **Jane Doe** (15 minutes ago)
The feature has been implemented. Please notify the customer and let us know if there are any questions!

On the other hand,

- When a given repository supports impersonation, but Tasktop cannot locate the person with the information provided from the artifact in the other repository,
Or,
- When a given repository does not support impersonation,

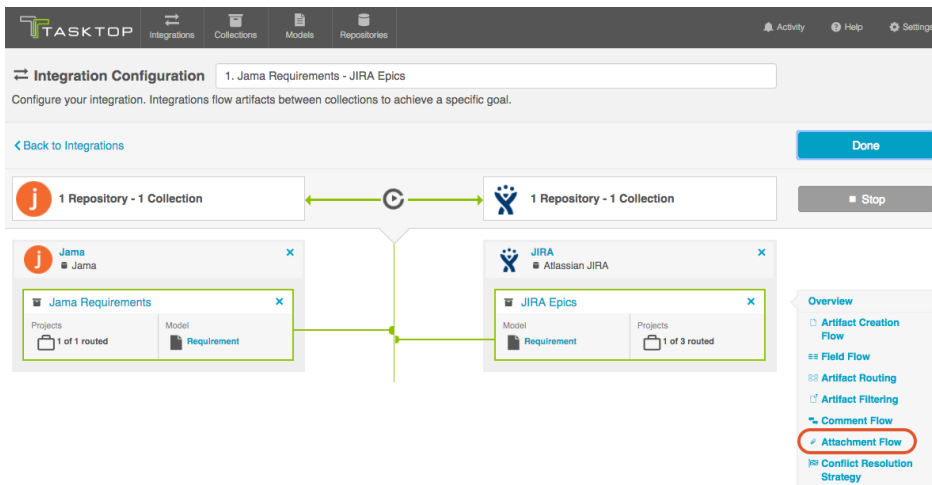
The comment will appear in your target repository as though it were created by the default user associated with your repository configuration in Tasktop, and the name of the user who truly recorded the comment will be listed at the beginning of the comment text.

In cases like the final two outlined above, your comment will look like this:

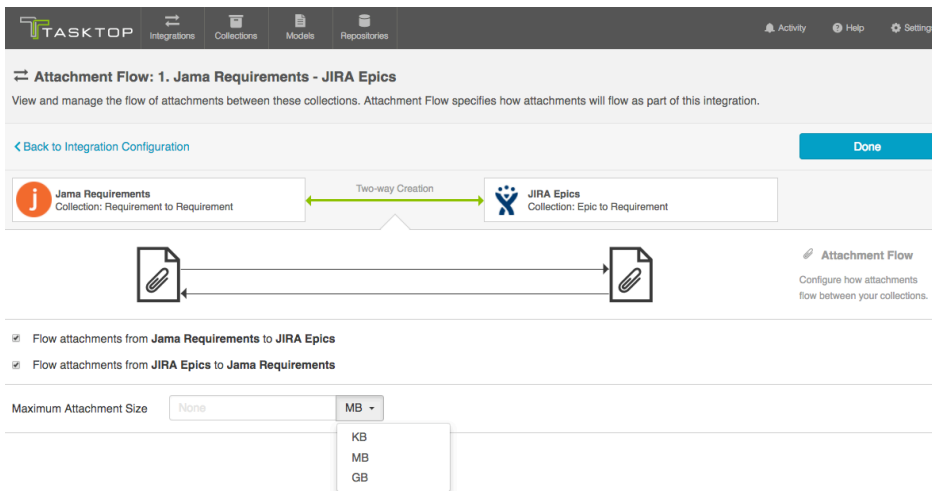
 **Tasktop Marketing** 
(Comment from Jane Doe):
The feature has been implemented. Please notify the customer and let us know if there are any questions!
[Comment](#) · [Like](#) · Today at 1:47 PM via Field API Access

Attachment Flow

When configuring a synchronize integration, you have the option of deciding to flow attachments between collections. To enable and configure Attachment Flow, click 'Attachment Flow' on the Integration Configuration screen.



This will bring you to the Attachment Flow screen:



If your collection enables attachment flow, you will be able to use the check-boxes to flow, or not flow, attachments as part of your integration. You can also configure the maximum attachment size. If attachments are larger than this size, they will be ignored by your integration.

💡 If you are unsure of the maximum attachment size allowed in your repository or if you leave this field blank and it turns out that the attachment is, in fact, larger than the maximum size the repository allows, you will see an error message in Tasktop for that attachment. You can then deduce, based on the error message in Tasktop, what the maximum size is, and use that data to populate the field on the Attachment Flow screen.

When a given repository supports impersonation (the ability for Tasktop to assign a specific user to a given artifact or artifact entity), Tasktop will assign the attachment to the proper user given we can locate the user with the information provided from the artifact in the other repository. If a given repository supports impersonation but Tasktop cannot locate the

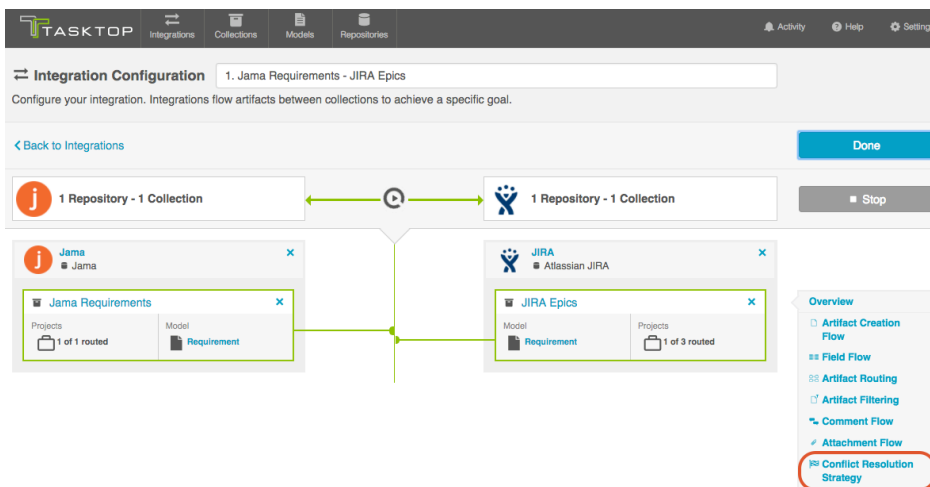
person with the information provided from the artifact in the other repository, Tasktop will write out the attachment attributed to the default Tasktop user. When a given repository does not support impersonation, Tasktop will similarly write out the attachment attributed to the default Tasktop user.

Conflict Resolution Strategy

Another unique byproduct of the Synchronize Integration is the Conflict Resolution Strategy, which allows users to control how a data conflict will be resolved. A data conflict occurs when mapped fields of an end repository get updated with different values in each repository before the synchronization scan runs, if bidirectional field flow is enabled for the Synchronize integration.

To illustrate this, let's imagine a scenario where you have been running a synchronize integration for defects between HPE ALM and JIRA with bidirectional field flow (the default setting) for some time. If HPE ALM's Defect #123 is modified to have one value (say, status = done), and its target artifact in JIRA is modified to have a conflicting value (say, status = in progress) before the next change detection interval, this data conflict will need to be resolved in some way before the two artifacts can be synchronized.

To select your Conflict Resolution Strategy, click the 'Conflict Resolution Strategy' link on the right side of the Integration page.

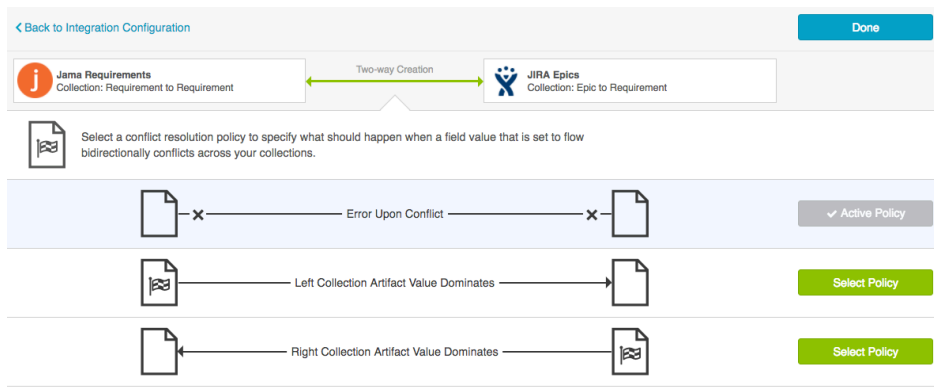


You will have three options for your Conflict Resolution Strategy:

1) Error upon Conflict: An error will be generated, and no updates will be made for the conflicted field, or any other fields on the artifact. The error message will notify you that the conflict occurred and will provide steps on how to resolve the conflict. Note that once a conflict is detected, no subsequent updates will be made to the artifact pair until the conflict is resolved.

2) Left Collection Artifact Value Dominates: Values from the artifact in the left collection will over-write the values in the right collection.

3) Right Collection Artifact Value Dominates: Values from the artifact in the right collection will over-write the values in the left collection.

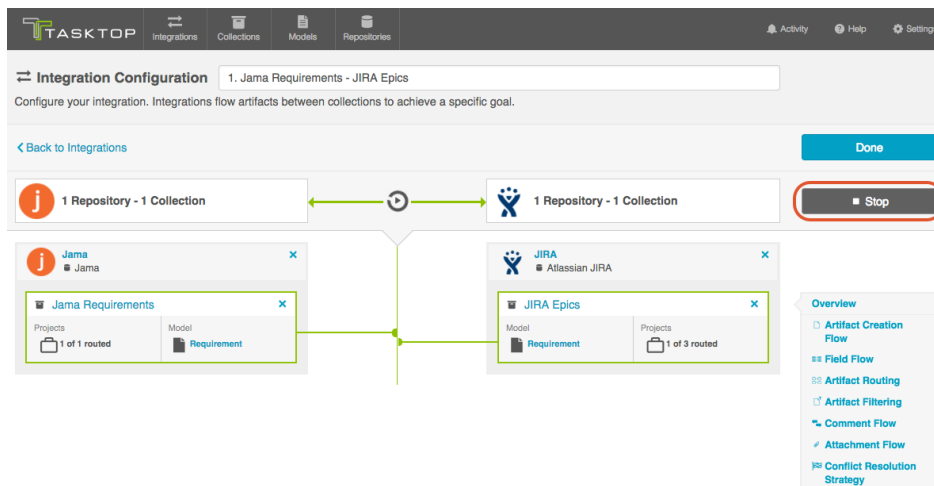
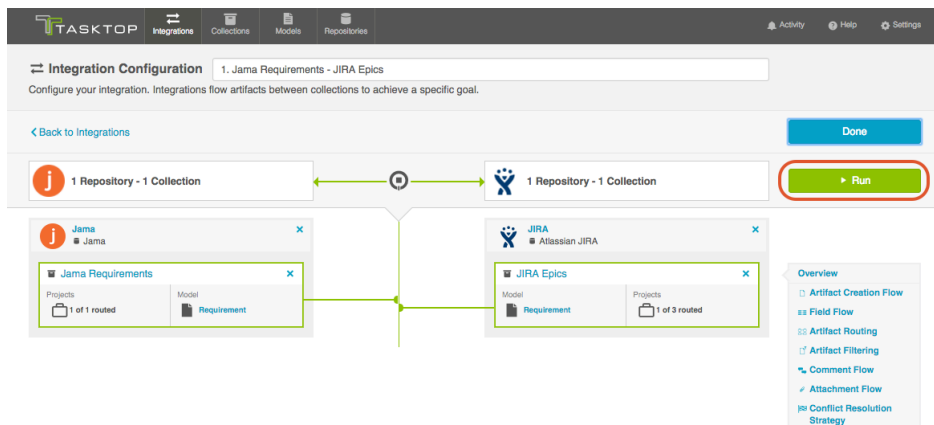


Running your Integration

There are two ways to start or stop your integration:

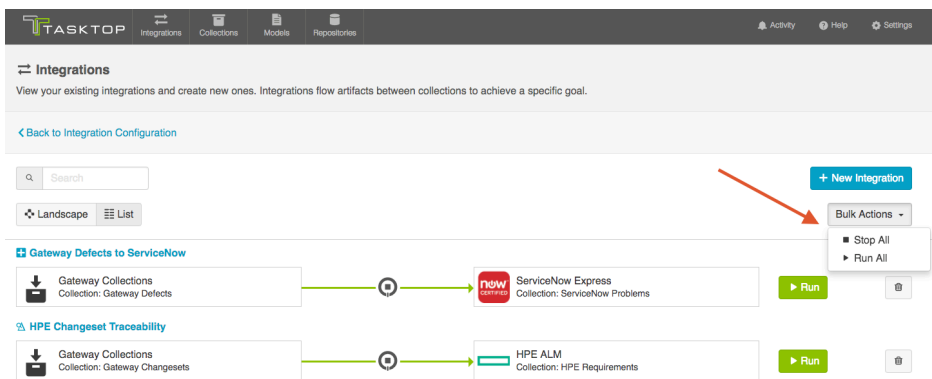
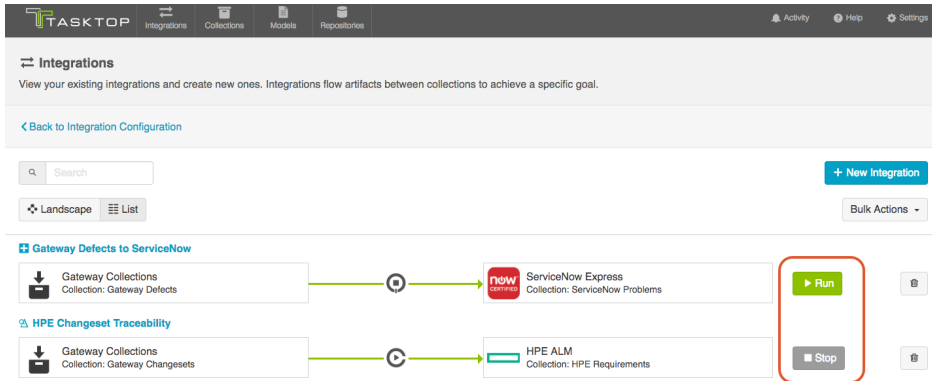
From the Integration Configuration Screen

Simply click 'Run' to run the integration, and 'Stop' to stop the integration.



From the Integrations List Page

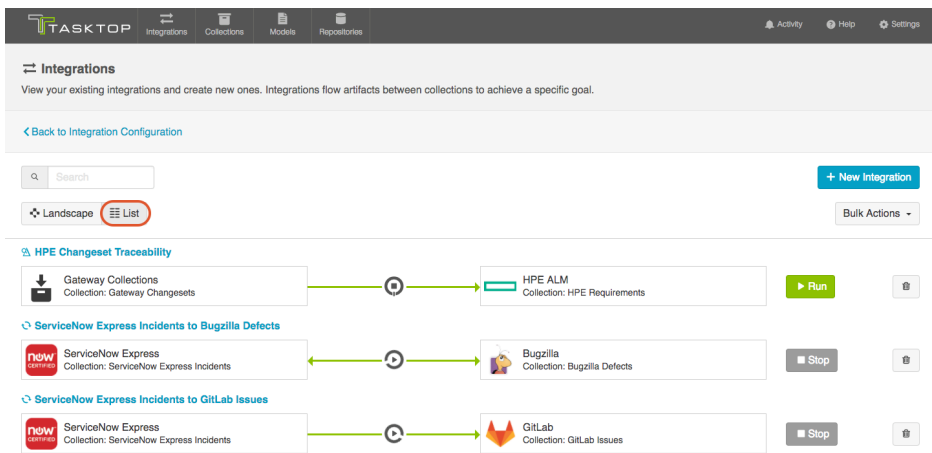
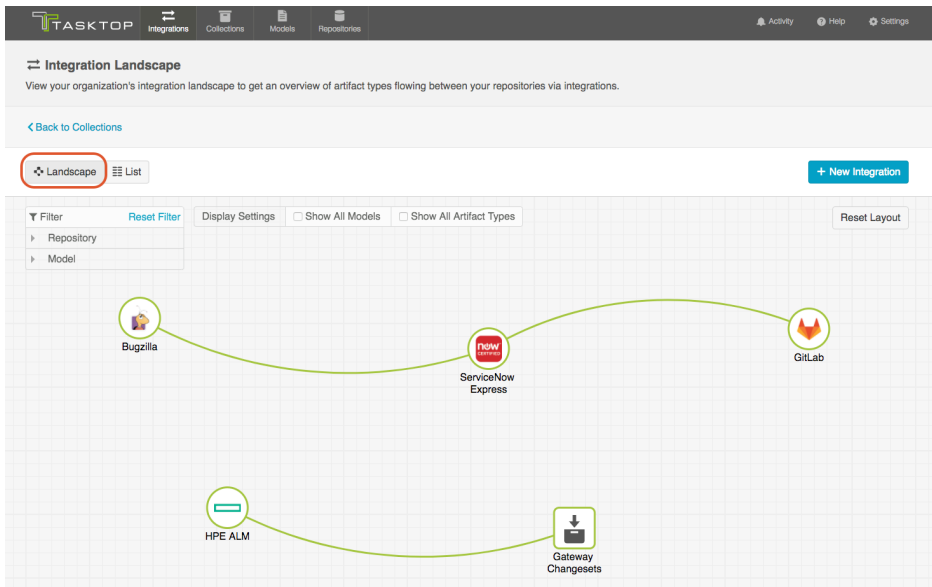
Click 'Run' or 'Stop' next to each integration you would like to update.
You can also use the 'Bulk Actions' button to run or stop all integrations.



Viewing Your Integrations

See [Tasktop Editions table](#) to determine if your edition contains Integration Landscape View functionality.

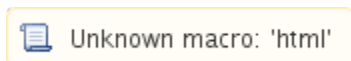
When viewing your integrations, you have the option of viewing them in either Landscape or List mode.



Landscape View

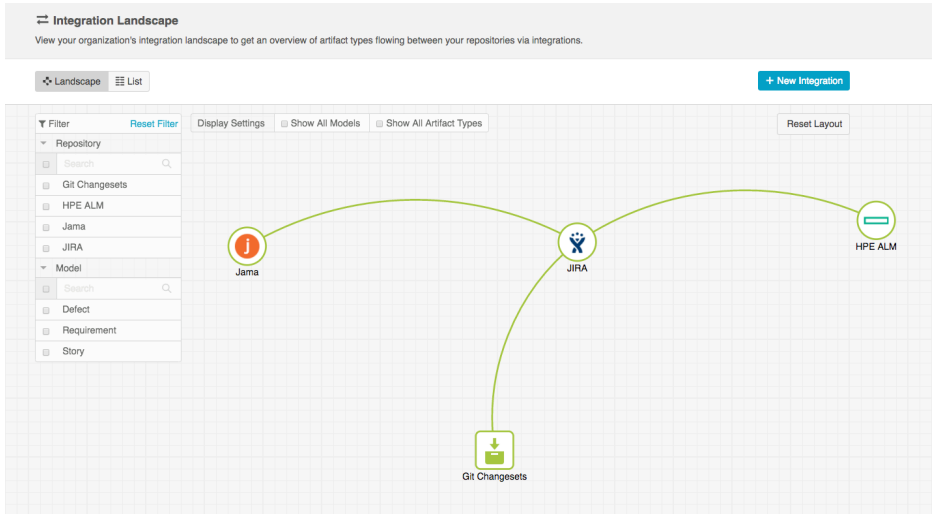
See [Tasktop Editions table](#) to determine if your edition contains Integration Landscape View functionality.

Learn more about the Integration Landscape View in the video below:



Tasktop will default to the Landscape View, which enables you to visualize your entire integration landscape and see how your integrations relate to one another. Use our built-in filters to see as little or as much information as you'd like!

Here's a simplified view:

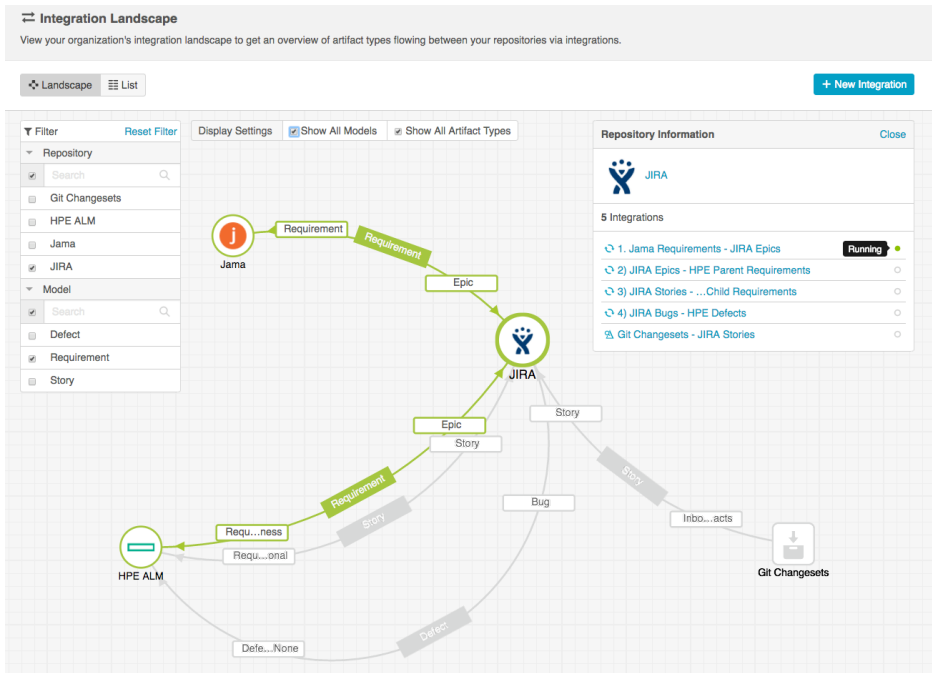


If you'd like to see additional information, you can utilize the filters, or click on a repository node to modify which information is shown.

Some examples of additional information you can see are:

- Models
- Artifact Types
- Artifact Creation Directionality Arrows
- List of all relevant integrations (see this by clicking on the repository node)
 - Indicator of whether each integration is running or not

Here's an example of a more detailed view:



List View

If you'd like, you can toggle to List View, which will show you a list of all

integrations you have created.

You can use this view to:

- Start an Integration
- Stop an Integration
- Delete an Integration
- Click into an Integration and modify its configuration

The screenshot shows the Tasktop Integrations page. At the top, there are navigation tabs for Integrations, Collections, Models, and Repositories. Below the tabs, there's a search bar and a '+ New Integration' button. A 'List' button is highlighted with a red circle. The main content area displays a list of integrations, each represented by a box with a collection icon and name, connected by a double-headed arrow. The integrations are:

- 1. Jama Requirements - JIRA Epics: Jama Collection: Jama Requirements ↔ JIRA Collection: JIRA Epics (Stop button)
- 2. JIRA Epics - HPE Parent Requirements: JIRA Collection: JIRA Epics ↔ HPE ALM Collection: HPE Parent Requirements (Run button)
- 3. JIRA Stories - HPE ALM Child Requirements: JIRA Collection: JIRA Stories ↔ HPE ALM Collection: HPE Child Requirements (Stop button)
- 4. JIRA Bugs - HPE Defects: JIRA Collection: JIRA Bugs ↔ HPE ALM Collection: HPE Defects (Stop button)
- Git Changesets - JIRA Stories: Gateway Collections Collection: Git Changesets ↔ JIRA Collection: JIRA Stories (Stop button)

Tips and Tricks

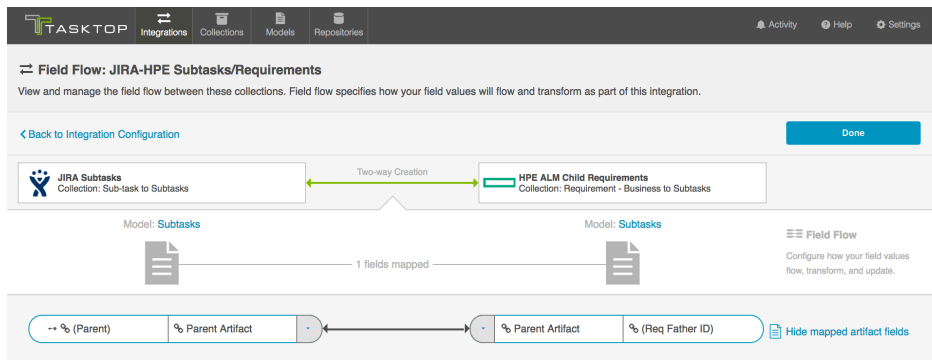
Synchronizing Internal Relationships

Tasktop affords you the ability to not only flow various artifacts between your collections, but also to mirror the relationships between those artifacts.

Below, we'll outline how to configure parent-child relationships in a synchronize integration, though the same process can be followed for a multitude of relationship types:

1. In Collection A, we have parent-tasks linked to child-subtasks. In Collection B, we have parent-requirements linked to child-sub-requirements. To flow these artifacts along with their relationships, we will need to configure two integrations:
 - a. Task-Requirement Synchronize Integration
 - b. Subtask - Sub-Requirement Synchronize Integration, with 'parent' relationship field
2. First, configure your Story-Story Synchronize Integration as you typically would
3. Next, configure your Subtask - Task Synchronize Integration
 - a. Ensure that your model includes the 'Parent' Smart Field.
 - b. On your Collection pages, click 'configure relationship types,' and map the 'parent' field appropriately
 - c. On your Integration Field Flow page, you will see the two relationship types mapped to one another.
4. Run both integrations. You will see your parent artifacts, your child

artifacts, as well as *their relationships to one another* successfully flow as part of your integration.



Synchronizing an Artifact ID or URL Reference

Imagine this scenario: You are flowing defects between two repositories: JIRA and HPE ALM. You'd like to have a way to know the ID, or URL, of the source artifact in JIRA when viewing its target artifact in HPE ALM (and vice versa). This will provide traceability between the source artifacts and the artifacts that have been created in your target repositories, via your integration.

To set this up, you will need to configure two different field mappings in each collection:

- You will need to specify which field to pull the source artifact's ID (or URL) from
- You will need to specify which field to use to store the source artifact's ID (or URL), in your target repository



In the diagram above, you can see that HPE ALM is flowing its ID field to a custom field in JIRA, and that JIRA is flowing its ID field to a custom field in HPE ALM. In order to set up this integration, you will need to configure your model to accept that ID field. We'll walk through how to do that below.

The instructions below will walk you through how to set up this configuration for the ID field, but the same instructions will also apply for location/URL:

1. Go to the Model that you are utilizing in the integration. Ensure that your model includes the Formatted ID field.

We've also shown the 'Location' field below, for reference, as a similar process can be followed to flow the source artifact's URL to a field on the target artifact, for traceability.

Standard Field	Label	Type	Required
Formatted ID	Formatted ID	String	<input checked="" type="checkbox"/>
Location	Location	Location	<input checked="" type="checkbox"/>

2. Go to the Collections page for each of your repositories, and set up mapping to tell the integration where to pull the ID from:

- a. Map the Formatted ID model field to the corresponding field in your end repository. This is the field that the collection will take the ID data from. Note that Formatted ID is called 'Key' in JIRA, but may be referred to using a different name in a different repository (i.e. 'issue ID')

- b. Click 'Configure' next to your mapping, and confirm that your Transforms are configured as shown below. The transform on the left should be 'None' (will display as 'Select Transform') and the transform on the right should be 'Copy.' This will tell the collection to *send* data from the Key field in your repository to the model, but not vice versa.

- c. Repeat these steps in your other repository.
- d. Here is how the mappings should look in each repository, for your *source* fields:

Field Mapping: JIRA Defects
View and manage the field mapping for this collection. The field mapping specifies how fields from your repository artifact map to fields in your model.

[← Back to Field Mapping](#) Done

Key ————— Formatted ID

String String

Transform Select Transform Transform Copy

Field Mapping: HPE ALM Defects
View and manage the field mapping for this collection. The field mapping specifies how fields from your repository artifact map to fields in your model.

[← Back to Field Mapping](#) Done

Formatted ID ————— Formatted ID

String String


Transform Select Transform Transform Copy

3. Now that our model is able to acquire ID data from each source repository, let's tell it where to store that data in the corresponding target repository. To do this, you will set up an additional mapping in each Collection:
 - a. Navigate to one of your Collections.
 - b. Map the Formatted ID model field to your repository once more, this time to determine where you would like to store this data in your target repository. The field mapping page will tell you that this is a 'duplicate,' but that is OK!

Field Mapping: Jira Defects
View and manage the field mapping for this collection. The field mapping specifies how fields from your repository artifact map to fields in your model.


[← Back to Collection](#) Cancel Save

JIRA: Bug



6 of 49 fields mapped

Model: Defect

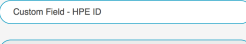


5 of 17 fields mapped

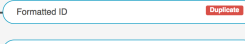
Field Mapping
Specify how the fields on your repository artifact align to fields in your model. The mapping between the fields on each side forms the field mapping for the collection.


Search "Jira Defects" artifact fields by name or type Select a field on each side to configure additional field mappings Search "Defect" model fields by name or type Suggest Mappings C

Custom Field - HPE ID

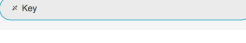


Formatted ID Duplicate





Configure 

Key




Formatted ID Duplicate



Configure 


In the image above, we have mapped 'formatted ID' to a custom field in JIRA called 'Custom Field - HPE ID'. This is the field that the HPE ALM Formatted ID data will flow to in JIRA.

 **Note:** Do not click 'Save' yet. If you do, you will get an error. Continue to the next step below.

- c. Click 'Configure' on the new mapping, and configure as shown below. This will tell the collection to take data from

the model and send it to the 'Description' field, but not vice versa.

The screenshot shows a field mapping configuration for 'Custom Field - HPE ID' to 'Formatted ID'. On the left, the 'Transform' dropdown is set to 'Copy'. On the right, the 'Transform' dropdown is open, showing options: 'Select Transform', 'Copy', 'Append (String)', and 'None'. The 'None' option is highlighted in blue.

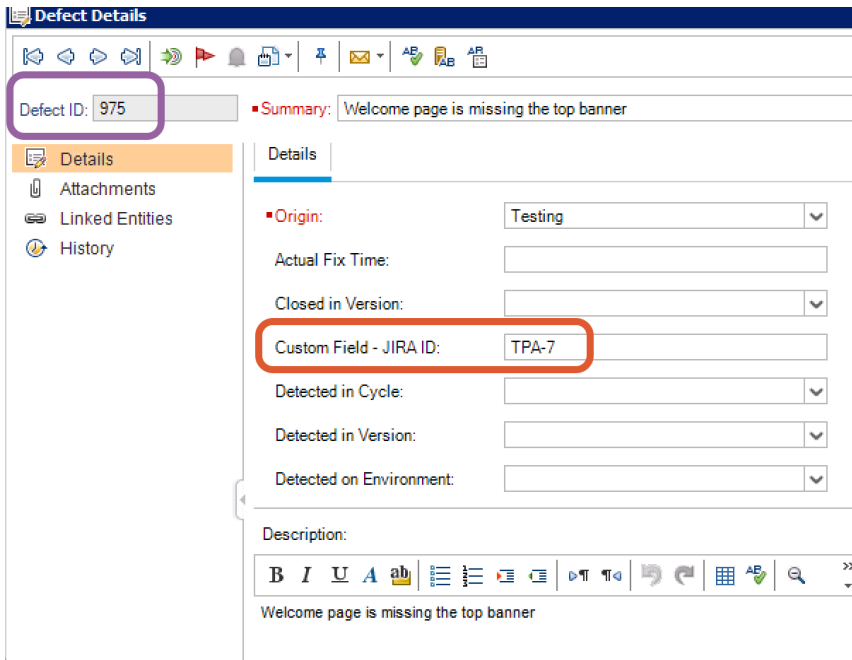
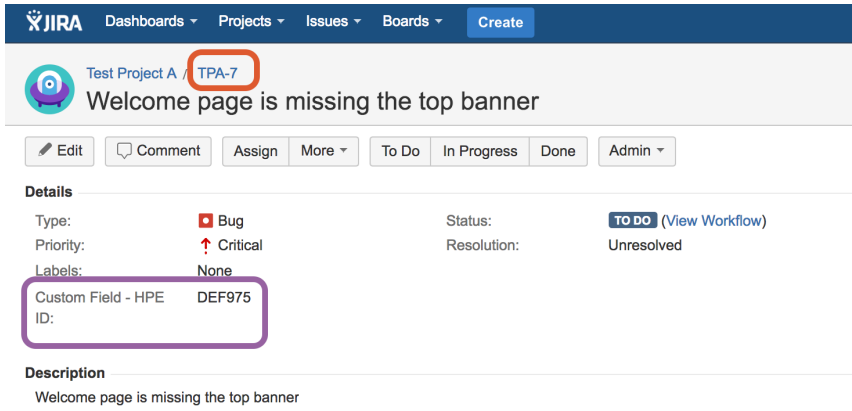
 Note: The transform on the left may be 'Copy,' 'Formatted String to Rich Text,' or some other transform depending on the field types of the repository field and model field. However, the important thing is that the transform on the right (on the model side) be set to 'None.' This ensures that data will only flow *into* the repository field, rather than *out* of it.

- d. Save your mapping and collection.
- e. Repeat these steps on your other collection.
- f. Here is how your transforms should look in each collection, for your *target* fields:

The screenshot shows the 'Field Mapping: Jira Defects' configuration page. It includes a title bar, a description, and navigation buttons: '< Back to Field Mapping', 'Cancel', and 'Save'. Below the title bar, there is a field mapping configuration for 'Custom Field - HPE ID' to 'Formatted ID'. The 'Transform' dropdown for the source field is set to 'Copy', and the 'Transform' dropdown for the target field is set to 'Select Transform'.

The screenshot shows the 'Field Mapping: HPE ALM Defects' configuration page. It includes a title bar, a description, and navigation buttons: '< Back to Field Mapping', 'Cancel', and 'Save'. Below the title bar, there is a field mapping configuration for 'Custom Field - JIRA ID' to 'Formatted ID'. The 'Transform' dropdown for the source field is set to 'Copy', and the 'Transform' dropdown for the target field is set to 'Select Transform'.

- 4. When you run the integration, the ID of the source artifact will now flow to a field on the target artifact (and vice versa), as specified in your field mapping:



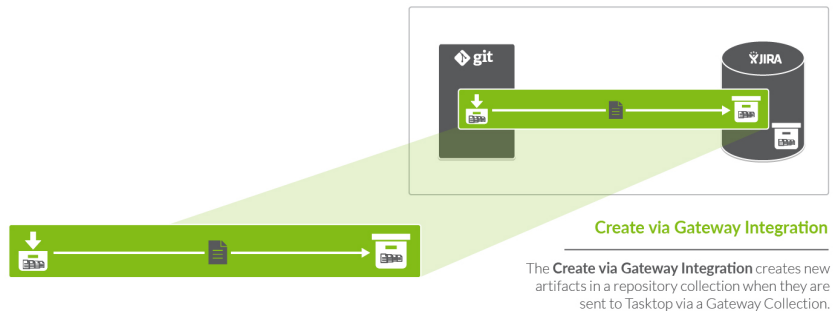
Create via Gateway Template

Tasktop: 17.4 Release

The Create via Gateway Integration Template is only available in Editions that contain the Gateway add-on. See [Tasktop Editions table](#) to determine if your edition contains this functionality.

What is a Create via Gateway Integration?

- What is a Create via Gateway Integration?
- Video Tutorial
- Use Case and Business Value
- Template Affordances
- How to Configure a Create via Gateway Integration
 - Field Flow
 - Field Flow Icons
 - Artifact Routing



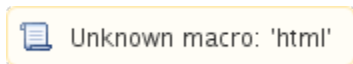
An *integration* is quite simply the flow of information between two or more collections. A *Create via Gateway Integration*, specifically, creates new artifacts in a repository collection, such as JIRA, when they are sent to Tasktop via a Gateway Collection. The Gateway Collection uses an inbound webhook to access information in an external DevOps tool, such as Git or Jenkins.

When you configure a Create via Gateway Integration, you can customize the field flow, artifact routing, and artifact filtering of your integration.

Video Tutorial

Check out the video below to learn how to configure the Create via Gateway Integration Template.

⚠ This video assumes that you have already configured your repositories, models, and collections as outlined in the [Quick Start Guide](#).

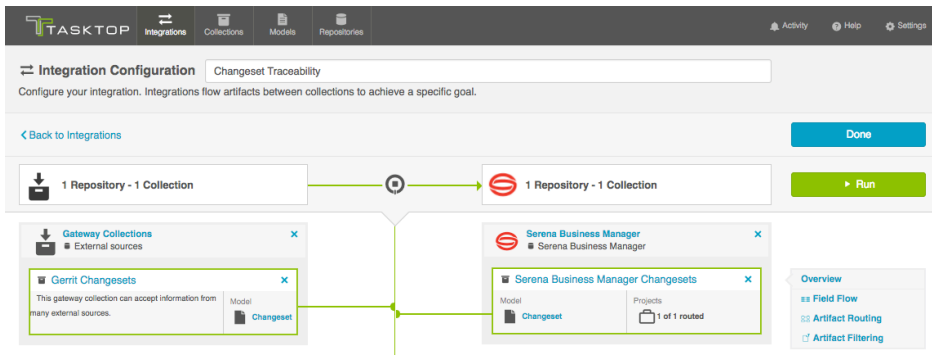


Use Case and Business Value

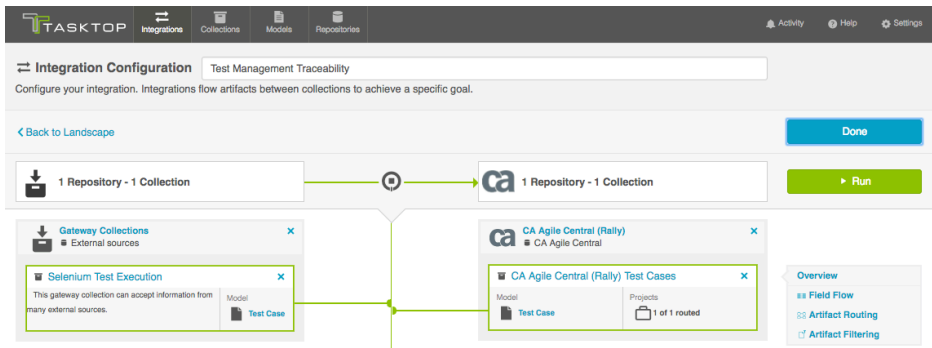
This integration creates traceability between artifacts across the software development lifecycle. New artifacts will be created in a repository collection when artifacts are sent to Tasktop via a Gateway collection. Optionally, these newly-created artifacts can be related to already-existing artifacts in the same repository.

For example, if your development team uses Gerrit for source code management and Serena Business Manager (SBM) for its agile story management, but would like traceability between changesets in Gerrit and stories in SBM, you could set up an integration that would trigger the creation of changesets in SBM when changesets were created in Gerrit. And if the changesets in Gerrit identify the stories in SBM to which they pertain, Tasktop would find the already existing story in SBM and create a relationship between the two artifacts.

- Static Artifact Routing
- Conditional Artifact Routing
- Artifact Filtering
- Running your Integration
 - From the Integration Configuration Screen
 - From the Integrations List Page
- Viewing Your Integrations
 - Landscape View
 - List View
- Tips and Tricks
 - Creating Relationships Between Newly Created Artifacts and Existing Artifacts

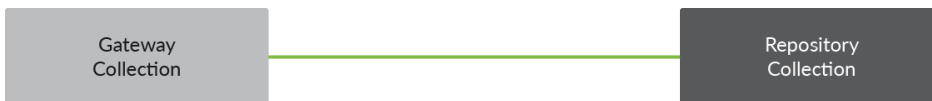


Additionally, if your QA team uses a tool like Selenium for test execution but CA Agile Central (Rally) for test management, you can set up an integration that would trigger the creation test results in CA Agile Central (Rally) when test results are created in Selenium. And if the test results from Selenium identify the tests in CA Agile Central (Rally) which they cover, Tasktop would find the already-existing test and create a relationship between the two artifacts.



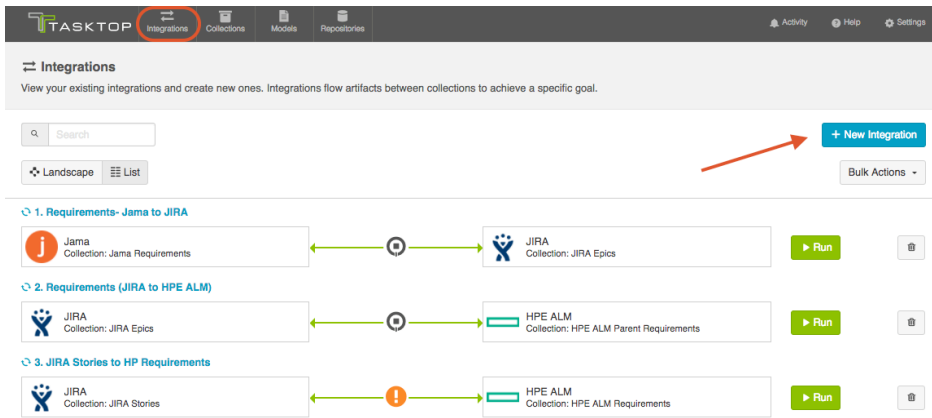
Template Affordances

The Create via Gateway Integration Template allows you to flow artifacts from a single gateway collection into a single repository collection. When a new artifact is sent to Tasktop via our REST API, an artifact will be created in the target repository collection.



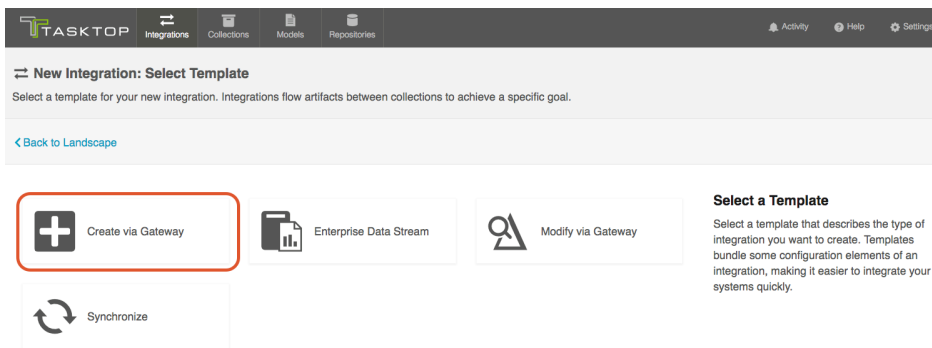
How to Configure a Create via Gateway Integration

To configure your integration, select 'Integrations' at the top of the screen, then click 'New Integration.'

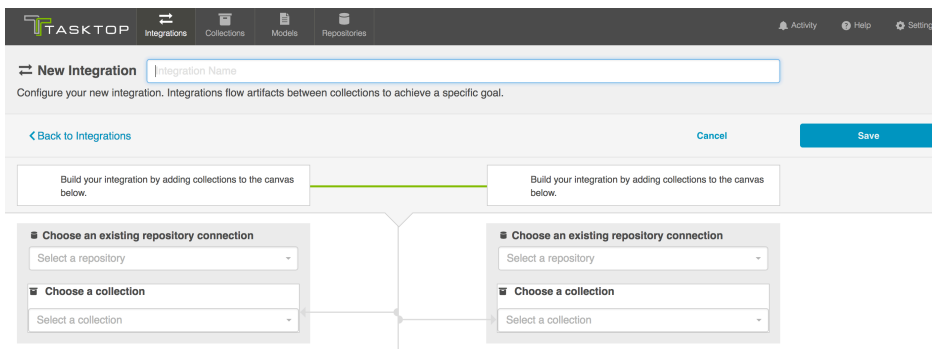


Select the 'Create via Gateway' template.

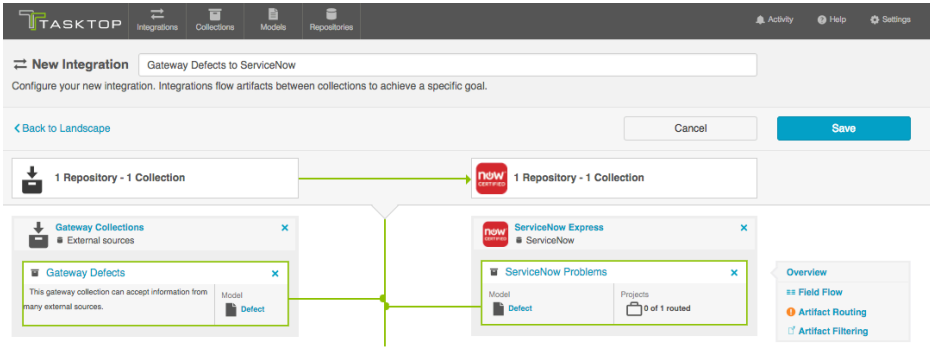
💡 Depending on the edition of Tasktop you are utilizing, you may not have all options available.



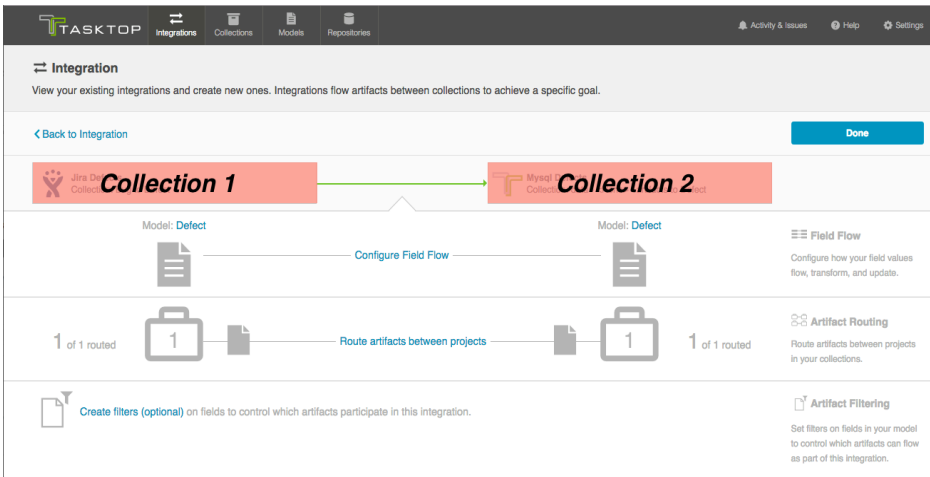
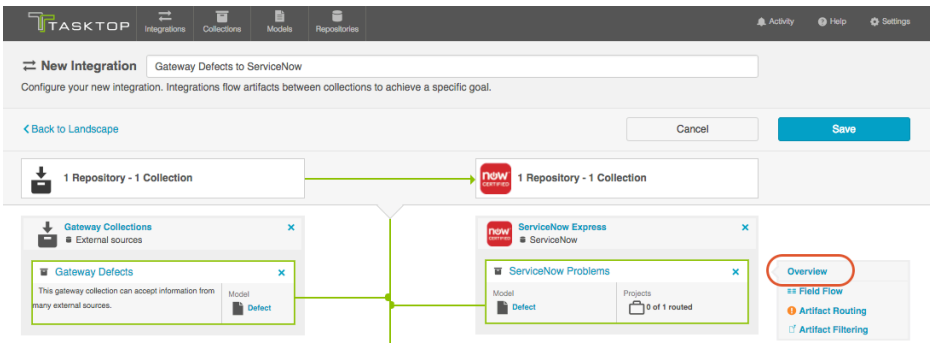
This will bring you to the New Integration Screen:



Name your integration and select your repositories and collections:



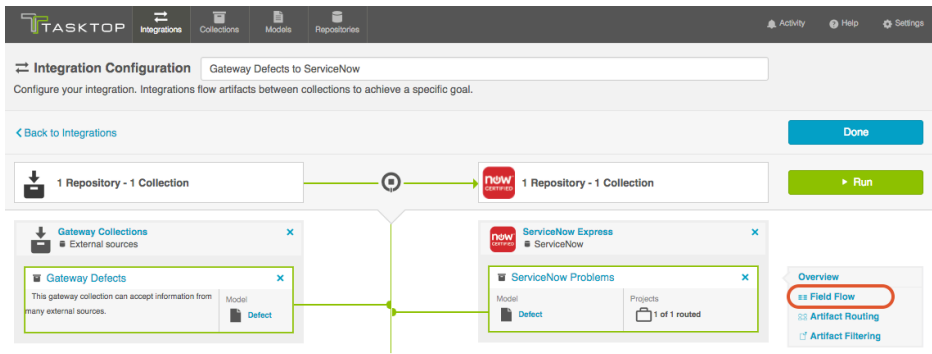
You can click the 'Overview' link on the right side of the Integration Page to get to the main display page (shown in the second screen shot):



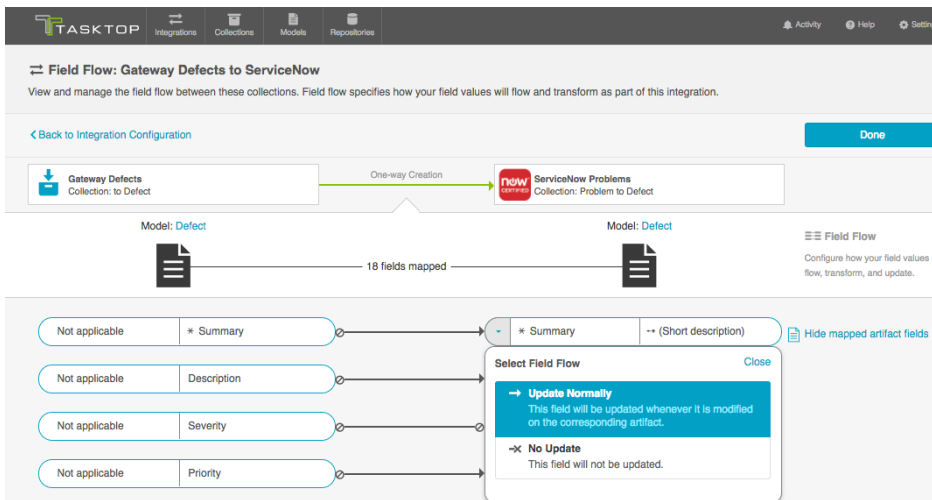
Field Flow

The field flow configured for a given integration specifies which fields should flow in that integration. For Create via Gateway integrations, you can choose to flow a given field (Update Normally) or to not flow a given field (No Update).

To get to the Field Flow screen, click 'Field Flow' on the right pane of the Integration Configuration screen:



You will be directed to the Field Flow Screen:



You can choose to flow a field ('update normally') or not flow it ('no update'). You'll notice that field flow goes in one direction only - from the gateway collection *into* the repository or database collection.







You can see the names of the mapped artifact fields for each collection on the far left and far right, with the model fields displayed in the middle. To hide the mapped artifact fields, select 'Hide mapped artifact fields' on the right.

⚠ Note: The field flow settings behave a bit differently for Constant Values. This is because constant values exist as part of your Tasktop configuration, and not on the artifact itself. Therefore, changes in constant values are not detected in the same way that updates made on the actual artifact are detected. If you change the constant value that is linked to your model, your integration will not automatically detect this update and sync it over. The value will only update if another field on that artifact is updated. Because of this, for constant values, "update normally" and "always update" will behave identically: meaning that the constant value will update whenever any other field is updated on that artifact.

Field Flow Icons

On the Field Flow page, you will see a number of icons, which will help

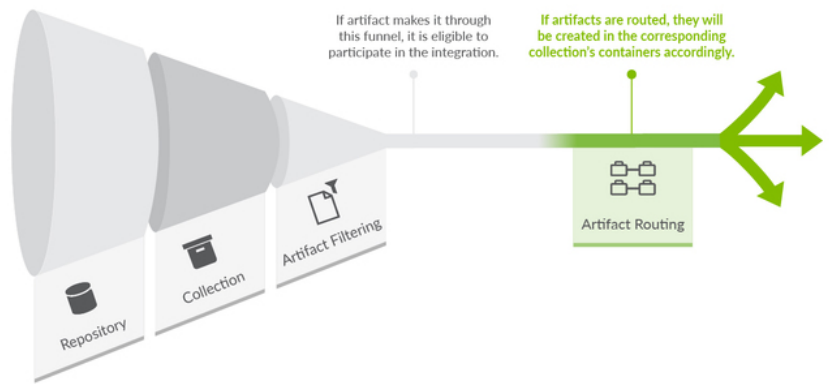
you understand any special properties or requirements for each field. If you hover your mouse over an icon, you will see a pop-up explaining what the icon means. You can also review their meanings in the legend below:

Icon	Meaning
	<p>A constant value will be sent.</p> <p>Note that:</p> <ul style="list-style-type: none"> • If the icon is on the side of the collection, this means that a constant value will be sent to your model. This means that any time this collection is integrated with another collection, the <i>other</i> collection will receive this constant value for the field in question. • If the icon is on the side of the model, this means a constant value will be sent to your collection. This means that any time this collection is integrated with another collection, that <i>this</i> collection will receive this constant value for the field in question.
	<p>Collection field is read-only and cannot receive data</p>
	<p>To create artifacts in your collection, this field must be mapped to your model.</p>
	<p>This is a required field in your model; it must be mapped to your collection.</p>
	<p>This field will not be updated as part of your integration, due to how you have configured it. This field flow configuration can be changed if you'd like.</p>
	<p>This field will not be updated as part of your integration because the mapping would be invalid. You do not have the option of changing this.</p>

→

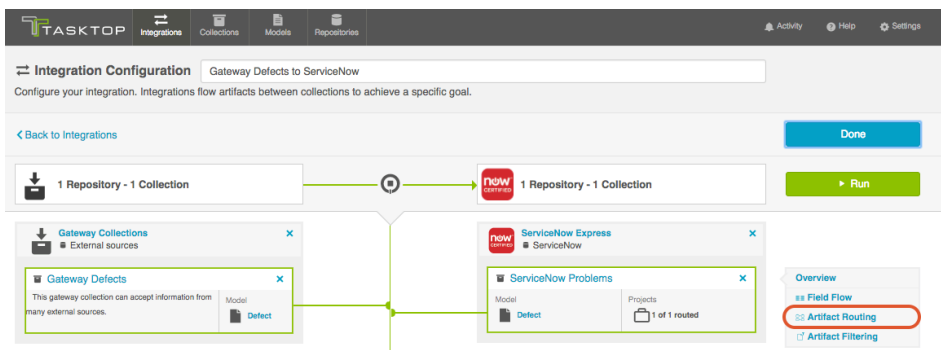
This field will update normally as part of your synchronize integration; this means it will be updated whenever it is modified on the corresponding artifact.

Artifact Routing



Artifact Routing is needed when artifacts are being created as part of an integration. In addition to knowing the repository in which artifacts should be created, Tasktop also needs to know which container (i.e. project, module, folder, etc) a given artifact should be created in. Specifying the artifact routing does this. If your integration does not entail artifact creation, you will not see or need to configure artifact routing.

To configure Artifact Routing, select 'Artifact Routing' on the right pane of the Integration Configuration screen



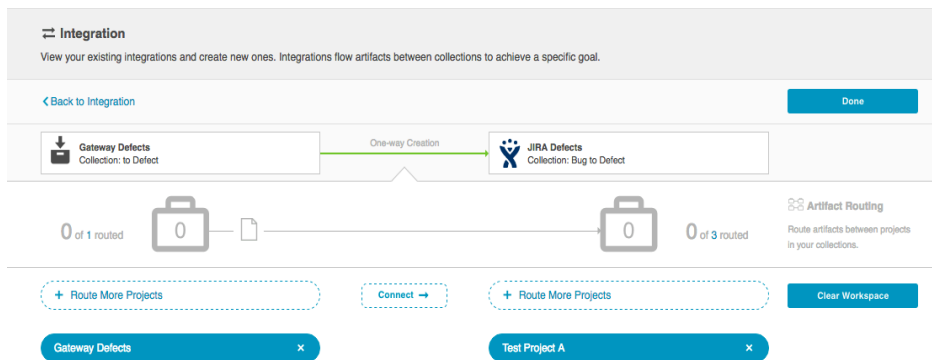
Static Artifact Routing

In some cases, the project an artifact is in in the source collection can sufficiently determine which project an artifact should be created in in the target collection. In these instances, you can configure what is known as 'static artifact routing' (also known as 'explicit artifact routing').

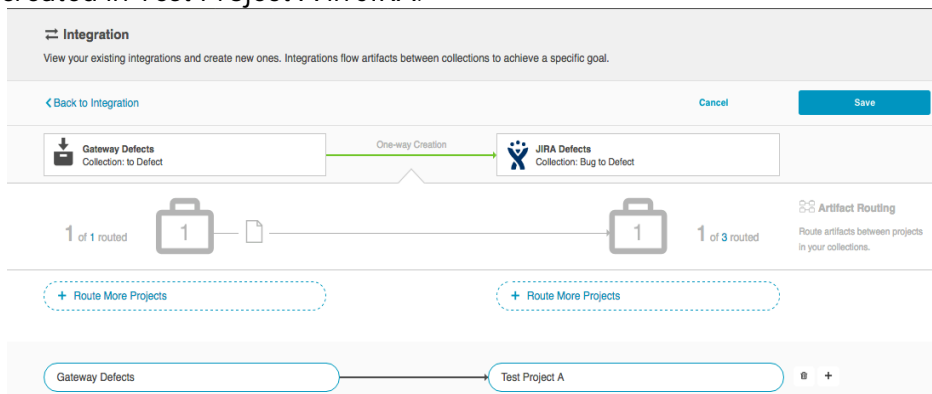
To configure a static artifact route, use the "Route More Projects" buttons to add projects from your collections to your working space and

connect them using the "Connect" button.

Note: Static artifact routes can have one or more source projects, but only a single target project.

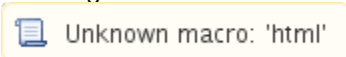


In the example shown below, artifacts from Gateway Defects will be created in Test Project A in JIRA.



Conditional Artifact Routing

Check out the video below to learn more about Conditional Artifact Routing:

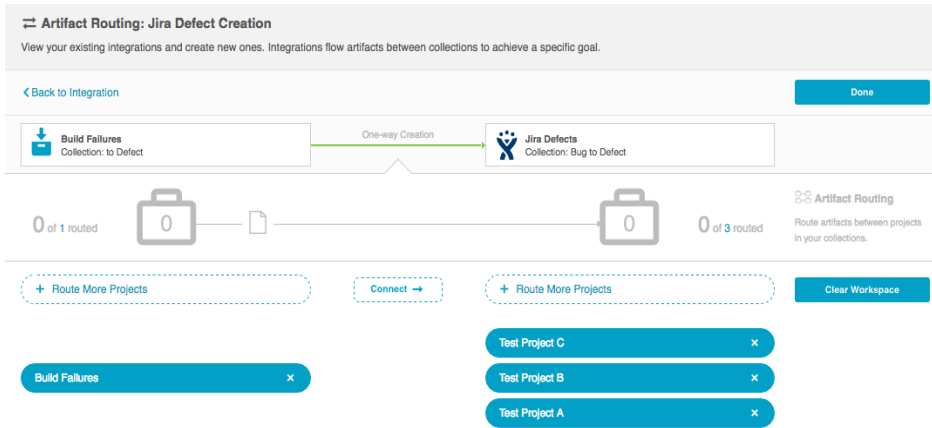


In other cases, the project an artifact is in in the source collection does not provide enough information to determine which project an artifact should be created in in the target collection. Oftentimes, in fact, some unique characteristic of an artifact is the factor that should be used to determine where an artifact should be created in the target collection.

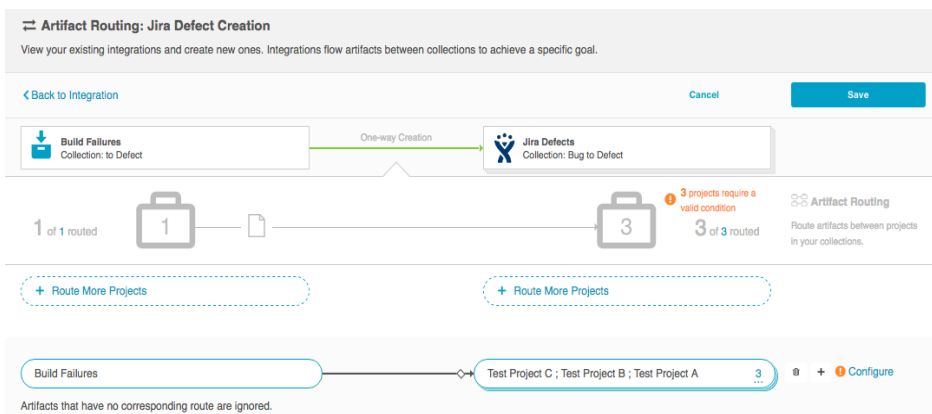
In these instances, artifacts are routed between projects across collections conditionally. Conditional artifact routing (also known as 'dynamic artifact routing') can be used to inspect a single-select field of an artifact and, depending on its value for that field, to route that artifact to be created in the appropriate project in the other collection.

Conditional artifact routes can have one or more source projects, and always have multiple target projects.


To create a conditional artifact route, use the "Route More Projects" buttons to add projects from your collections to your working space and connect them using the "Connect" button.



Notice that after you've created your conditional artifact routing group, you'll be prompted to set the conditions that will define that route.



Click 'Save,' and then click 'Configure.' You'll be brought to the Conditional Artifact Routing screen. Here you'll start by selecting the model field on the artifact that you would like to use to determine your artifact route.

 Note: Conditional Artifact Routes can only be configured based on single-select fields in your model.

In the example below, the field "Application" contains the unique values that should determine the project an artifact will be created in in JIRA.

Conditional Routing: Jira Defect Creation
View your existing integrations and create new ones. Integrations flow artifacts between collections to achieve a specific goal.

[Back to Artifact Routing](#) Done

Build Failures
Collection: to Defect

One-way Creation

Jira Defects
Collection: Bug to Defect

0 conditions configured 3 projects require a valid condition

Build Failures → Test Project C ; Test Project B ; Test Project A 3

Configure which condition the artifacts must match to flow to a project on the right.

Model field that contains the unique routing identifier:

Specify Model Field...

Project
Status
Application

Target project to receive artifacts based on a condition:

Test Project C
Test Project B
Test Project A

Specify handling for artifacts not matched by conditions above:

- Error: Artifacts that do not meet any of the conditions specified will result in an error.
- Ignore: Artifacts that do not meet any of the conditions specified will be ignored.
- Default Route: Artifacts that do not meet any of the conditions specified will be routed to a particular project.

Route unidentified artifacts from this side → Select a Target Project

After you select the model field, you can identify one or more value to correspond to each target project. You can also use the 'Manage Values' link to select from a list of values.

Conditional Routing: Jira Defect Creation
View your existing integrations and create new ones. Integrations flow artifacts between collections to achieve a specific goal.

[Back to Artifact Routing](#) Cancel Save

Build Failures
Collection: to Defect

One-way Creation

Jira Defects
Collection: Bug to Defect

2 conditions configured 1 project requires a valid condition

Build Failures → Test Project C ; Test Project B ; Test Project A 3

Configure which condition the artifacts must match to flow to a project on the right.

Model field that contains the unique routing identifier:

Application

Model field equals one or more unique values:

Mobile Manage Values → Test Project C

Web Manage Values → Test Project B

Specify Unique Values... → Test Project A

Desktop

Once you've done this, you'll see your full conditional artifact routing group:

Conditional Routing: Jira Defect Creation
View your existing integrations and create new ones. Integrations flow artifacts between collections to achieve a specific goal.

[Back to Artifact Routing](#) Cancel Save

Build Failures
Collection: to Defect

One-way Creation

Jira Defects
Collection: Bug to Defect

3 conditions configured All projects can receive artifacts.

Build Failures → Test Project C ; Test Project B ; Test Project A 3

Configure which condition the artifacts must match to flow to a project on the right.

Model field that contains the unique routing identifier:
Application

Model field equals one or more unique values:

Mobile [X] → Test Project C Manage Values

Web [X] → Test Project B Manage Values

Desktop [X] → Test Project A Manage Values

Target project to receive artifacts based on a condition:

In this example:

- defects from Gateway Defects with a value of "Mobile" in the Application field will cause the creation of a corresponding defect in Test Project C in JIRA,
- defects from Gateway Defects with a value of "Web" in the Application field will cause the creation of a corresponding defect in Test Project B in JIRA, and
- defects from Gateway Defects with a value of "Desktop" in the Application field will cause the creation of a corresponding defect in Test Project A in JIRA.

You can specify how you'd like to handle defects from Gateway Defects that do not meet any of the conditions specified (for instance, its value for "Application" is "Other",) by selecting one of the options provided at the bottom of the screen:

Specify handling for artifacts not matched by conditions above:

Error: Artifacts that do not meet any of the conditions specified will result in an error.

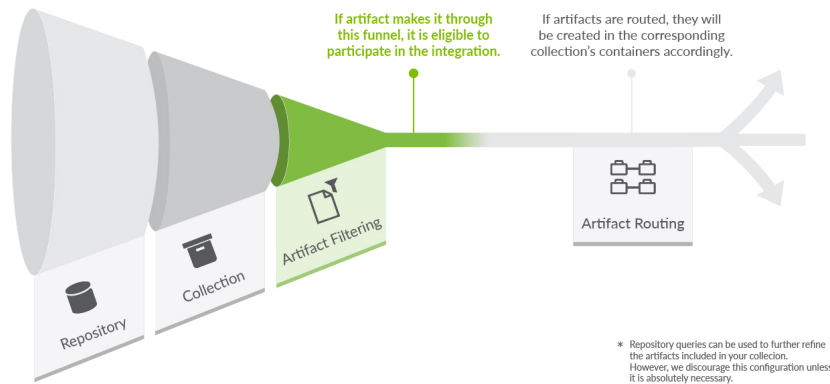
Ignore: Artifacts that do not meet any of the conditions specified will be ignored.

Default Route: Artifacts that do not meet any of the conditions specified will be routed to a particular project.

Route unidentified artifacts from this side → Select a Target Project

Artifact Filtering

When configuring your integration, you have several options available to refine which artifacts are eligible to flow. The final mechanism available is *artifact filtering*, which is configured at the Integration level.

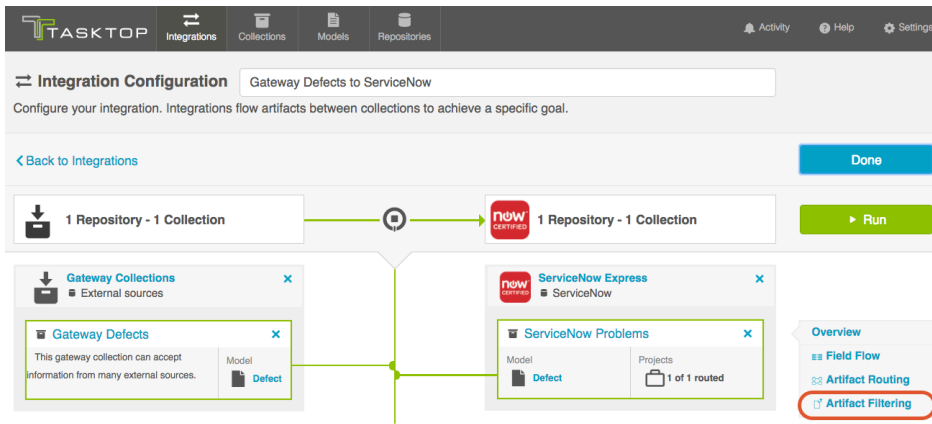


Artifact Filtering enables you to set filters in order to limit which artifacts are eligible to flow in an integration.

To use a field for artifact filtering, it must:

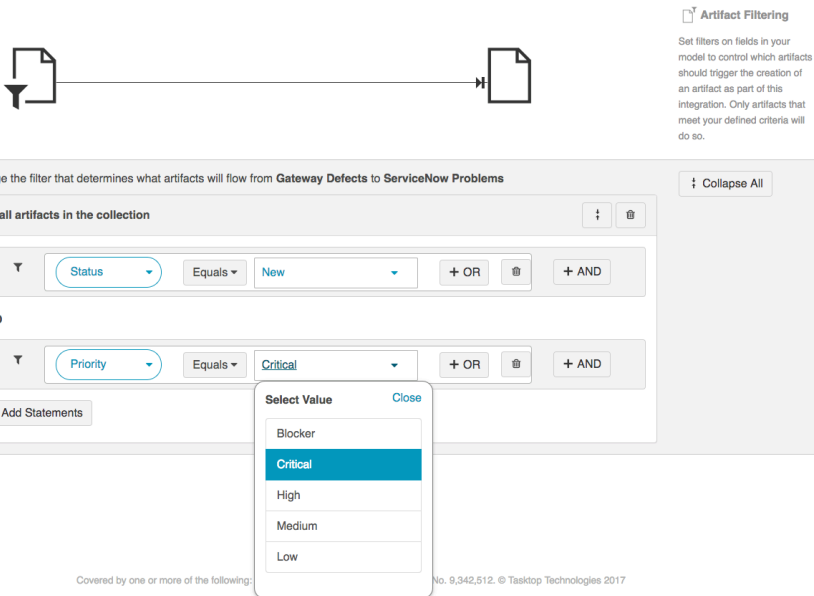
- Be a part of your model, and be mapped to the collection you are filtering from
- Be one of the following field types:
 - Single Select
 - Note that in cases where 'allow unmapped values to flow' is enabled in the model, only fields that are already a part of the model will be considered for artifact filtering
 - Multi-Select
 - Note that in cases where 'allow unmapped values to flow' is enabled in the model, only fields that are already a part of the model will be considered for artifact filtering
 - Date
 - Date/Time
 - Duration
 - String

To configure *Artifact Filtering*, select 'Create filters (optional)' from the Integration Configuration Overview screen, or select 'Artifact Filtering' from the right pane of the Integration Configuration screen.:



This will lead you to the Artifact Filtering Configuration screen, where you can configure one or more criteria for artifact filtering.

💡 You can click the 'Collapse All' button to view an easier-to-read summary of your artifact filtering statements.



Running your Integration

There are two ways to start or stop your integration:

From the Integration Configuration Screen

Simply click the 'Run' button to run the integration, and the 'Stop' button to stop the integration.

Integration Configuration Gateway Defects to ServiceNow

Configure your integration. Integrations flow artifacts between collections to achieve a specific goal.

[Back to Integrations](#) **Done**

1 Repository - 1 Collection → 1 Repository - 1 Collection **Run**

Gateway Collections External sources

- Gateway Defects** This gateway collection can accept information from many external sources. Model: Defect

ServiceNow Express ServiceNow

- ServiceNow Problems** Model: Defect Projects: 1 of 1 routed

Overview

- Field Flow
- Artifact Routing
- Artifact Filtering

Integration Configuration Gateway Defects to ServiceNow

Configure your integration. Integrations flow artifacts between collections to achieve a specific goal.

[Back to Integrations](#) **Done**

1 Repository - 1 Collection → 1 Repository - 1 Collection **Stop**

Gateway Collections External sources

- Gateway Defects** This gateway collection can accept information from many external sources. Model: Defect

ServiceNow Express ServiceNow

- ServiceNow Problems** Model: Defect Projects: 1 of 1 routed

Overview

- Field Flow
- Artifact Routing
- Artifact Filtering

From the Integrations List Page

Click 'Run' or 'Stop' next to each integration you would like to update.
You can also use the 'Bulk Actions' button to run or stop all integrations.

Integrations

View your existing integrations and create new ones. Integrations flow artifacts between collections to achieve a specific goal.

[Back to Integration Configuration](#) **+ New Integration**

Search

Landscape **List** **Bulk Actions**

- ServiceNow Express Incidents to Bugzilla Defects**
 ServiceNow Express Collection: ServiceNow Express Incidents → Bugzilla Collection: Bugzilla Defects **Stop**
- ServiceNow Express Incidents to GitLab Issues**
 ServiceNow Express Collection: ServiceNow Express Incidents → GitLab Collection: GitLab Issues **Stop**
- Gateway Defects to ServiceNow**
 Gateway Collections Collection: Gateway Defects → ServiceNow Express Collection: ServiceNow Problems **Run**

The screenshot shows the 'Integrations' page in Tasktop. At the top, there are navigation tabs for Integrations, Collections, Models, and Repositories. Below the header, there's a search bar and a '+ New Integration' button. A 'Bulk Actions' dropdown menu is highlighted with a red box, containing options for 'Stop All' and 'Run All'. The main content area displays three integration cards:

- ServiceNow Express Incidents to Bugzilla Defects:** A bidirectional integration between 'ServiceNow Express' and 'Bugzilla' collections.
- ServiceNow Express Incidents to GitLab Issues:** A one-way integration from 'ServiceNow Express' to 'GitLab' collections.
- Gateway Defects to ServiceNow:** A one-way integration from 'Gateway Collections' to 'ServiceNow Express' collections.

Viewing Your Integrations

See [Tasktop Editions table](#) to determine if your edition contains Integration Landscape View functionality.

When viewing your integrations, you have the option of viewing them in either Landscape or List mode.

The screenshot shows the 'Integration Landscape' view in Tasktop. It features a grid-based interface with a 'Landscape' view selected. The main area displays a network diagram of integrations between various repositories and models. The nodes in the diagram include:

- Git Changesets
- HPE ALM (qc-270)
- ServiceNow
- JIRA
- Jama
- Reporting Database
- HPE ALM

Green lines connect these nodes, representing the flow of artifacts between them. The interface also includes a filter section on the left and a 'Reset Filter' button.

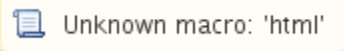
The screenshot shows the Tasktop Integrations page. At the top, there are navigation tabs for Integrations, Collections, Models, and Repositories. Below the header, there's a search bar and a '+ New Integration' button. A 'Landscape' button is highlighted with a red circle, and a 'List' button is also visible. The main content area displays a series of integration flows, each with a title, source collection, target collection, and a 'Run' button. The flows are as follows:

- 1. Help Desk Escalation:** ServiceNow (Collection: ServiceNow Problems) to JIRA (Collection: JIRA Defects from SNOW).
- 2. Requirements (JIRA to HPE ALM):** JIRA (Collection: JIRA Epics) to HPE ALM (Collection: HPE ALM Parent Requirements).
- 3. JIRA Stories to HP Requirements:** JIRA (Collection: JIRA Stories) to HPE ALM (Collection: HPE ALM Requirements).
- 3. Requirements- Jama to JIRA:** Jama (Collection: Jama Requirements) to JIRA (Collection: JIRA Epics).
- 4. Defects (HPE ALM to JIRA):** HPE ALM (Collection: HPE ALM Defects) to JIRA (Collection: JIRA Defects).
- 5. JIRA Changeset Traceability:** Gateway Collections (Collection: Git Changesets) to JIRA (Collection: JIRA Stories).
- 7. Orasi (HPE Defects to JIRA):** JIRA (Collection: JIRA Defects (Orasi)) to HPE ALM (Collection: HPE Defects (Orasi)).
- 7. Orasi (JIRA Stories to HPE ALM Requirements):** JIRA (Collection: JIRA Stories (Orasi)) to HPE ALM (Collection: HPE Requirements (Orasi)).
- Requirement Reporting:** 3 Repositories (3 Collections) to Reporting Database (Collection: Reporting Database).

Landscape View

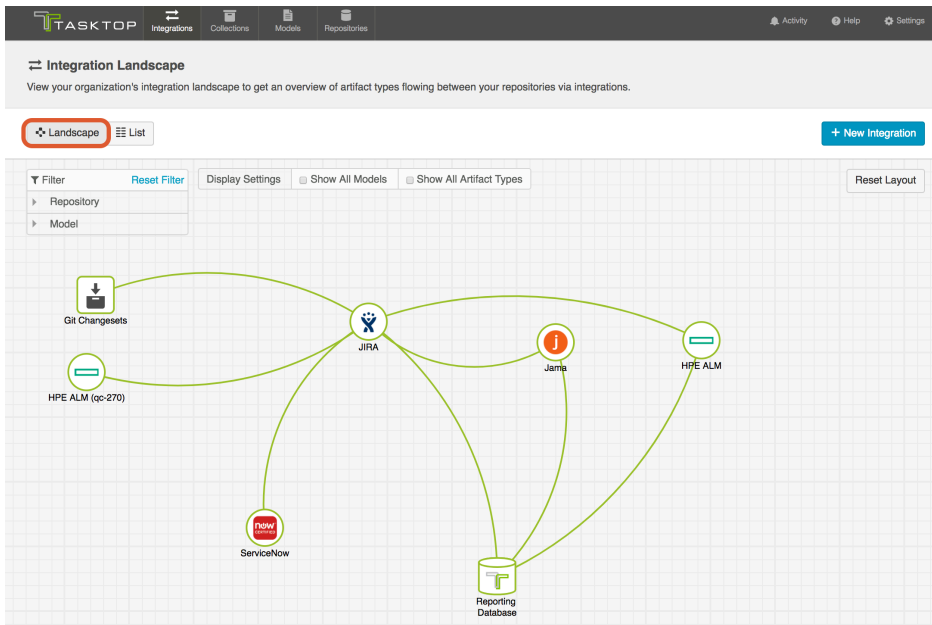
See [Tasktop Editions table](#) to determine if your edition contains Integration Landscape View functionality.

Learn more about the Integration Landscape View in the video below:



Tasktop will default to the Landscape View, which enables you to visualize your entire integration landscape and see how your integrations relate to one another. Use our built-in filters to see as little or as much information as you'd like!

Here's a simplified view:

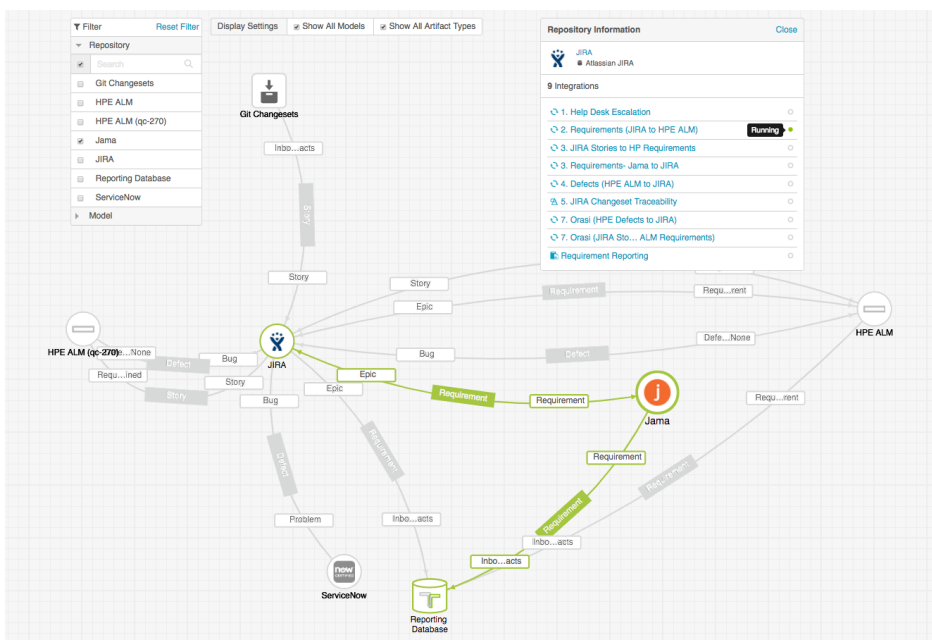


If you'd like to see additional information, you can utilize the filters, or click on a repository node to modify which information is shown.

Some examples of additional information you can see are:

- Models
- Artifact Types
- Artifact Creation Directionality Arrows
- List of all relevant integrations (see this by clicking on the repository node)
 - Indicator of whether each integration is running or not

Here's an example of a more detailed view:



List View

If you'd like, you can toggle to List View, which will show you a list of all

integrations you have created.

You can use this view to:

- Start an Integration
- Stop an Integration
- Delete an Integration
- Click into an Integration and modify its configuration

The screenshot shows the Tasktop Integrations page. At the top, there's a navigation bar with 'TASKTOP' and tabs for 'Integrations', 'Collections', 'Models', and 'Repositories'. Below the navigation bar, there's a header for 'Integrations' with a sub-header: 'View your existing integrations and create new ones. Integrations flow artifacts between collections to achieve a specific goal.' A link for '< Back to Integration Configuration' is visible. A search bar and a '+ New Integration' button are also present. A 'List' button is circled in red. Below the search bar, there's a 'Bulk Actions' dropdown. The main content area displays a list of integrations, each with a title, source collection, target collection, and a 'Run' button. The integrations are:

- 1. Help Desk Escalation: ServiceNow Collection: ServiceNow Problems to JIRA Collection: JIRA Defects from SNOW
- 2. Requirements (JIRA to HPE ALM): JIRA Collection: JIRA Epics to HPE ALM Collection: HPE ALM Parent Requirements
- 3. JIRA Stories to HP Requirements: JIRA Collection: JIRA Stories to HPE ALM Collection: HPE ALM Requirements
- 3. Requirements- Jama to JIRA: Jama Collection: Jama Requirements to JIRA Collection: JIRA Epics
- 4. Defects (HPE ALM to JIRA): HPE ALM Collection: HPE ALM Defects to JIRA Collection: JIRA Defects
- 5. JIRA Changeset Traceability: Gateway Collections Collection: Git Changesets to JIRA Collection: JIRA Stories
- 7. Orasi (HPE Defects to JIRA): JIRA Collection: JIRA Defects (Orasi) to HPE ALM (qp-270) Collection: HPE Defects (Orasi)
- 7. Orasi (JIRA Stories to HPE ALM Requirements): JIRA Collection: JIRA Stories (Orasi) to HPE ALM (qp-270) Collection: HPE Requirements (Orasi)
- Requirement Reporting: 3 Repositories 3 Collections to Reporting Database Collection: Reporting Database

Tips and Tricks

Creating Relationships Between Newly Created Artifacts and Existing Artifacts

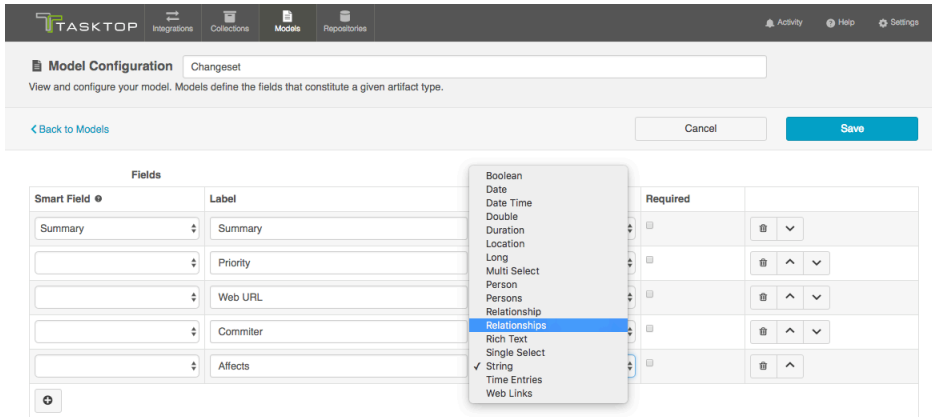
If you'd like to create relationships between your newly created artifacts and existing artifacts in the same repository, please follow the additional steps listed below:

At the Model level: When creating your model, you can create a field that is of type "relationship" or "relationships". You should use "relationship" when the newly-created artifact can only relate to one other artifact and "relationships" when the newly-created artifact can relate to multiple artifacts.

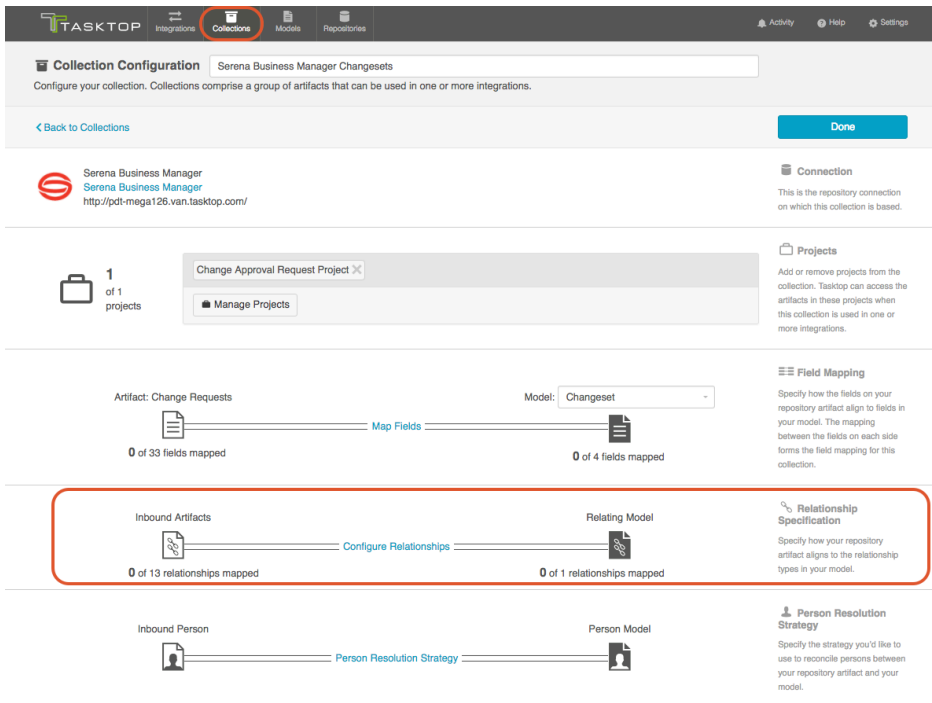
For example, the relationship field type, "Parent," should generally be

singular, as most artifacts usually only have a single parent. However, if the relationship field type is called "Blocks", it can likely be plural, as one artifact can block many artifacts.

In the use case example described at the top of this page, I want the relationship to be "Affects" because any incoming changeset can affect many stories. So I'd configure a *relationships* field.




At the Repository Collection level: When creating your repository collection, you will need to map a field in your repository to the relationship(s) field in your model. So, in the same example, if you want the relationship between the new changeset and the existing story to be "affects", but the relationship is actually called "items linked" in Serena, you would need to map those two fields. You'll need to do this for each relationship type configured in your model.



Relationship Specification: Serena Business Manager Changesets
View and manage the relationship specification for this collection. The relationship specification defines how your repository artifact aligns to the relationship types in your model.


[Back to Collection](#) Cancel Save

Serena Business Manager: Change Requests



1 of 13 relationships mapped

Model: Changeset



1 of 1 relationships mapped

Relationship Specification

Specify how your repository artifact aligns to the relationship types in your model.

Search "Serena Business Manager Changesets" artifact... Select a field on each side to configure additional relationship mappings Search "Changeset" model fields by name or type Suggest Mappings

Item Links % Affects Configure

At the Gateway Collection Level: When creating your Gateway collection, you will see that for each model field that is of relationship(s) type, you must specify the target repository that contains the related artifact(s). Once this is selected, the information needed for Tasktop to successfully locate the artifact will be added to the example Payload.

Gateway Collection
View your existing collections and create new ones. Collections comprise a group of artifacts that can be used in one or more integrations.

[Back to Collections](#) Done

Path

Token

Model

Payload Transformation

Script

Relationship Field Configuration

Affects Choose an existing repository connection

Access Details

Uri

HPE ALM	HPE QC / ALM
Modern Requirements4TFS	Microsoft Team Foundation Server
Serena Business Manager	Serena Business Manager
ServiceNow Express	ServiceNow

Method

Content-Type

Example Payload

```
{
  "summary": "String",
  "priority": "String",
  "web_url": "String",
  "committer": "String",
  "affects": []
}
```

Example Script

```
curl -H 'Content-Type: application/json' --data-binary '{"summary":"String","priority":"String","web_url":"String",
```

Gateway Collection Gerrit Changesets

View your existing collections and create new ones. Collections comprise a group of artifacts that can be used in one or more integrations.

[Back to Collections](#) Cancel Save

Path


Token

Model

Payload Transformation

Script

Relationship Field Configuration

Affects  Serena Business Manager

Access Details

Uri

Method POST

Content-Type application/json

Example Payload

```
{
  "summary": "String",
  "priority": "String",
  "web_url": "String",
  "committer": "String",
  "affects": []
}
```

Example Script

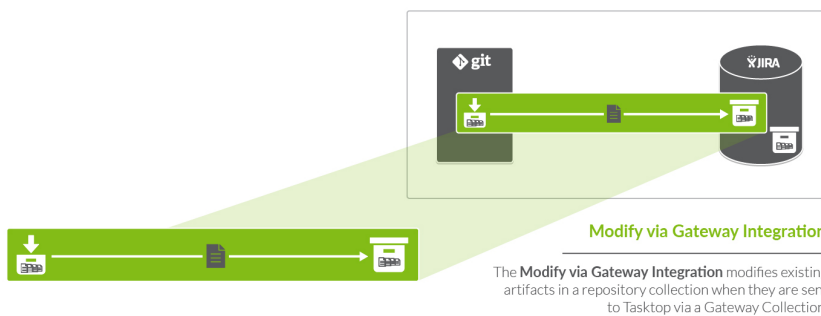
```
curl -H 'Content-Type: application/json' --data-binary '{"summary":"String","priority":"String","web_url":"String",
```

Modify via Gateway Template

Tasktop: 17.4 Release

The Modify via Gateway Integration Template is only available in Editions that contain the Gateway add-on. See [Tasktop Editions table](#) to determine if your edition contains this functionality.

What is a Modify via Gateway Integration?



An *integration* is quite simply the flow of information between two or more collections. A *Modify via Gateway Integration*, specifically, locates and modifies existing artifacts in a repository collection when they are sent to Tasktop via a Gateway Collection. A Gateway Collection accesses information in an external tool, such as Git or Jenkins, via an inbound webhook.

When you configure a Modify via Gateway integration, you can

- What is a Modify via Gateway Integration?
- Video Tutorial
- Use Case and Business Value
- Template
- Affordances
- How to Configure a Modify via Gateway Integration
 - Configuring your Repository Collection
 - Configuring Your Integration
 - Field Flow
 - Specifying Your Key
 - Configure Field Flow
 - Artifact Filtering
 - Running your Integration
 - From the Integrati

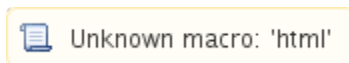
customize the field flow and artifact filtering.

on

Video Tutorial

Check out the video below to learn how to configure the Modify via Gateway Integration Template.

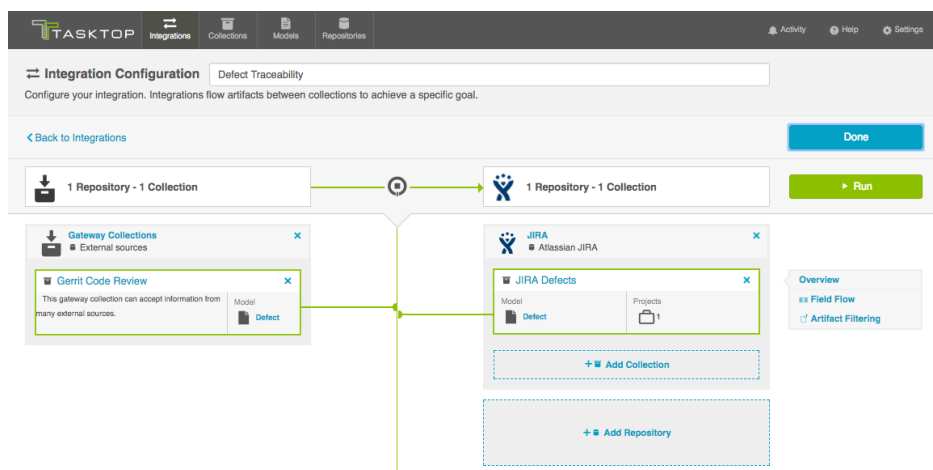
⚠ This video assumes that you have already configured your repositories, models, and collections as outlined in the [Quick Start Guide](#)



Use Case and Business Value

The 'Modify via Gateway' integration creates traceability between artifacts across the software development lifecycle. Already existing artifacts in a repository collection will be located and modified in a specified way when artifacts are sent to Tasktop via a Gateway collection.

For example, if your development team uses Gerrit for code review and JIRA for its agile work management, but would like to know which defects in JIRA a given code review affects, or conversely which code reviews are associated with a given defect, you could set up an integration that would find an already-existing defect in JIRA anytime a code review is sent in and append one of its fields with that code review's URL. The integration can even include updating other JIRA artifacts to which code reviews might pertain, such as stories and tech debt.



Template Affordances

The Modify via Gateway Integration Template allows you to update already-existing artifacts in target repository collections when artifacts are sent to Tasktop via a Gateway collection.



Configuration Screen

- From the Integrations List Page
- Viewing Your Integrations
 - Landscape View
 - List View
- Example Use Case

How to Configure a Modify via Gateway Integration

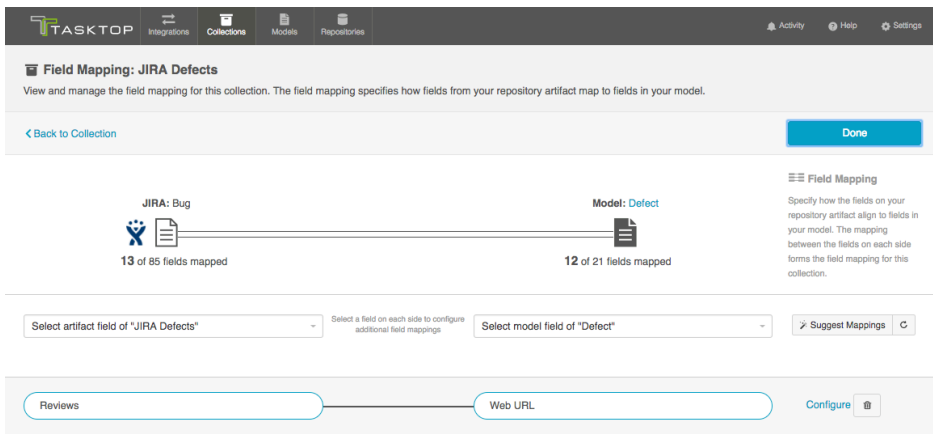
Configuring your Repository Collection

Before you begin configuring the integration itself, there are some steps that must be taken at the repository collection level:

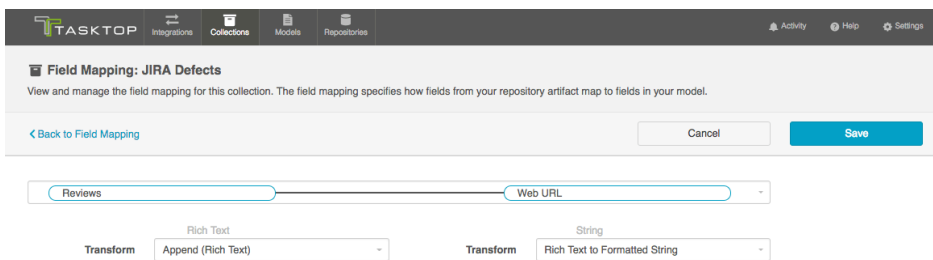
To specify just how you would like incoming artifacts from your gateway collection to modify already existing artifacts in your repository collection, you need to identify which field(s) on your already-existing artifacts you would like to modify and then configure how the field(s) should be changed. In the example above, the URL to any incoming code reviews from a gateway collection is being added to the review field of the JIRA defect.

This means that the JIRA collection-to-model mapping is configured as such.

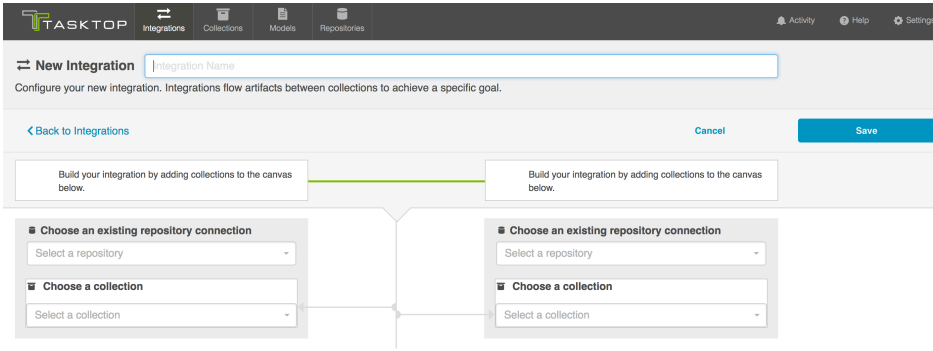
Here is the collection to model mapping:



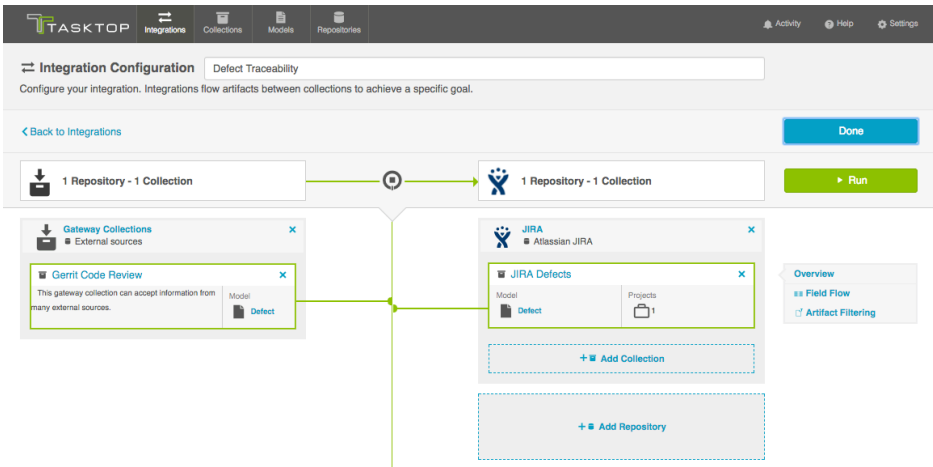
And here are how the transformations are configured between these fields:



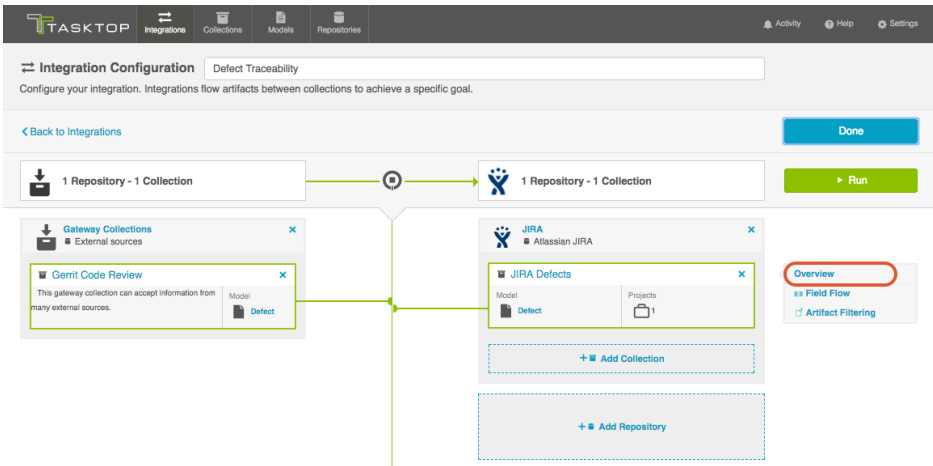
You can see that information coming in from the Gateway collection and through the model is appended to the JIRA reviews field, leaving the JIRA artifact itself looking like this:

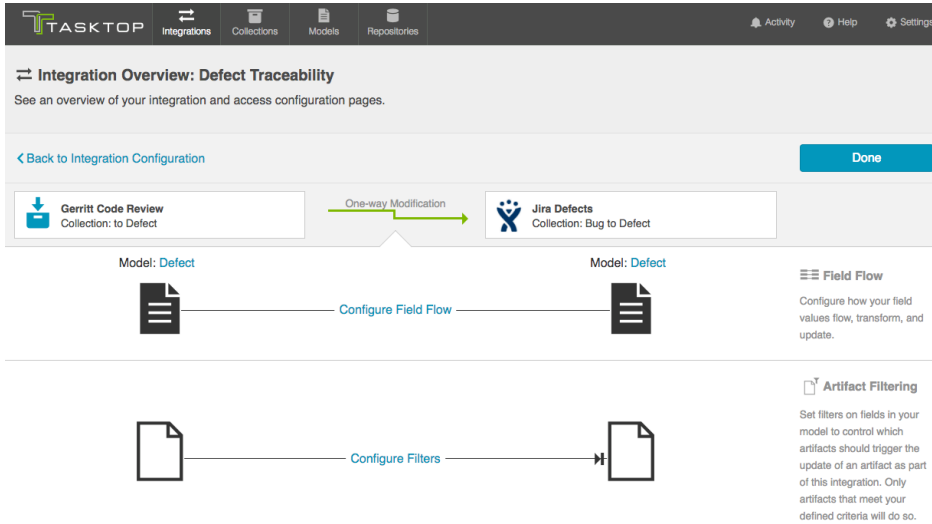


Name your integration and select your repositories and collections:



You can click the 'Overview' link on the right side of the Integration Page to get to the main display page (shown in the second screen shot):

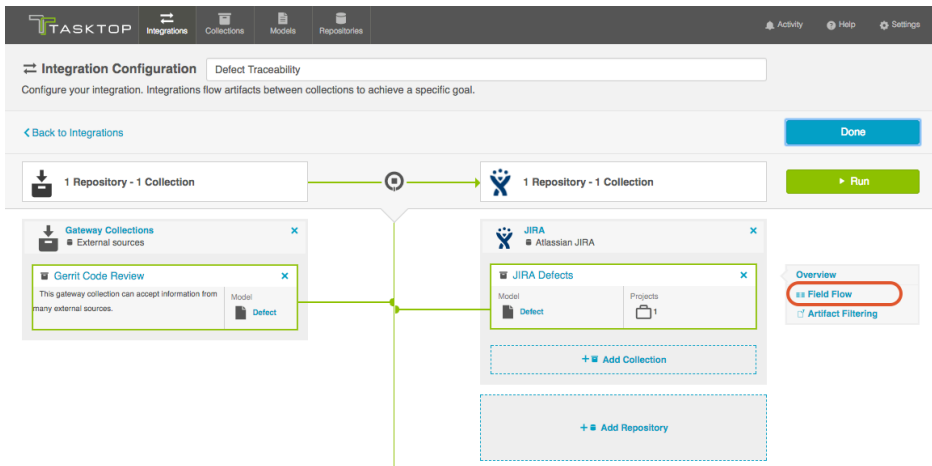




Field Flow

The field flow configured for a given integration specifies which fields should flow in that integration. For Modify via Gateway integrations, you can choose to flow a given field (Update Normally) or to not flow a given field (No Update).

To get to the Field Flow screen, click 'Field Flow' on the right side of the Integration Configuration screen:



Specifying Your Key

The first thing you will need to do when you get to the Field Flow screen is to specify your key.

Specifying a key will enable Tasktop to find the existing artifact in your repository collection that is to be modified by the incoming gateway artifact(s). The key contains information, such as the ID of an artifact, that Tasktop can use to locate the existing artifact in your repository.

The field used as your key must be a part the model mapped to your collection.

TASKTOP Integrations Collections Models Repositories Activity Help Settings

Field Flow: Defect Traceability

View and manage the field flow between these collections. Field flow specifies how your field values will flow and transform as part of this integration.

[Back to Integration Configuration](#) Done

Gerritt Code Review
Collection: to Defect

One-way Modification

Jira Defects
Collection: Bug to Defect

Model: Defect

4 fields mapped

Model: Defect

Field Flow

Configure how your field values flow, transform, and update.

The key is used to locate the artifacts in the receiving collection that you would like to modify. Only string and relationship fields are valid key fields. If a string field is selected the value will be used to search for an artifact by formatted id. If a relationship field is selected then the artifact it references will be updated.

Specify Key

Not applicable

Formatted ID

→

Formatted ID

* Key

Hide mapped artifact fields

Not applicable

* Summary

→

* Summary

-- (Summary)

TASKTOP Integrations Collections Models Repositories Activity Help Settings

Field Flow: Defect Traceability

View and manage the field flow between these collections. Field flow specifies how your field values will flow and transform as part of this integration.

[Back to Integration Configuration](#) Cancel Save

Gerritt Code Review
Collection: to Defect

One-way Modification

Jira Defects
Collection: Bug to Defect

Model: Defect

4 fields mapped

Model: Defect

Field Flow

Configure how your field values flow, transform, and update.

The key is used to locate the artifacts in the receiving collection that you would like to modify. Only string and relationship fields are valid key fields. If a string field is selected the value will be used to search for an artifact by formatted id. If a relationship field is selected then the artifact it references will be updated.

Specify Key

Specify the key for this integration pair. Close

Formatted ID

→

Formatted ID

* Key

Hide mapped artifact fields

Summary

→

* Summary

-- (Summary)

Note: Some repositories require extra information in order to uniquely identify a single artifact across multiple projects. One prime example is HPE. To ensure that enough information is sent in via your Gateway collection to allow Tasktop to find the specific artifact you would like to modify, please take these steps:

1. Add a field in your model of type Relationship

Model Configuration Defect

View and configure your model. Models define the fields that constitute a given artifact type.

[Back to Models](#) Cancel Save

Smart Field	Label	Type	Required	
Formatted ID	Formatted ID	String	<input type="checkbox"/>	🗑️ ↓
Summary	Summary	String	<input checked="" type="checkbox"/>	🗑️ ^ ↓
Description	Description	Rich Text	<input type="checkbox"/>	🗑️ ^ ↓
Severity	Severity	Single Select <small>Field Values... Sev 1, Sev 2, Sev 3, ... Allow unmapped values to flow: No</small>	<input type="checkbox"/>	🗑️ ^ ↓
	Target Artifact	Relationship	<input type="checkbox"/>	🗑️ ^

- In your Gateway collection, notice that for the new field you are prompted to pick a target repository. Select the repository you'd like to target in this Gateway Integration

Gateway Collection Gerrit Code Review

View your existing collections and create new ones. Collections comprise a group of artifacts that can be used in one or more integrations.

[Back to Collections](#) Cancel Save

Path:

Token:

Model:

Payload Transformation:

Relationship Field Configuration

Target Artifact: Choose an existing repository connection

Select a repository

- HPE ALM** (Selected)
 - HPE QC / ALM
- JIRA
 - Atlassian JIRA
- MySQL
 - Tasktop SQL

Access Details

Uri:

Method:

- When you save, note that the example payload will be updated to include the pieces of information we need for that field to uniquely find artifacts

Gateway Collection Gerritt Code Review

View your existing collections and create new ones. Collections comprise a group of artifacts that can be used in one or more integrations.

[Back to Collections](#) Done

Path: gerritt-code-review

Token: 75489207-b9ef-4b5e-be45-6614c0bfoebb

Model: Defect

Payload Transformation: None

Relationship Field Configuration

Target Artifact: HPE ALM (HPE QC / ALM)

Access Details

Uri: [Redacted]

Method: POST

Content-Type: application/json

Example Payload

```
{
  "formatted_id": "String",
  "summary": "String",
  "description": "text with <em>HTML</em> tags",
  "severity": "Sev_1",
  "target_artifact": {
    "domain": "DEFAULT",
    "project": "Case10664",
    "formattedid": "String"
  }
}
```

4. Finally, in your integration select that field as your key on the Field Flow screen.

TASKTOP Integrations Collections Models Repositories Activity Help Settings

Field Flow: HPE Defect Traceability

View and manage the field flow between these collections. Field flow specifies how your field values will flow and transform as part of this integration.

[Back to Integration Configuration](#) Cancel Save

Gerritt Code Review Collection: to Defect One-way Modification HPE Defects Collection: Defect - None to Defect

Model: Defect 5 fields mapped Model: Defect Field Flow

Target Artifact

Specify the key for this integration pair. Close

Formatted ID Formatted ID Hide mapped artifact fields

Summary * Summary --> (Summary)

Target Artifact Description (Description)

Configure Field Flow

Once you have specified your key, you can configure your field flow. For each field, you can choose to flow information ('update normally') or not flow information ('no update'). You'll notice that field flow goes in one direction only - from the gateway collection *into* the repository or database collection.

You can see the names of the mapped artifact fields for each collection on the far left and far right, with the model fields displayed in the middle. To hide the mapped artifact fields, select 'Hide mapped artifact fields' on the right.

⚠ Note: The field flow settings behave a bit differently for Constant Values. This is because constant values exist as part of your Tasktop








configuration, and not on the artifact itself. Therefore, changes in constant values are not detected in the same way that updates made on the actual artifact are detected. If you change the constant value that is linked to your model, your integration will not automatically detect this update and sync it over. The value will only update if another field on that artifact is updated.

The screenshot displays the TASKTOP interface for configuring field flow. At the top, navigation tabs include Integrations, Collections, Models, and Repositories. The main heading is "Field Flow: HPE Defect Traceability". Below this, a "One-way Modification" arrow points from "Gerritt Code Review" to "HPE Defects". A "5 fields mapped" status is shown between the two collections. A "Specify Key" section shows a mapping of "Formatted ID" fields. A "Select Field Flow" dialog is open, showing "Update Normally" selected for the "Formatted ID" field.

Field Flow Icons

On the Field Flow page, you will see a number of icons, which will help you understand any special properties or requirements for each field. If you hover your mouse over an icon, you will see a pop-up explaining what the icon means. You can also review their meanings in the legend below:

Icon	Meaning
------	---------

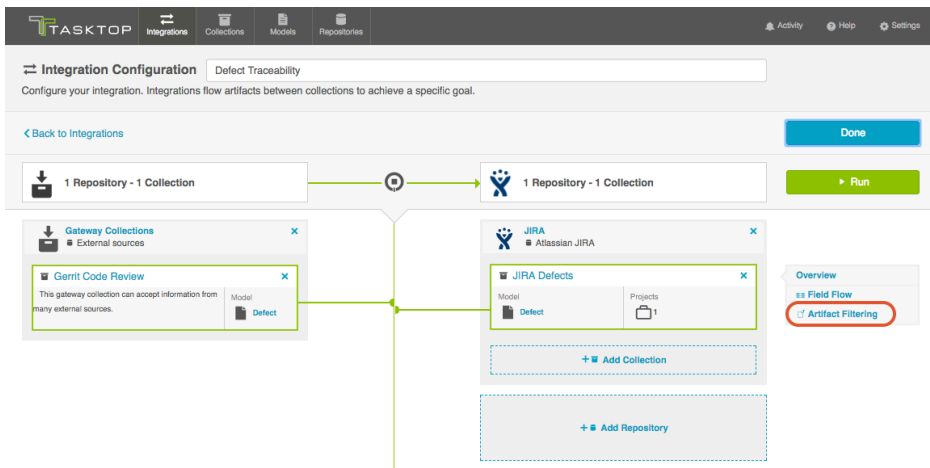
	<p>A constant value will be sent.</p> <p>Note that:</p> <ul style="list-style-type: none"> • If the icon is on the side of the collection, this means that a constant value will be sent to your model. This means that any time this collection is integrated with another collection, the <i>other</i> collection will receive this constant value for the field in question. • If the icon is on the side of the model, this means a constant value will be sent to your collection. This means that any time this collection is integrated with another collection, that <i>this</i> collection will receive this constant value for the field in question.
	<p>Collection field is read-only and cannot receive data</p>
	<p>To create artifacts in your collection, this field must be mapped to your model.</p>
	<p>This is a required field in your model; it must be mapped to your collection.</p>
	<p>This field will not be updated as part of your integration, due to how you have configured it. This field flow configuration can be changed if you'd like.</p>
	<p>This field will not be updated as part of your integration because the mapping would be invalid. You do not have the option of changing this.</p>
	<p>This field will update normally as part of your synchronize integration; this means it will be updated whenever it is modified on the corresponding artifact.</p>

Artifact Filtering enables you to set filters on an integration in order to limit which artifacts are eligible to flow in your integration.


To use a field for artifact filtering, it must:

- Be a part of your model, and be mapped to the collection you are filtering from
- Be one of the following field types:
 - Single Select
 - Note that in cases where 'allow unmapped values to flow' is enabled in the model, only fields that are already a part of the model will be considered for artifact filtering
 - Multi-Select
 - Note that in cases where 'allow unmapped values to flow' is enabled in the model, only fields that are already a part of the model will be considered for artifact filtering
 - Date
 - Date/Time
 - Duration
 - String

To configure *Artifact Filtering*, select 'Create filters (optional)' from the Integration Configuration Overview screen, or select 'Artifact Filtering' from the right pane of the Integration Configuration screen.



This will lead you to the Artifact Filtering Configuration screen, where you can configure one or more criteria for artifact filtering.

 You can click the 'Collapse All' button to view an easier-to-read summary of your artifact filtering statements.



Artifact Filtering

Set filters on fields in your model to control which artifacts should trigger the update of an artifact as part of this integration. Only artifacts that meet your defined criteria will do so.

Manage the filter that determines what artifacts will flow from Gerrit Code Review to Jira Defects

For all artifacts in the collection

Release Defec... Equals July 2016 + OR + AND

AND

Priority Equals Critical + OR + AND

+ Add Statements

Select Value Close

- Blocker
- Critical
- High
- Medium
- Low

+

Collapse All

Running your Integration

There are two ways to start or stop your integration:

From the Integration Configuration Screen

Simply click the 'Run' button to run the integration, and the 'Stop' button to stop the integration.

TASKTOP Integrations Collections Models Repositories Activity Help Settings

Integration Configuration Defect Traceability

Configure your integration. Integrations flow artifacts between collections to achieve a specific goal.

< Back to Integrations Done

1 Repository - 1 Collection → 1 Repository - 1 Collection Run

Gateway Collections External sources

- Gerrit Code Review Model Defect

JIRA Atlassian JIRA

- JIRA Defects Model Defect Projects 1

+ Add Collection

+ Add Repository

Overview

- Field Flow
- Artifact Filtering

TASKTOP Integrations Collections Models Repositories Activity Help Settings

Integration Configuration HPE Changeset Traceability

Configure your integration. Integrations flow artifacts between collections to achieve a specific goal.

< Back to Integrations Done

1 Repository - 1 Collection → 1 Repository - 1 Collection Stop

Gateway Collections External sources

- Gateway Changesets Model Changeset

HPE ALM HPE QC / ALM

- HPE Requirements Model Changeset Projects 2

+ Add Collection

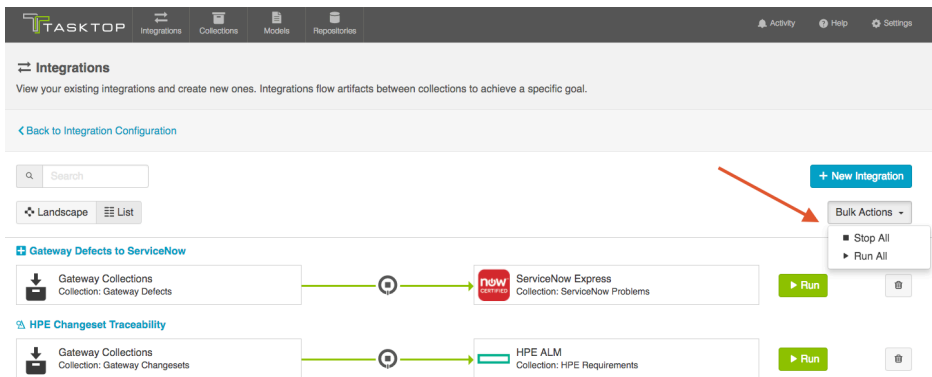
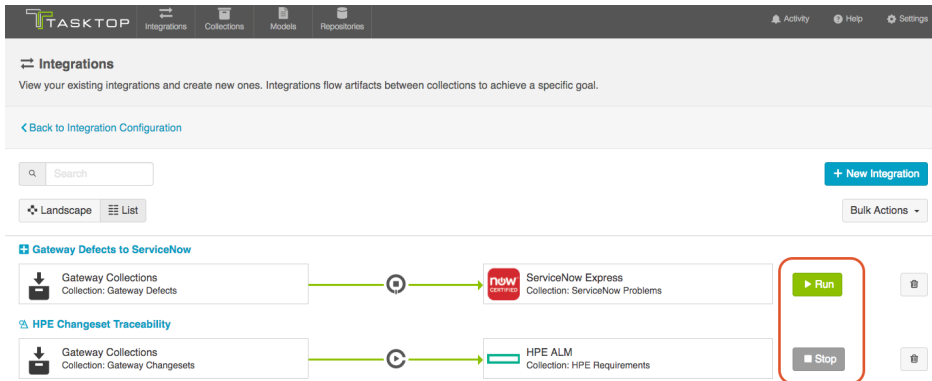
+ Add Repository

Overview

- Field Flow
- Artifact Filtering

From the Integrations List Page

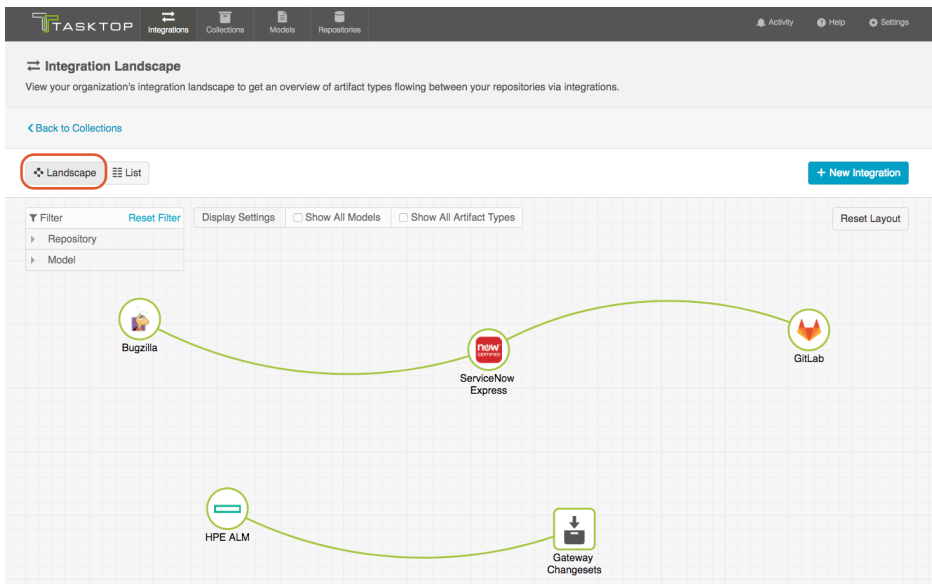
Click 'Run' or 'Stop' next to each integration you would like to update.
You can also use the 'Bulk Actions' button to run or stop all integrations.

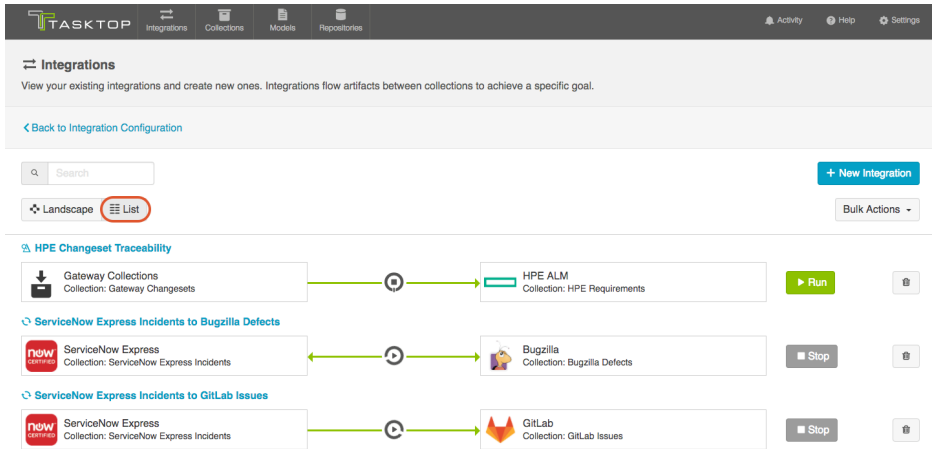


Viewing Your Integrations

See [Tasktop Editions table](#) to determine if your edition contains Integration Landscape View functionality.

When viewing your integrations, you have the option of viewing them in either Landscape or List mode.

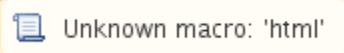




Landscape View

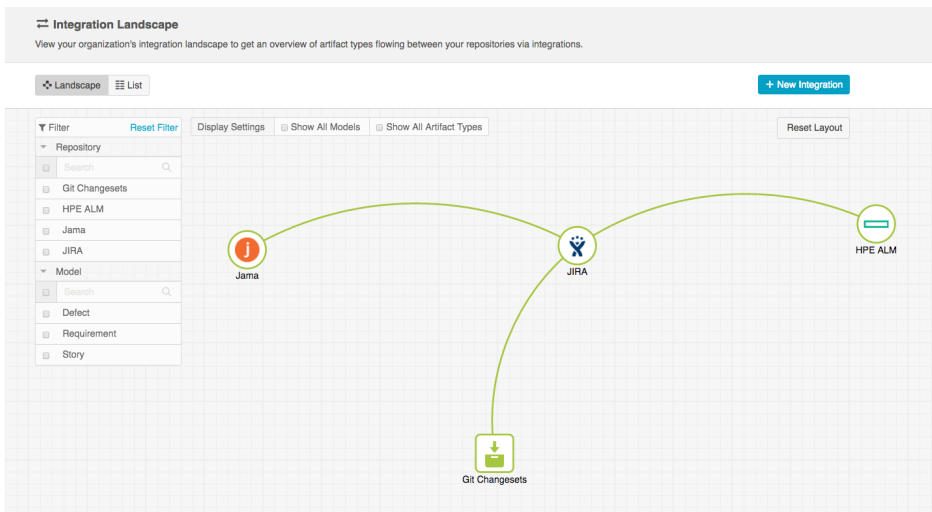
See [Tasktop Editions table](#) to determine if your edition contains Integration Landscape View functionality.

Learn more about the Integration Landscape View in the video below:



Tasktop will default to the Landscape View, which enables you to visualize your entire integration landscape and see how your integrations relate to one another. Use our built-in filters to see as little or as much information as you'd like!

Here's a simplified view:



If you'd like to see additional information, you can utilize the filters, or click on a repository node to modify which information is shown.

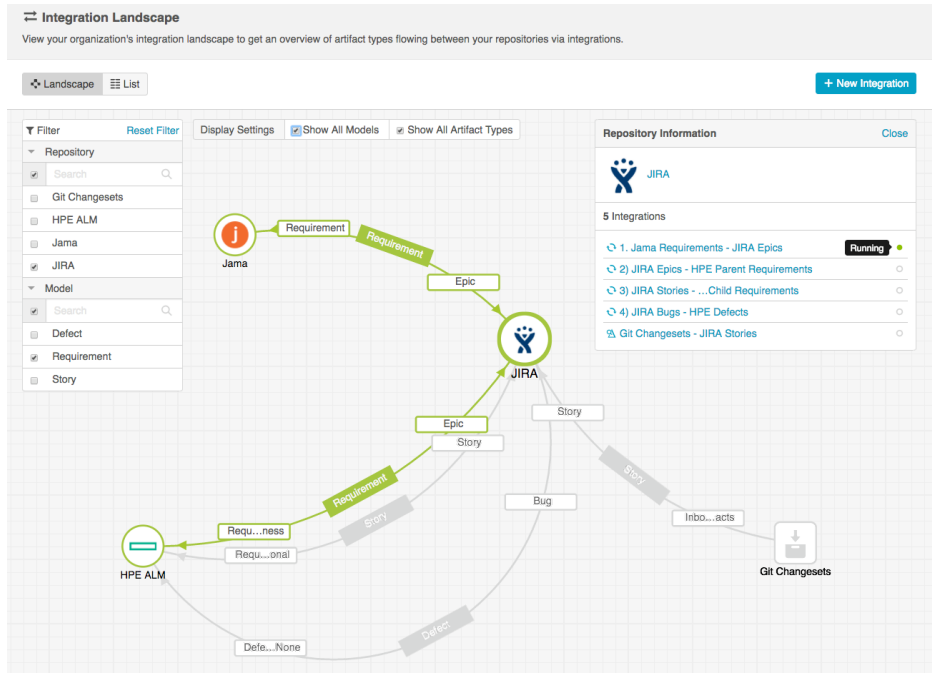
Some examples of additional information you can see are:

- Models
- Artifact Types
- Artifact Creation Directionality Arrows
- List of all relevant integrations (see this by clicking on the

repository node)

- Indicator of whether each integration is running or not

Here's an example of a more detailed view:

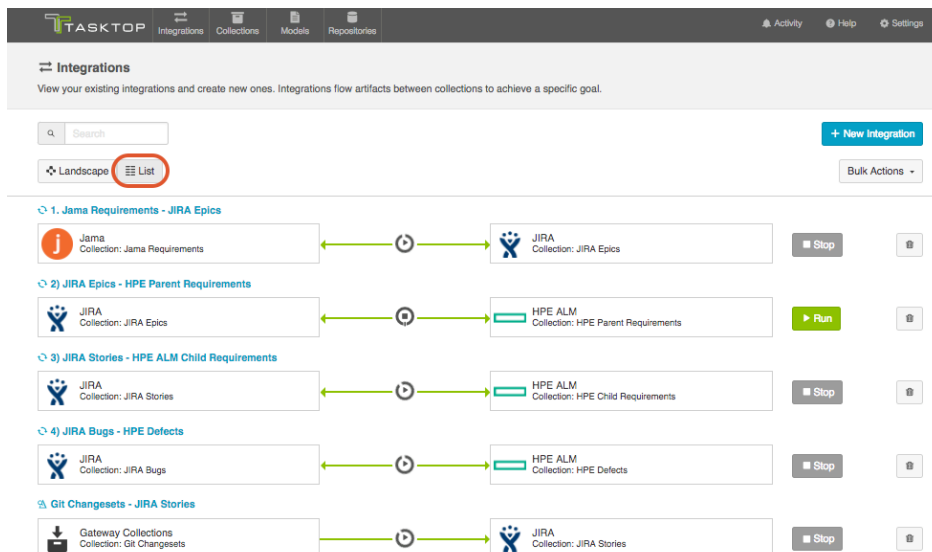


List View

If you'd like, you can toggle to List View, which will show you a list of all integrations you have created.

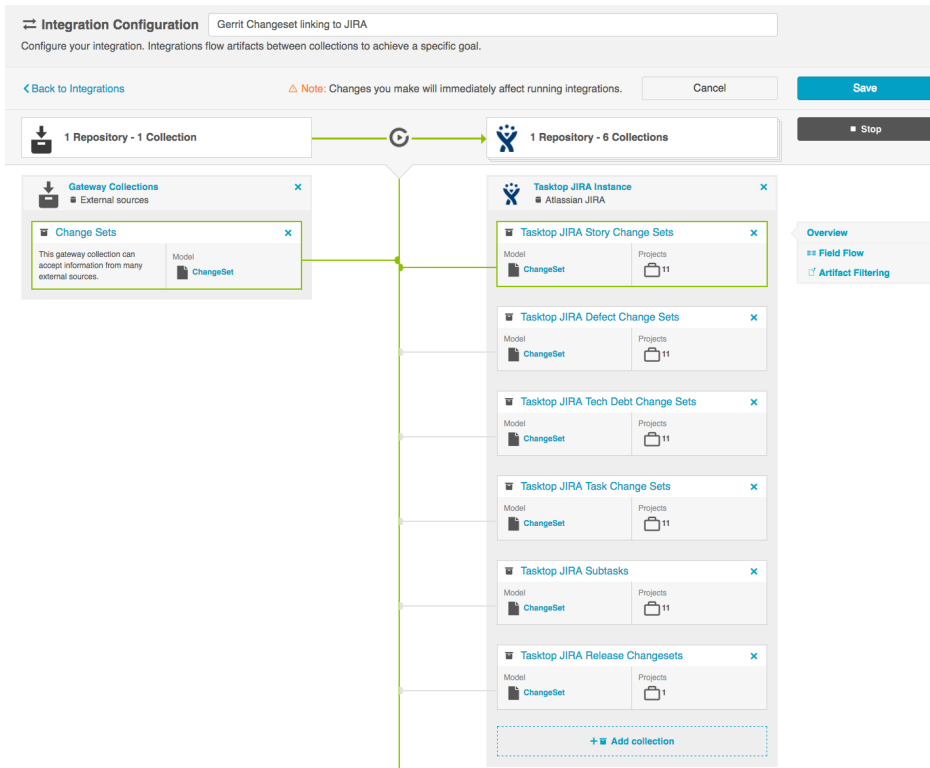
You can use this view to:

- Start an Integration
- Stop an Integration
- Delete an Integration
- Click into an Integration and modify its configuration

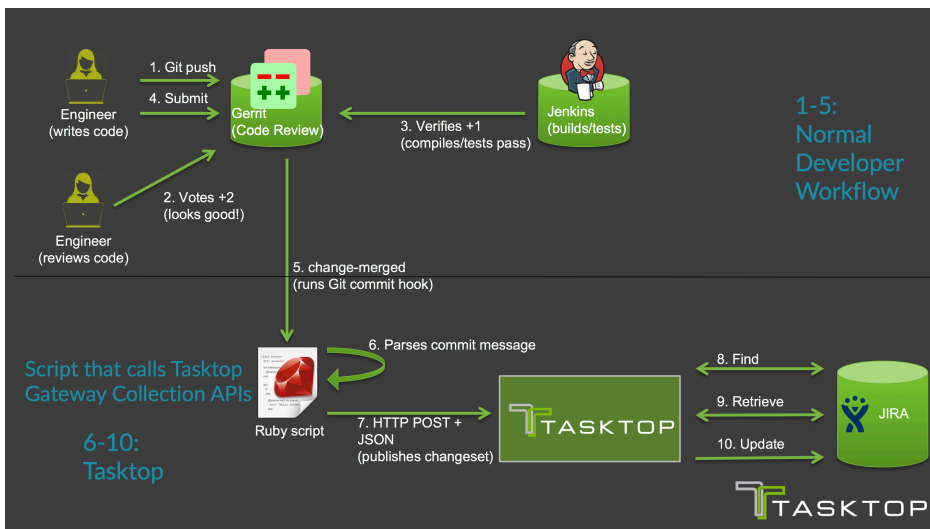


Example Use Case

This is an example of how we at Tasktop utilize the Modify via Gateway template. On the integration canvas, our integration, in which incoming changesets are modifying already-existing artifacts in JIRA, looks like this:



The image below illustrates just how the changeset is sent to Tasktop after the developers' normal workflow, at which point they then participate in the integration show above.



This is an example of the script that we use to automate the changesets being sent to Tasktop:

```
#!/usr/bin/ruby
```

```

require 'rubygems'
require 'logger'
require 'net/http'
require 'openssl'
require 'json'

def getOption(name)
  return ARGV[ARGV.index("--"+name)+1]
end

def sendToLink(data)
  request = Net::HTTP::Post.new(LINK_URL)
  request.body = JSON.generate(data)
  request.content_type = 'application/json'
  request.basic_auth "tasktop-platform",
"tasktopSecret"
  uri = URI.parse(LINK_URL)
  response = Net::HTTP.start(uri.hostname,
uri.port, :use_ssl => uri.scheme == 'https',
:verify_mode => OpenSSL::SSL::VERIFY_NONE) do
|http|
  http.request(request)
  end
  if ! response.kind_of? Net::HTTPSuccess
    LOGGER.warn "Error sending to link:
#{response.body}"
  end
end

LINK_URL =
"https://tt-data350:8443/api/v1/artifacts/changese
ts"
TASK_ID_PATTERN =
/Task-Url:\s*https:\/\/tasktop.atlassian.net\/brow
se\/([\s]*)\/
REVIEW_URL_PATTERN = /\.Reviewed-on:\s+([\s]*)\/m
LOGGER =
Logger.new('/shared/gerrit/tasktop-site/logs/hook-
change-merged.log','monthly')
ENABLED_PROJECT_KEYS = ["APPS", "SYN", "SDK",
"PLAT", "OPS", "CON", "DEV", "QA", "RLIASE"]

project = getOption('project')
commit = getOption('commit')
branch = getOption('branch')

LOGGER.debug("Processing merge for commit
#{commit} on project #{project}")

gitPath = ENV['GIT_DIR']
message = `git --git-dir #{gitPath} show -s

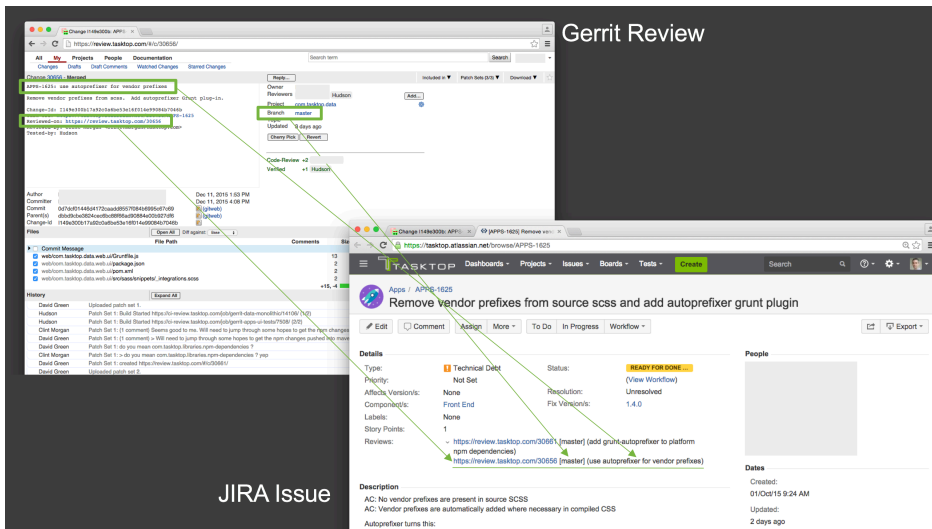
```

```

--format=%B  #{commit}`
taskIdMatch = TASK_ID_PATTERN.match(message)
if taskIdMatch
  taskKey = taskIdMatch.captures[0]
  LOGGER.debug("Detected taskKey: #{taskKey}")
  taskKeyMatches = ENABLED_PROJECT_KEYS.any? {
|project| taskKey.start_with?(project + "-")}
  if ! taskKeyMatches
    LOGGER.info("#{taskKey} project not enabled,
skipping");
    exit()
  end
  reviewUrlMatch =
REVIEW_URL_PATTERN.match(message)
  webUrl = nil
  if reviewUrlMatch
    webUrl = reviewUrlMatch.captures[0]
  else
    LOGGER.error("Could not get webUrl from commit
#{commit}")
    webUrl = "commit #{commit}"
  end
  firstLineOfMessage = message.lines.first.chomp
  firstLineOfMessage =
firstLineOfMessage.gsub(/#{taskKey}:? /, '')
  sendToLink({"formatted_id" => taskKey, "info" =>
"#{webUrl} [#{branch}] (#{firstLineOfMessage})"})
else
  LOGGER.debug("No task key found")
end

```

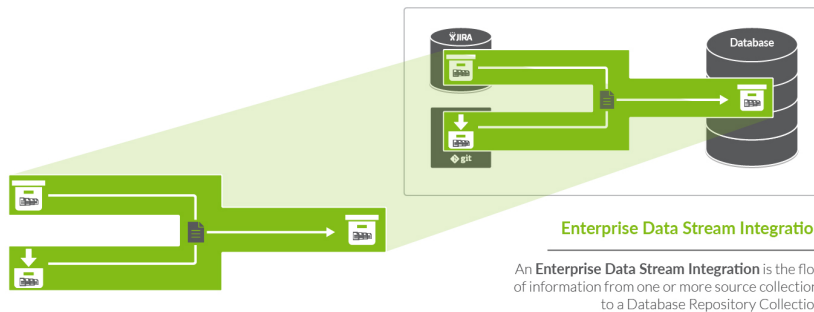
This image more clearly highlights how these changesets are reflected on the JIRA artifacts:



Tasktop: 17.4 Release

The Enterprise Data Stream Template is only available in Editions that contain the Enterprise Data Stream add-on. See [Tasktop Editions table](#) to determine if your edition contains this functionality.

What is an Enterprise Data Stream Integration?



An *integration* is quite simply the flow of information between two or more collections. An Enterprise Data Stream Integration, specifically, is the flow of information from one or more source collections (either Repository Collections or Gateway Collections) to one central table held in a Database Collection.

When you configure your Enterprise Data Stream Integration, you can customize the field flow, artifact routing, and artifact filtering.

Video Tutorial

Check out the video below to learn how to configure an Enterprise Data Stream Integration.

⚠️ This video assumes that you have already configured your repositories, models, and collections as outlined in the [Quick Start Guide](#).

Unknown macro: 'html'

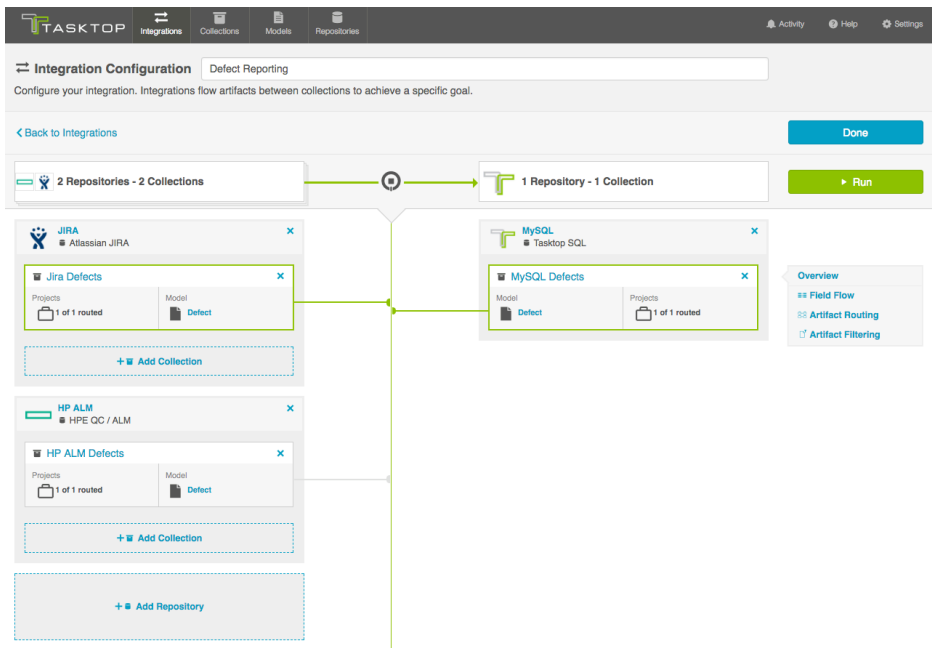
Use Case and Business Value

This integration simplifies enterprise reporting by unlocking software lifecycle data from its application tool silos and providing a rich data repository for near real-time analytics. Records will be created in a single database when artifacts from one or more collections are created or changed.

For example, if your organization uses multiple tools for defect discovery and resolution, such as Atlassian JIRA and HPE ALM, but would like to report on defects across both of the tools, you could set up an integration that would flow artifacts from your JIRA and HPE ALM collections into a single database table. You could then report directly

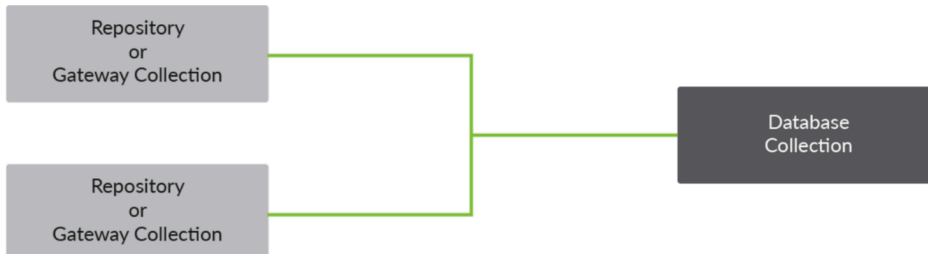
- What is an Enterprise Data Stream Integration?
- Video Tutorial
- Use Case and Business Value
- Template Affordances
- Before You Start
 - Data Structures
 - Database Output
- How to Configure
 - Field Flow
 - Artifact Routing
 - Artifact Filtering
 - Running your Integration
 - Viewing Your Integrations
- Reporting
 - To ETL or Not To ETL?
 - Example Reports

from this aggregated table or, more likely, ETL it into your existing reporting infrastructure.



Template Affordances

The Enterprise Data Stream Template allows you to flow artifacts from multiple repository collections and/or gateway collections into a single database collection.



Gateway Collections are only available in editions that contain the Gateway add-on. See [Tasktop Editions table](#) to determine if your edition has this functionality.

Before You Start

Before you begin, here are a few concepts it's important to understand when configuring an Enterprise Data Stream Integration

Data Structures

An Enterprise Data Stream Integration populates a table with rows corresponding to the state of artifacts at a specific point in time. As an artifact changes, new rows are inserted corresponding to the new state of the artifact. The result is that each artifact has a series of rows

corresponding to the state of the artifact at each point in time. The rows for all artifacts in a table can be thought of as an event stream.

Please note: Tasktop will examine your repositories for changes as specified in the [polling interval](#) that you have configured. This means that if you have configured the polling interval to be 1 minute, and a given artifact is changed twice in that minute, you'll only get a single record that reflects both changes.

The database table populated by the Enterprise Data Stream Integration has columns corresponding to fields in the artifact model, as well as some built-in fields that are designed to facilitate reporting. The following is an example of a database table corresponding to a simple Defect model:

```
CREATE TABLE `Defect` (  
  `id` BIGINT (19) AUTO_INCREMENT,  
  `formatted_id` VARCHAR (1000) NOT NULL,  
  `project` VARCHAR (255) NOT NULL,  
  `type` VARCHAR (255) NOT NULL,  
  `severity` VARCHAR (255) NOT NULL,  
  `status` VARCHAR (255) NOT NULL,  
  `summary` VARCHAR (1000) NOT NULL,  
  `repository_id` VARCHAR (255),  
  `repository_url` VARCHAR (255),  
  `artifact_id` VARCHAR (255),  
  `artifact_url` VARCHAR (255),  
  `artifact_event_type` VARCHAR (255),  
  PRIMARY KEY (`id`)  
);
```

Database Output


Default Information that Tasktop will Flow

The following columns represent information that will automatically be flowed to your database table.

Column	Description
id*	A surrogate key, can be used in reports to uniquely identify a row.
repository_id*	The unique identifier of the connection, can be used in reports to identify a repository connection.

<code>repository_url*</code>	The URL of the repository, can be used in reports to identify a repository.
<code>artifact_id*</code>	An id of an artifact that is globally unique, can be used in reports to uniquely identify an artifact across repositories and collections. The value of the <code>artifact_id</code> is an opaque value; assumptions should not be made about its structure or content. It should be noted that the <code>artifact_id</code> does not correspond to the id of the artifact as it is represented in the repository itself, but is useful for reporting since it is globally unique.
<code>artifact_url</code>	The URL of the artifact for browser access, can be used in reports to identify an artifact.
<code>artifact_event_type</code>	The type of event for the artifact that caused this entry. It can be used to see if the artifact has been added, changed or removed from the collection.

*Denotes that this is a required field, meaning that your target database table will need to have a column to store this information.

 Note: If you use the Suggest DDL to create your table, all of the fields above will be included. If you are creating your table without that mechanism, you'll need to ensure that a column exists for the required pieces of information and, ideally, for the non-required fields as well. Your database table columns will need to be named as displayed above in either upper or lower case, but with the underscores as displayed.

Ordering of Rows

Though it may appear that rows in the table are inserted in an order corresponding to the point in time that changes occurred, the order of rows in the table is not guaranteed. Reports should use a mapped field from the model (such as `modified`) to determine when a change occurred.

Artifact Event Type

In the artifact event type column of your database table, you'll see either "changed", "removed", or "filtered"

Changed

Changed indicates that either an existing artifact was changed or that a new artifact was added to your collection.

Removed

Removed indicates that a given artifact is in a project that has been removed from the collection. Here is a sample scenario to illustrate this event type:

In this Enterprise Data Stream Integration Project B and C are routed to the database table in my SQL collection at the start of an integration. Artifacts flow and records get written out:

id	formatted_id	project	type	created	modified	severity	status	summary	description	repository_id	repository_url	artifact_id	artifact_url	artifact_event_ty...
1	TPB-8	Test...	Bug	2016-...	2016-0...	Blocker	To Do	d33269d5e...	desc	c004d8cc-6...	http://ga-jira...	[com.ta...	http://ga-j...	changed
2	TPB-1	Test...	Bug	2015-...	2016-0...	Major	To Do	test bug B1	test bug B	c004d8cc-6...	http://ga-jira...	[com.ta...	http://ga-j...	changed
3	TPC-1	Test...	Bug	2015-...	2016-0...	Major	To Do	test bug C1	test bug C	c004d8cc-6...	http://ga-jira...	[com.ta...	http://ga-j...	changed

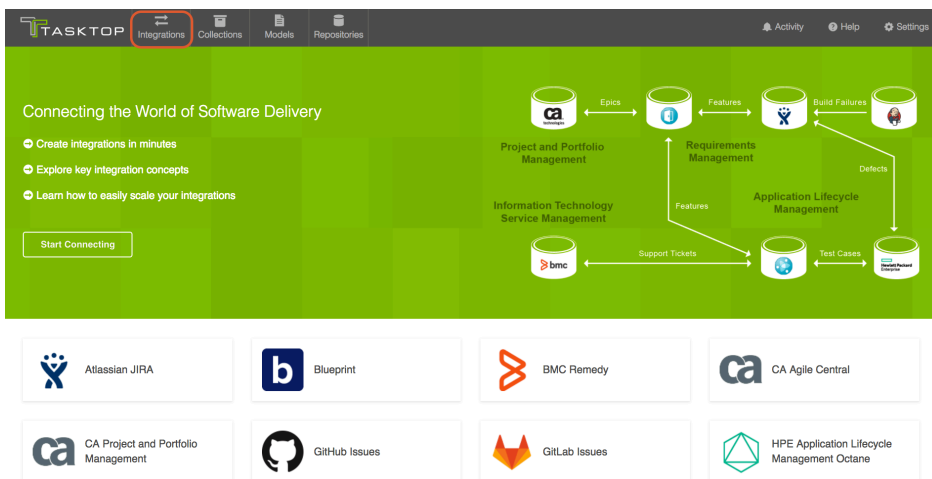
Project C is then removed from the source collection. At next full scan (one of the [polling intervals configured on the Settings page](#)), you'll see an event to denote that any artifacts in that collection have been removed:

id	formatted_id	project	type	created	modified	severity	status	summary	description	repository_id	repository_url	artifact_id	artifact_url	artifact_event_ty...
1	TPB-8	Test...	Bug	2016-...	2016-0...	Blocker	To Do	d33269d5e...	desc	c004d8cc-6...	http://ga-jira...	[com.ta...	http://ga-j...	changed
2	TPB-1	Test...	Bug	2015-...	2016-0...	Major	To Do	test bug B1	test bug B	c004d8cc-6...	http://ga-jira...	[com.ta...	http://ga-j...	changed
3	TPC-1	Test...	Bug	2015-...	2016-0...	Major	To Do	test bug C1	test bug C	c004d8cc-6...	http://ga-jira...	[com.ta...	http://ga-j...	changed
4	TPC-1	Test...	Bug	2015-...	2016-0...	Major	To Do	test bug C1	test bug C	c004d8cc-6...	http://ga-jira...	[com.ta...	http://ga-j...	removed

⚠ Note: If the project is added back to the collection and routed, records will not instantly be written out for all artifacts in that project; this will happen only when those artifacts change again.

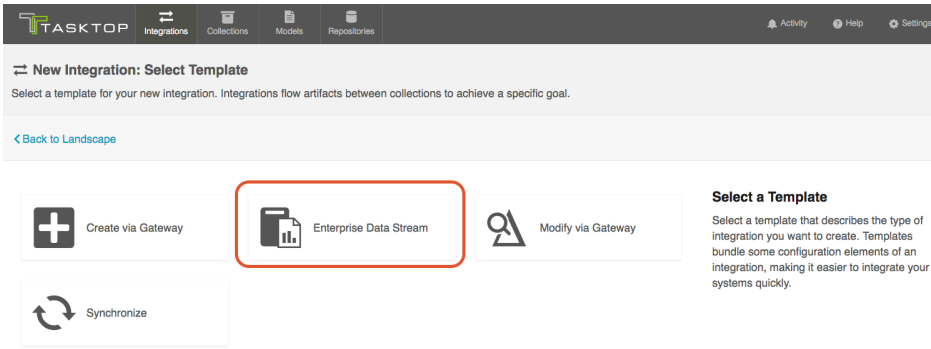
How to Configure

To configure your integration, select 'Integrations' at the top of the screen, then click 'New Integration.'

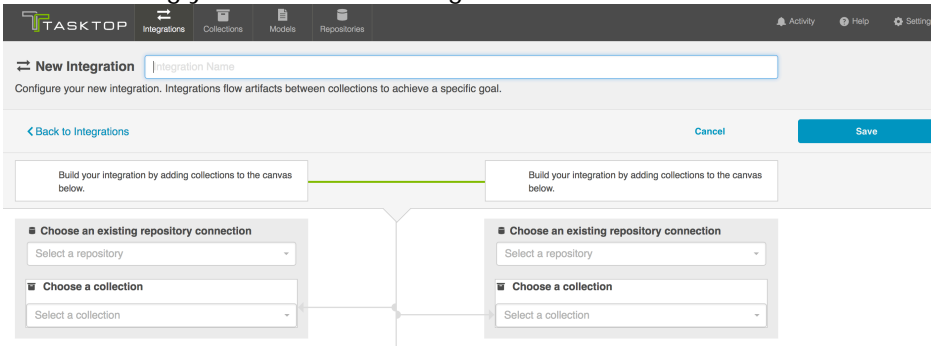


Select the 'Enterprise Data Stream' template.

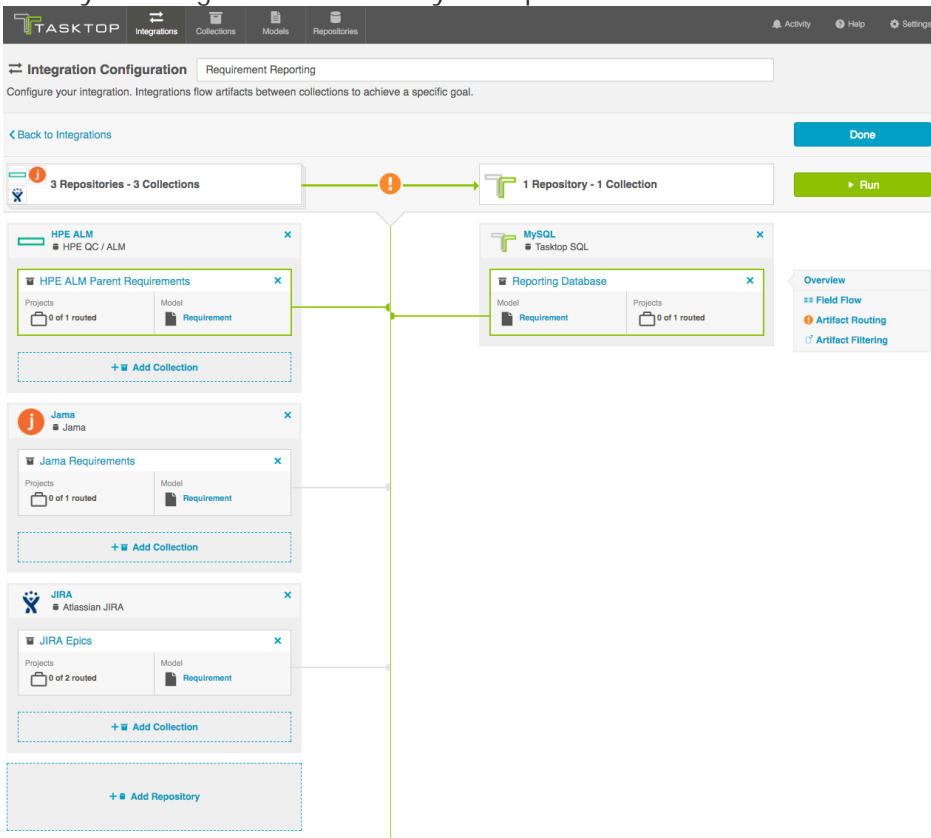
💡 Depending on the [edition](#) of Tasktop you are utilizing, you may not have all options available.



This will bring you to the New Integration Screen:

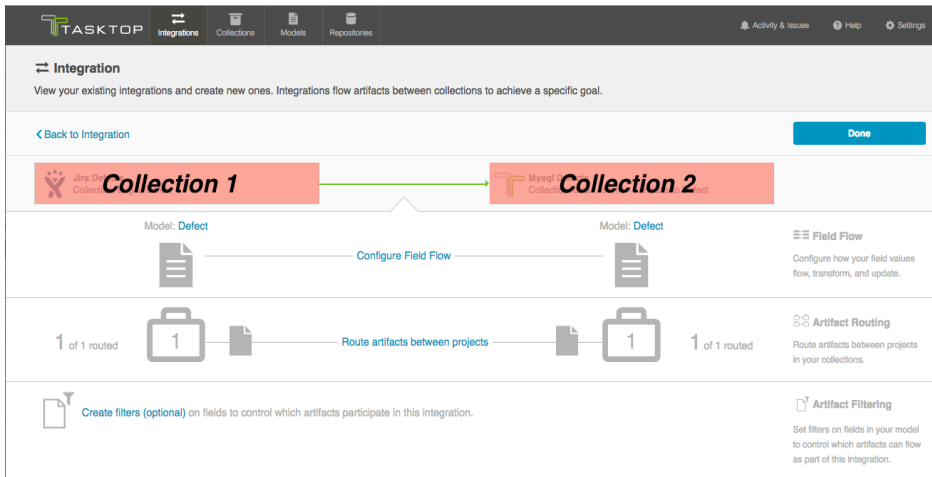
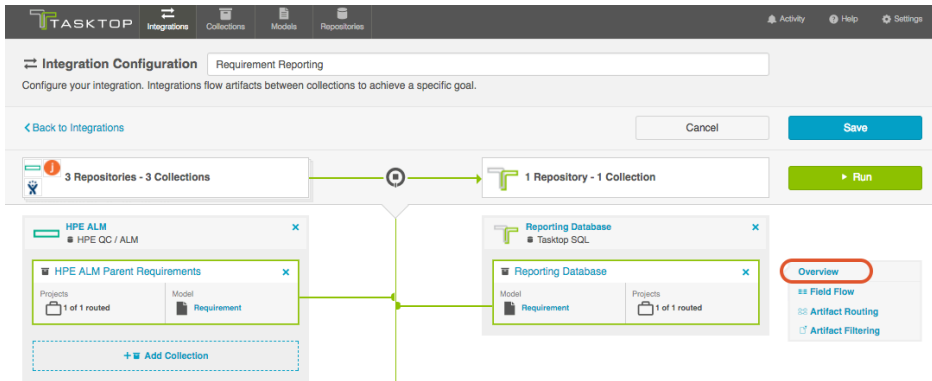


Name your integration and select your repositories and collections:



You can click the 'Overview' link on the right side of the Integration Page to get to the main display page (shown in the second screen shot).

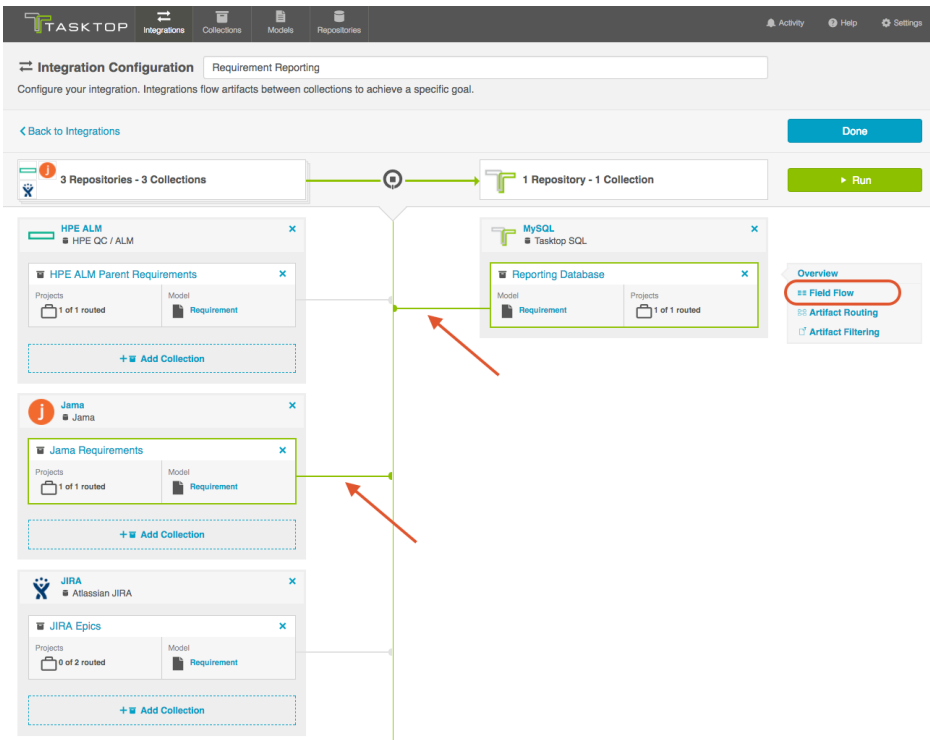
Note: The Overview page will only show two repositories at a time - one source repository and one target repository. If there are multiple source repositories in your integration, click on the one you are interested in before clicking 'Overview.'



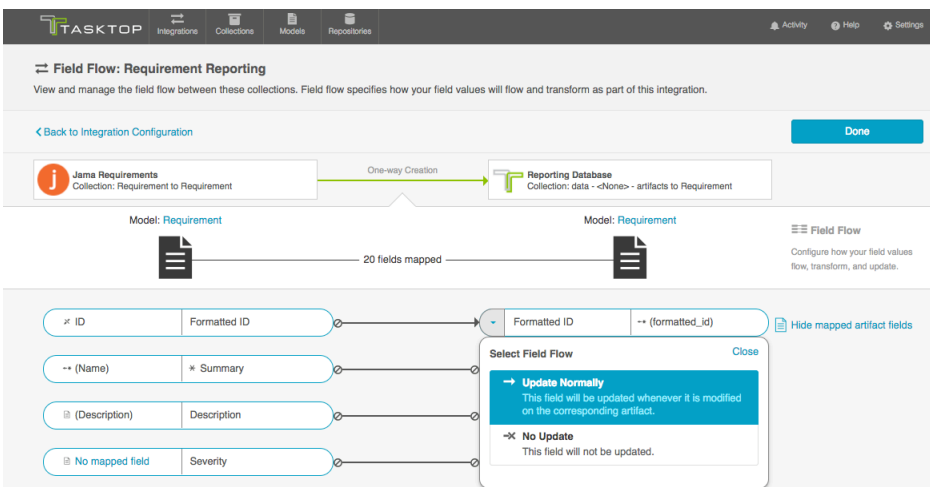
Field Flow

The field flow configured for a given integration specifies which fields should flow in that integration. For Enterprise Data Stream integrations, you can choose to flow a given field (Update Normally) or to not flow a given field (No Update).

To view field flow, select the two repositories you are interested in (you will see them highlighted in green once selected), and then click 'Field Flow'



You will be directed to the Field Flow screen:



You can choose to flow a field ('update normally') or not flow it ('no update'). You'll notice that field flow goes in one direction only - from the repository or gateway collection *into* the database collection.

You can see the names of the mapped artifact fields for each collection on the far left and far right, with the model fields displayed in the middle. To hide the mapped artifact fields, select 'Hide mapped artifact fields' on the right.

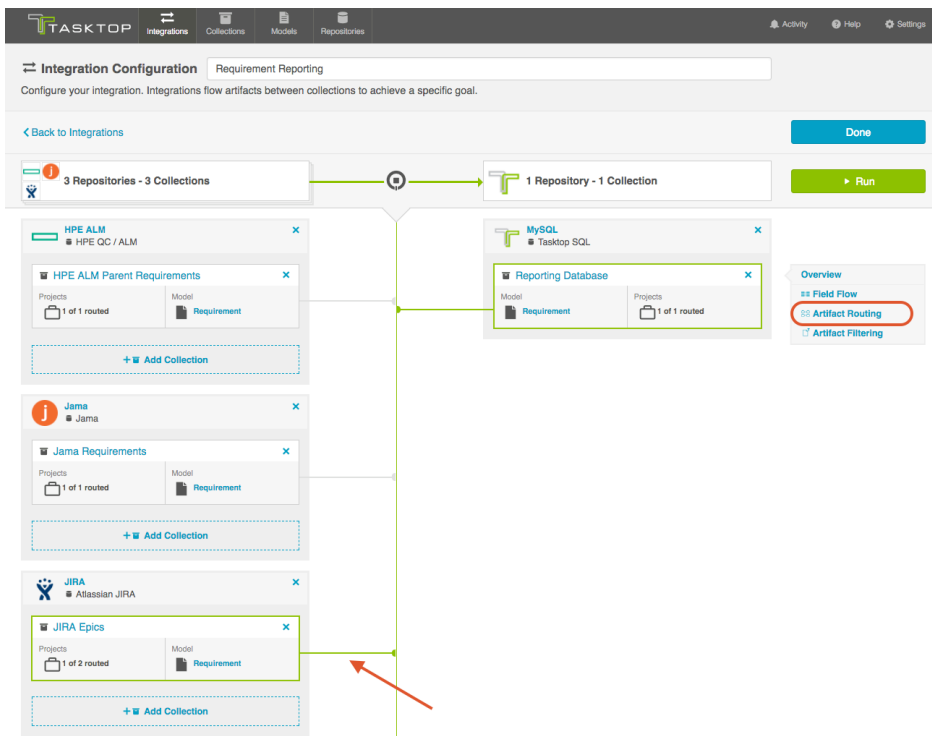
⚠ Note: The field flow settings behave a bit differently for Constant Values. This is because constant values exist as part of your Tasktop configuration, and not on the artifact itself. Therefore, changes in constant values are not detected in the same way that updates made on the actual artifact are detected. If you change the constant value that is

linked to your model, your integration will not automatically detect this update and sync it over. The value will only update if another field on that artifact is updated.

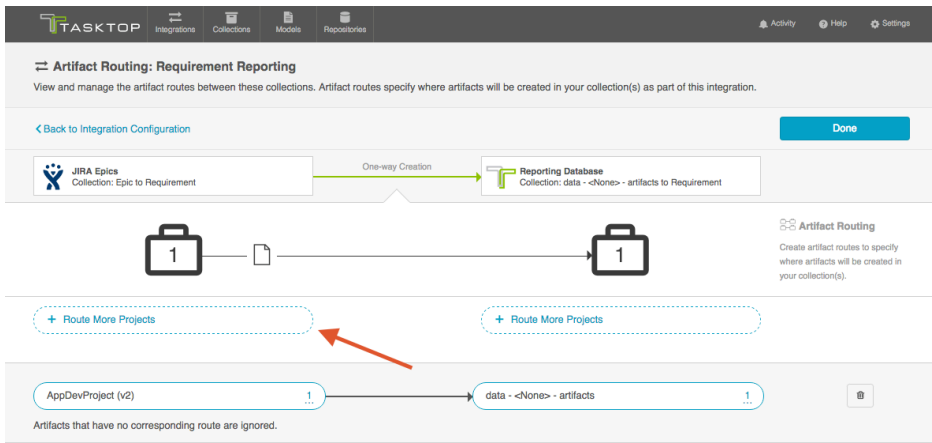
Artifact Routing

For an Enterprise Data Stream Integration, Artifact Routing is used to specify which projects (or other containers) you would like to participate in your integration. For example, your JIRA Epics collection may contain 10 different projects which are utilized in various integrations. However, for the purpose of your Enterprise Data Stream Integration, you may want only one of those projects to participate. You can specify that project on the Artifact Routing Screen.

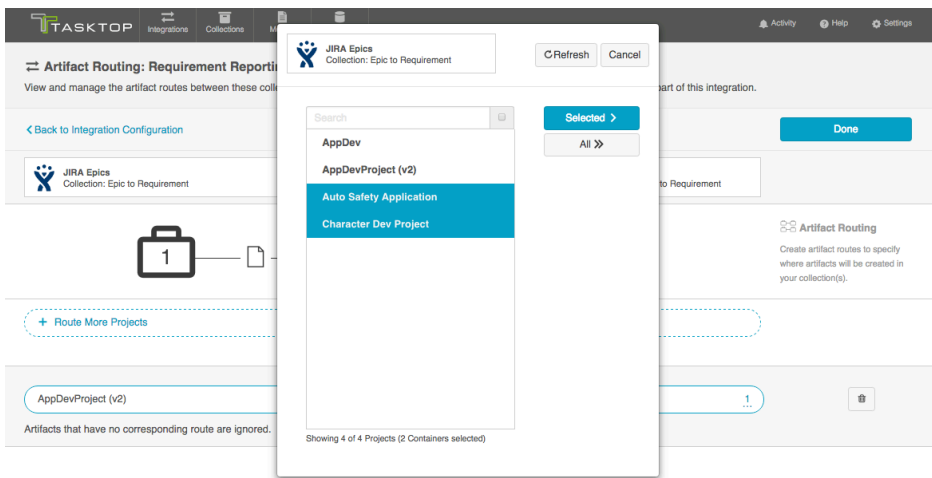
To configure Artifact Routing, select the relevant repositories and then click 'Artifact Routing':



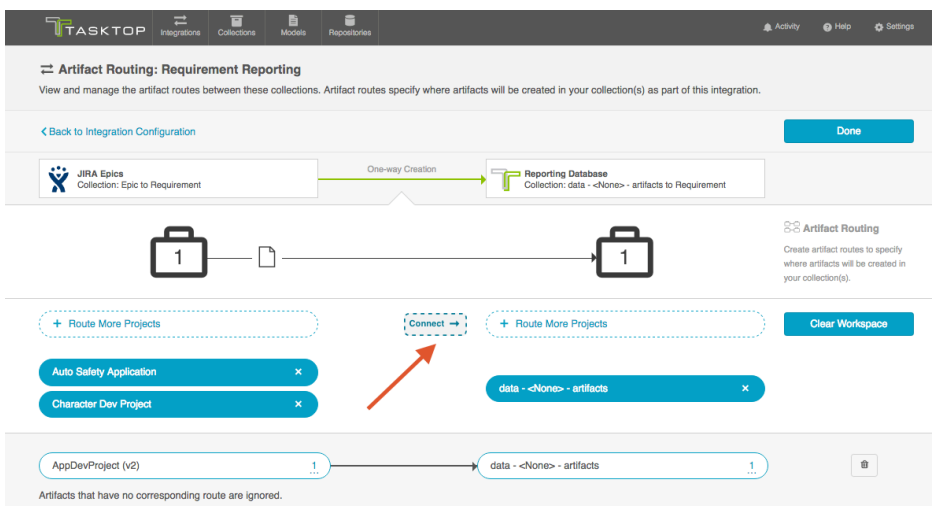
This will bring you to the Artifact Routing screen. You can click 'Route More Projects' to add additional projects to your route:



Select the projects you would like to participate in the integration and click 'Selected'



Click 'Connect'



You will see your artifact route on the pane below. Click 'Save' and 'Done.'

Artifact Routing: Requirement Reporting
View and manage the artifact routes between these collections. Artifact routes specify where artifacts will be created in your collection(s) as part of this integration.

[Back to Integration Configuration](#) Cancel Save

JIRA Epics
Collection: Epic to Requirement

One-way Creation

Reporting Database
Collection: data - <None> - artifacts to Requirement

Artifact Routing
Create artifact routes to specify where artifacts will be created in your collection(s).

+ Route More Projects

AppDevProject (v2)	1	data - <None> - artifacts	1
Auto Safety Application : Character Dev Project	2	data - <None> - artifacts	1

Artifacts that have no corresponding route are ignored.

Artifact Filtering

When configuring your integration, you have several options available to refine which artifacts are eligible to flow. The final mechanism available is *artifact filtering*, which is configured at the Integration level. Artifact Filtering allows you to filter which artifacts flow in your integration, based on a field value on that artifact.

To use a field for artifact filtering, it must:

- Be a part of your model, and be mapped to the collection you are filtering from
- Be one of the following field types:
 - Single Select
 - Note that in cases where 'allow unmapped values to flow' is enabled in the model, only fields that are already a part of the model will be considered for artifact filtering
 - Multi-Select
 - Note that in cases where 'allow unmapped values to flow' is enabled in the model, only fields that are already a part of the model will be considered for artifact filtering
 - Date
 - Date/Time
 - Duration
 - String

💡 Note that you can utilize our transforms to filter based on an 'unsupported' collection field type, if that field is mapped to a supported field type in your model. For example, you could filter based on a Boolean field in your repository, if that boolean field is mapped to a single select field in your model.

Unique Behavior for Enterprise Data Stream

The filtering behavior is somewhat unique when using the Enterprise Data Stream Template:

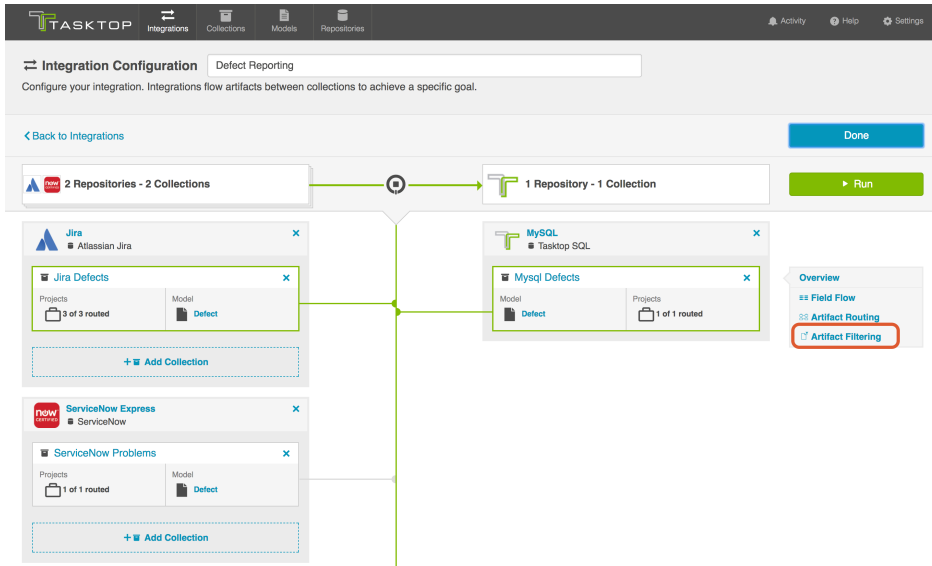
Though setting filters is meant to limit which artifacts flow in an integration, the impacts of setting filters on an Enterprise Data Stream Template are somewhat unique. Because it would not be ideal to have records in your database output that represent artifacts that have been filtered in an integration, given that these records would be stale and would not denote why a given artifact was not changing over time, it is the case that artifacts that are filtered on an Enterprise Data Stream Integration will still have records written out to the database but will have the "filtered" event type denoted.

Note the following:

- When you set a filter on an Enterprise Data Stream integration, records will not automatically be written out for artifacts that do not meet filtering criteria. When artifacts that should be filtered out change, we'll then write out a record with the "filtered" event type.
- When a once filtered artifact field changes such that it now meets the filter criteria set, records will be written out right away.
- If you relax the filter and more artifacts are now in scope, the now in scope artifacts will only flow when the artifacts themselves change again.
- If an artifact is filtered out of the Enterprise Data Stream Integration, and then its project is removed from the collection, records will be written out for all artifacts in that collection at next full scan and marked as "removed", whether or not they have been filtered out of the integration (This effectively means that the "removed" designation supersedes "filtered" designation.)
 - If you add the project back to the collection and routed in the integration, changes to artifacts will create a new record with either the "changed" or "filtered" event type, depending on whether or not the artifact meets the filter criteria.

How to Configure Artifact Filtering

To configure *Artifact Filtering*, select the relevant repository, then click 'Artifact Filtering' from the right pane of the Integration Configuration screen.

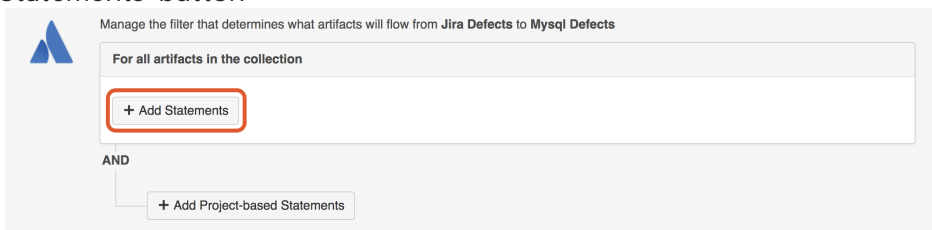


This will lead you to the Artifact Filtering Configuration screen, where you can configure your artifact filtering statement(s).

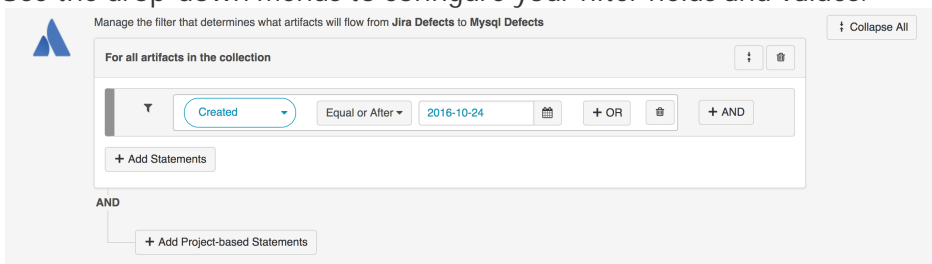
You can either add a statement that will apply to all artifacts in your collection, or to all artifacts within certain projects of your collection.

Apply Filter to All Artifacts in Collection

To apply a filter to all artifacts in the collection, simply click the '+Add Statements' button

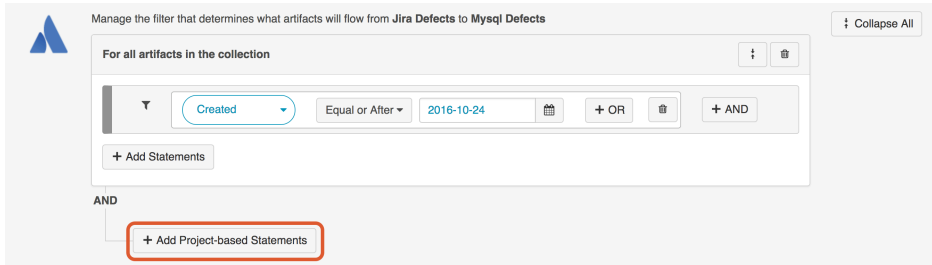


Use the drop-down menus to configure your filter fields and values:



Apply Filter to Artifacts in Certain Projects

To apply a filter to artifacts within a specific project, click the '+Add Project-based Statements' button.

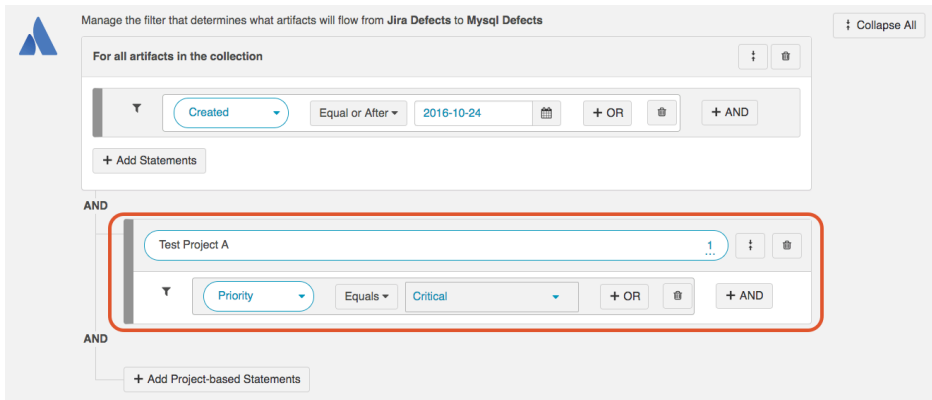


Click '+Add Projects' to select your project.



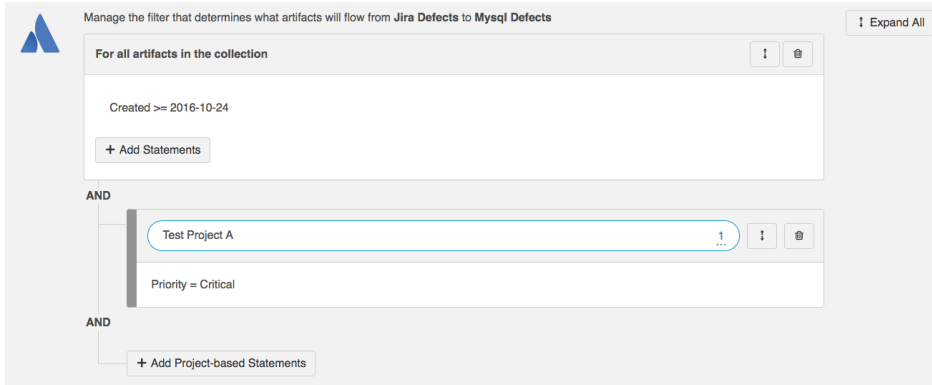
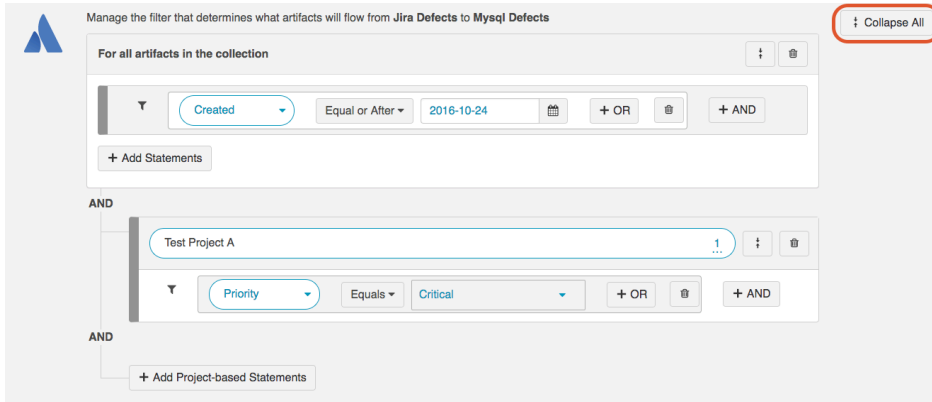
Select the project(s) you'd like your filter to apply to.

Then click 'Select Field...' to begin configuring your filtering statement.



Viewing Artifact Filter Statements

You can click the 'Collapse All' button to view an easier-to-read version of your artifact filtering statements.



Filtering via Repository Queries

In rare cases, you may find that the best option to restrict the artifacts eligible to flow is by setting a query within the repository itself.

⚠ Repository Queries are advanced functionality, and should only be used when you are truly unable to filter as desired using the built-in Tasktop functionality of Repositories, Collections, and Artifact Filtering.

If you plan to utilize repository queries, check the box next to 'Enable collections to be refined by setting a repository query,' on the [Repository Connection](#) screen.

Repository Connection
View and configure your repository connection.

[Back to Integration Configuration](#) **Done**

Jama

Label

Location

Authentication

Username

Password

Proxy Server **Use proxy server**

Proxy Host Address

Username

Password

Additional Settings

Repository Query **Enable collections to be refined by setting a repository query**

Concurrency Limit

Once this is selected, you will be able to select a repository query at the [Collection level](#) for any collections utilizing this repository.

Collection Configuration
Configure your collection. Collections comprise a group of artifacts that can be used in one or more integrations.

[Back to Collections](#) **Done**

Jama
Jama
<http://pdt-jama188.van.tasktop.com:8080/contour>

Projects
1 of 29 projects
[Manage Projects](#)

Field Mapping
Specify how the fields on your repository artifact align to fields in your model. The mapping between the fields on each side forms the field mapping for this collection.

Artifact: Requirement **Model: Requirement**

Map Fields

13 of 51 fields mapped | 13 of 15 fields mapped

Relationship Specification
Specify how your repository artifact aligns to the relationship types in your model.

Inbound Artifacts **Relating Model**

2 of 27 relationships mapped | 2 of 5 relationships mapped

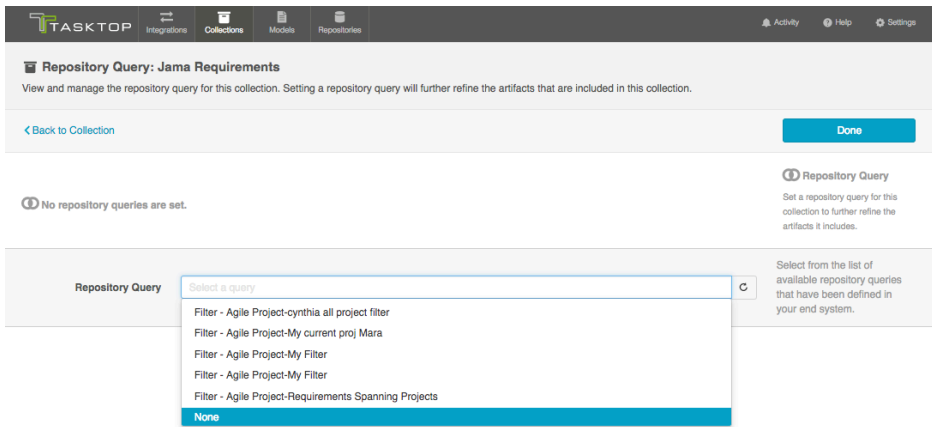
Person Resolution Strategy
Specify the strategy you'd like to use to reconcile persons between your repository artifact and your model.

Inbound Person **Person Model**

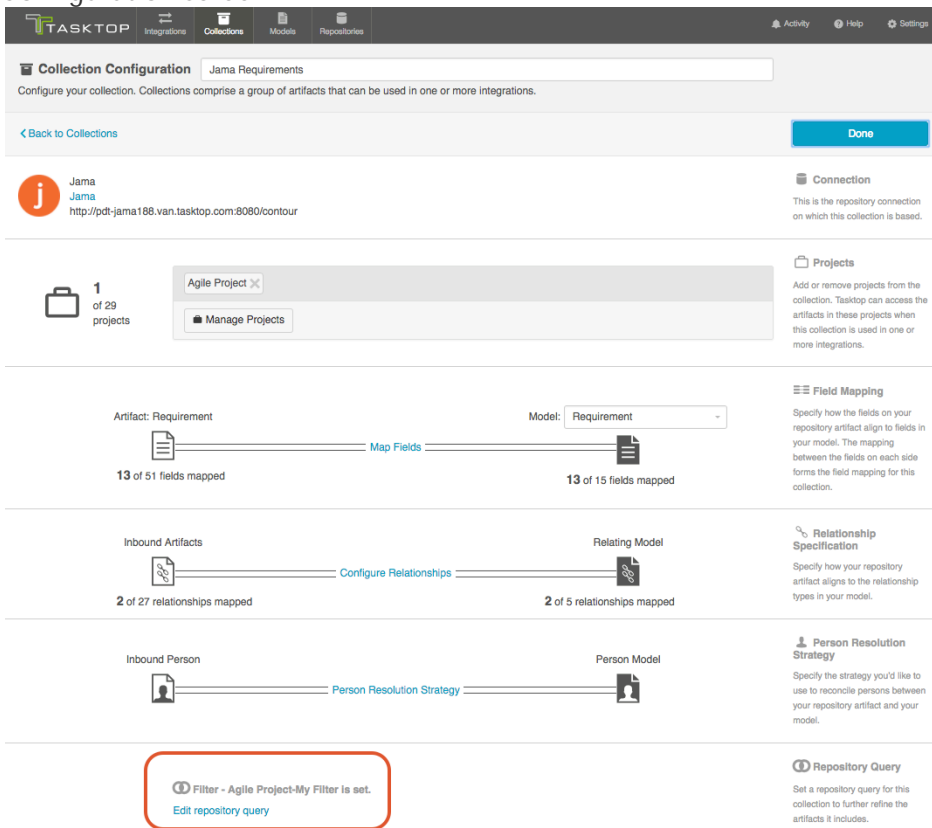
Repository Query
Set a repository query for this collection to further refine the artifacts it includes.

No repository queries are set.
[Set repository query](#)

On the drill-in page, you'll see a list of available repository queries. Select the query you'd like to use, and click 'Save,' and then 'Done.'



You will then see the selected repository query on the Collection Configuration screen:



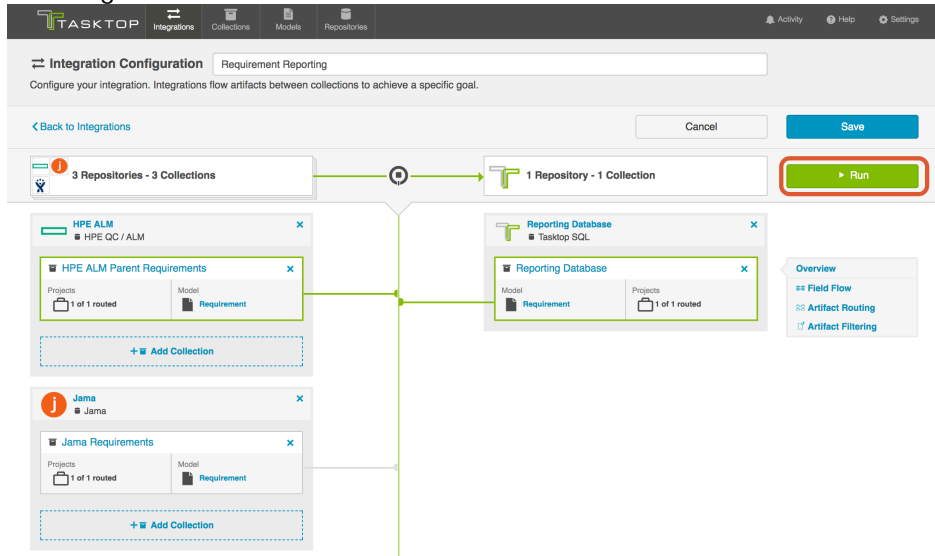
💡 Remember, applying a repository query to a collection will only further refine the artifacts that are already included in that collection. If you select a query that encompasses artifacts in projects not in your collection, these artifacts will not be added to the collection unless you also add those projects to your collection as you normally would.

Running your Integration

There are two ways to start or stop your integration:

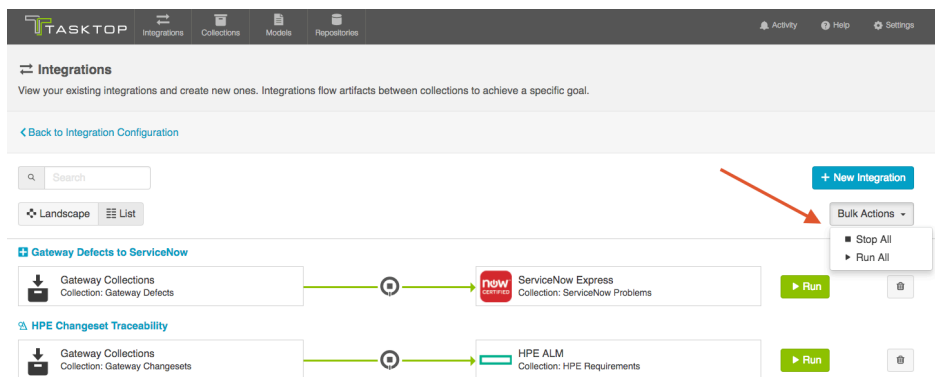
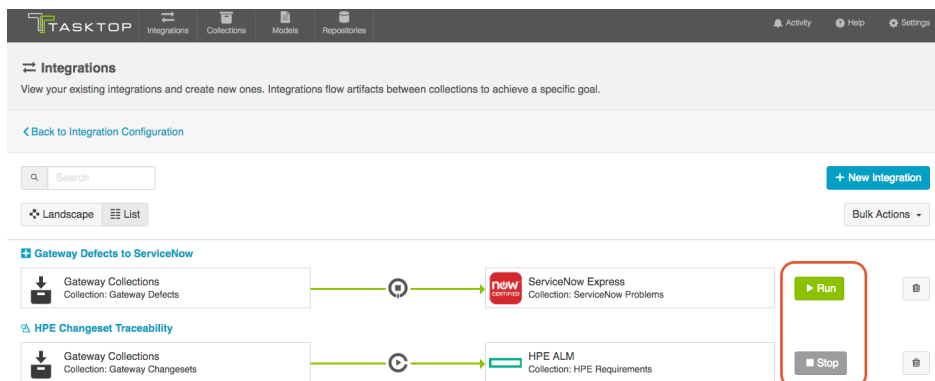
From the Integration Configuration Screen

Simply click the 'Run' to run the integration, and the 'Stop' button to stop the integration.



From the Integrations List Page

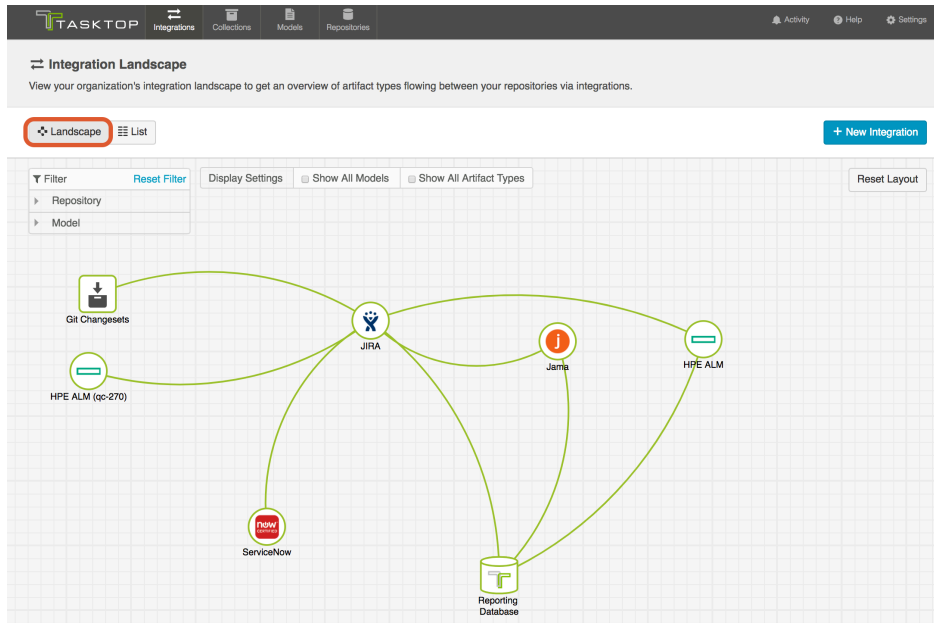
Click 'Run' or 'Stop' next to each integration you would like to update. You can also use the 'Bulk Actions' button to run or stop all integrations.



Viewing Your Integrations

See [Tasktop Editions table](#) to determine if your edition contains Integration Landscape View functionality.

When viewing your integrations, you have the option of viewing them in either Landscape or List mode.




Integration Title	Source Collection	Target Collection	Action
1. Help Desk Escalation	ServiceNow Collection: ServiceNow Problems	JIRA Collection: JIRA Defects from SNOW	Run
2. Requirements (JIRA to HPE ALM)	JIRA Collection: JIRA Epics	HPE ALM Collection: HPE ALM Parent Requirements	Run
3. JIRA Stories to HP Requirements	JIRA Collection: JIRA Stories	HPE ALM Collection: HPE ALM Requirements	Run
3. Requirements- Jama to JIRA	Jama Collection: Jama Requirements	JIRA Collection: JIRA Epics	Run
4. Defects (HPE ALM to JIRA)	HPE ALM Collection: HPE ALM Defects	JIRA Collection: JIRA Defects	Run
5. JIRA Changeset Traceability	Gateway Collections Collection: Git Changesets	JIRA Collection: JIRA Stories	Run
7. Orasi (HPE Defects to JIRA)	JIRA Collection: JIRA Defects (Orasi)	HPE ALM (qc-270) Collection: HPE Defects (Orasi)	Run
7. Orasi (JIRA Stories to HPE ALM Requirements)	JIRA Collection: JIRA Stories (Orasi)	HPE ALM (qc-270) Collection: HPE Requirements (Orasi)	Run
Requirement Reporting	3 Repositories 3 Collections	Reporting Database Collection: Reporting Database	Run

Landscape View

See [Tasktop Editions table](#) to determine if your edition contains Integration

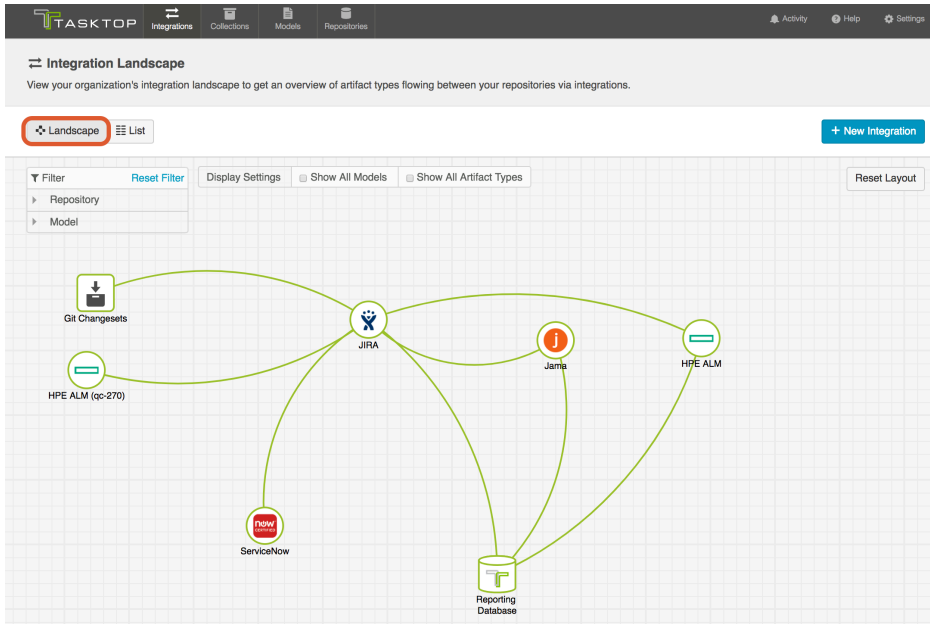
Landscape View functionality.

Learn more about the Integration Landscape View in the video below:

 Unknown macro: 'html'

Tasktop will default to the Landscape View, which enables you to visualize your entire integration landscape and see how your integrations relate to one another. Use our built-in filters to see as little or as much information as you'd like!

Here's a simplified view:

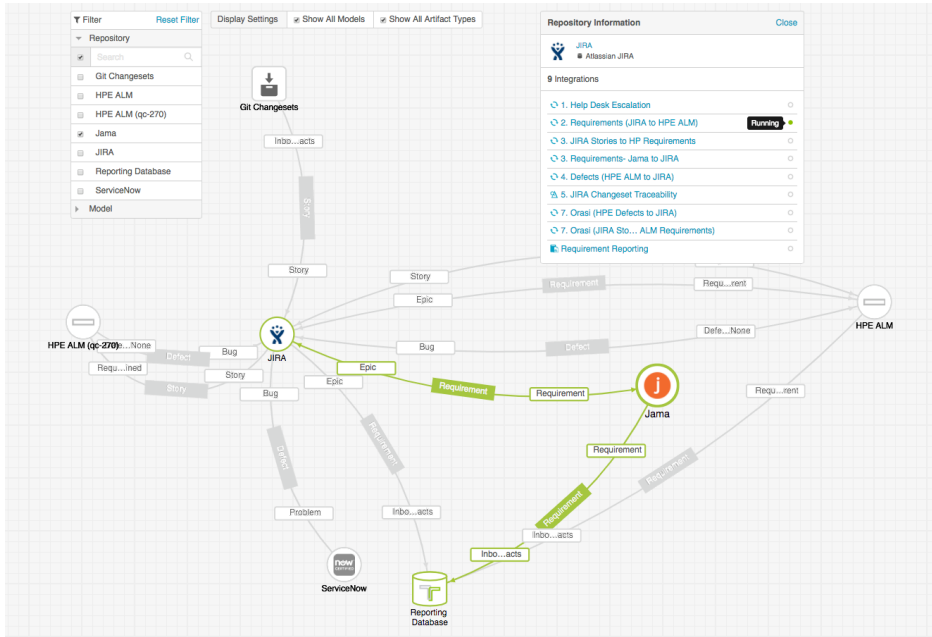


If you'd like to see additional information, you can utilize the filters, or click on a repository node to modify which information is shown.

Some examples of additional information you can see are:

- Models
- Artifact Types
- Artifact Creation Directionality Arrows
- List of all relevant integrations (see this by clicking on the repository node)
 - Indicator of whether each integration is running or not

Here's an example of a more detailed view:



List View

If you'd like, you can toggle to List View, which will show you a list of all integrations you have created.

You can use this view to:

- Start an Integration
- Stop an Integration
- Delete an Integration
- Click into an Integration and modify its configuration

The screenshot shows the Tasktop Integrations page. At the top, there are navigation tabs for Integrations, Collections, Models, and Repositories. Below the header, there's a search bar and a '+ New Integration' button. A 'List' button is circled in red. The main content area displays a list of integrations, each with a title, source and target collections, a 'Run' button, and a trash icon. The integrations are as follows:

- 1. Help Desk Escalation:** ServiceNow (Collection: ServiceNow Problems) to JIRA (Collection: JIRA Defects from SNOW).
- 2. Requirements (JIRA to HPE ALM):** JIRA (Collection: JIRA Epics) to HPE ALM (Collection: HPE ALM Parent Requirements).
- 3. JIRA Stories to HP Requirements:** JIRA (Collection: JIRA Stories) to HPE ALM (Collection: HPE ALM Requirements).
- 3. Requirements- Jama to JIRA:** Jama (Collection: Jama Requirements) to JIRA (Collection: JIRA Epics).
- 4. Defects (HPE ALM to JIRA):** HPE ALM (Collection: HPE ALM Defects) to JIRA (Collection: JIRA Defects).
- 5. JIRA Changeset Traceability:** Gateway Collections (Collection: Git Changesets) to JIRA (Collection: JIRA Stories).
- 7. Orasi (HPE Defects to JIRA):** JIRA (Collection: JIRA Defects (Orasi)) to HPE ALM (Collection: HPE Defects (Orasi)).
- 7. Orasi (JIRA Stories to HPE ALM Requirements):** JIRA (Collection: JIRA Stories (Orasi)) to HPE ALM (Collection: HPE Requirements (Orasi)).
- Requirement Reporting:** 3 Repositories (3 Collections) to Reporting Database (Collection: Reporting Database).

Reporting

To ETL or Not To ETL?

ETL (Extract, Transform, Load) is a process where data is extracted from a database, transformed to be more suitable for reporting or analytics, and loaded into a database which is normally used for reporting.

The data structures populated directly by Tasktop are intended to be used as a source for ETL; Some kinds of reports are not easily produced without first performing an ETL process. ETL can also be beneficial for performance of reports.

Some reports are possible without first performing an ETL process. Examples of such reports include Artifact Cycle Time and Defect Count By State By Cycle Time.

Example Reports

Following are examples of some reports that can be driven directly from the database tables populated by an Enterprise Data Stream Integration:

Artifact Cycle Time

Artifact Cycle Time is often a valuable metric to measure as it can help

identify areas where efficiencies can be gained and ensure “lean flow”. We have provided a model called “Artifact Cycle Time” and can be used to easily flow the necessary data to your database – enabling you to create a variety of metrics and visualizations based on the cycle time of any artifact type.

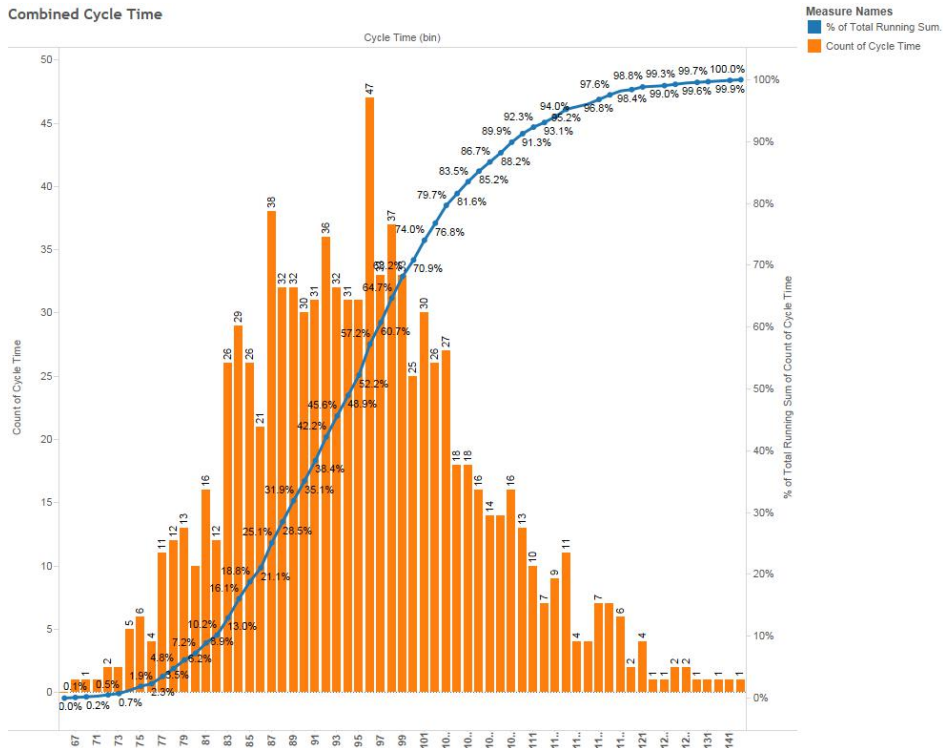
Artifact Cycle Time Model

Artifact Cycle Time
Formatted ID
Project
Type
Created
Modified
Severity
Status
Priority
Release
Assignee

If you use this model, you can easily produce visualizations such as a histogram that can identify the historical trend of cycle times.

Artifact Cycle Time Histogram

Combined Cycle Time



SQL

```

SELECT A.FORMATTED_ID, B.MODIFIED AS
StatusOpen, C.MODIFIED AS StatusInProgress,
D.MODIFIED AS StatusReadyForTesting,
E.MODIFIED AS StatusReadyForVerification,
F.MODIFIED AS StatusComplete, G.MODIFIED AS
StatusShipped, A.STATUS AS CurrentStatus FROM
ARTIFACT A
  LEFT OUTER JOIN ARTIFACT B
    ON B.ARTIFACT_ID = A.ARTIFACT_ID
    AND B.STATUS = 'Open'
    AND NOT EXISTS (SELECT * FROM ARTIFACT
WHERE ARTIFACT_ID = A.ARTIFACT_ID AND
(MODIFIED < B.MODIFIED OR (MODIFIED =
B.MODIFIED AND ID < B.ID)) AND STATUS =
B.STATUS)
  LEFT OUTER JOIN ARTIFACT C
    ON C.ARTIFACT_ID = A.ARTIFACT_ID
    AND C.STATUS = 'In Progress'
    AND NOT EXISTS (SELECT * FROM ARTIFACT
WHERE ARTIFACT_ID = A.ARTIFACT_ID AND
(MODIFIED < C.MODIFIED OR (MODIFIED =
C.MODIFIED AND ID < C.ID)) AND STATUS =
C.STATUS)
  LEFT OUTER JOIN ARTIFACT D
    ON D.ARTIFACT_ID = A.ARTIFACT_ID
    AND D.STATUS = 'Ready for Testing'
    AND D.MODIFIED > (SELECT MAX(MODIFIED)
FROM ARTIFACT WHERE ARTIFACT_ID =

```

```

A.ARTIFACT_ID AND STATUS IN ('Open', 'In
Progress'))
    AND NOT EXISTS (SELECT * FROM ARTIFACT
WHERE ARTIFACT_ID = A.ARTIFACT_ID AND
(MODIFIED < D.MODIFIED OR (MODIFIED =
D.MODIFIED AND ID < D.ID)) AND STATUS =
D.STATUS
    AND MODIFIED > (SELECT MAX(MODIFIED)
FROM ARTIFACT WHERE ARTIFACT_ID =
A.ARTIFACT_ID AND STATUS IN ('Open', 'In
Progress'))))
LEFT OUTER JOIN ARTIFACT E
ON E.ARTIFACT_ID = A.ARTIFACT_ID
AND E.STATUS = 'Ready for Verification'
AND E.MODIFIED > (SELECT MAX(MODIFIED)
FROM ARTIFACT WHERE ARTIFACT_ID =
A.ARTIFACT_ID AND STATUS IN ('Open', 'In
Progress', 'Ready for Testing'))
AND NOT EXISTS (SELECT * FROM ARTIFACT
WHERE ARTIFACT_ID = A.ARTIFACT_ID AND
(MODIFIED < E.MODIFIED OR (MODIFIED =
E.MODIFIED AND ID < E.ID)) AND STATUS =
E.STATUS
    AND MODIFIED > (SELECT MAX(MODIFIED)
FROM ARTIFACT WHERE ARTIFACT_ID =
A.ARTIFACT_ID AND STATUS IN ('Open', 'In
Progress', 'Ready for Testing'))))
LEFT OUTER JOIN ARTIFACT F
ON F.ARTIFACT_ID = A.ARTIFACT_ID
AND F.STATUS = 'Complete'
AND F.MODIFIED > (SELECT MAX(MODIFIED)
FROM ARTIFACT WHERE ARTIFACT_ID =
A.ARTIFACT_ID AND STATUS IN ('Open', 'Ready
for Testing', 'Ready for Verification'))
AND NOT EXISTS (SELECT * FROM ARTIFACT
WHERE ARTIFACT_ID = A.ARTIFACT_ID AND
(MODIFIED < F.MODIFIED OR (MODIFIED =
F.MODIFIED AND ID < F.ID)) AND STATUS =
F.STATUS
    AND MODIFIED > (SELECT MAX(MODIFIED)
FROM ARTIFACT WHERE ARTIFACT_ID =
A.ARTIFACT_ID AND STATUS IN ('Open', 'Ready
for Testing', 'Ready for Verification'))))
LEFT OUTER JOIN ARTIFACT G
ON G.ARTIFACT_ID = A.ARTIFACT_ID
AND G.STATUS = 'Shipped'
AND G.MODIFIED > (SELECT MAX(MODIFIED)
FROM ARTIFACT WHERE ARTIFACT_ID =
A.ARTIFACT_ID AND STATUS IN ('Open', 'Ready
for Testing', 'Ready for Verification',
'Complete'))

```



```
AND NOT EXISTS (SELECT * FROM ARTIFACT
WHERE ARTIFACT_ID = A.ARTIFACT_ID AND
(MODIFIED < G.MODIFIED OR (MODIFIED =
G.MODIFIED AND ID < G.ID)) AND STATUS =
G.STATUS
AND MODIFIED > (SELECT MAX(MODIFIED)
FROM ARTIFACT WHERE ARTIFACT_ID =
A.ARTIFACT_ID AND STATUS IN ('Open', 'Ready
for Testing', 'Ready for Verification',
'Complete'))))
WHERE NOT EXISTS (SELECT * FROM ARTIFACT
WHERE ARTIFACT_ID = A.ARTIFACT_ID AND
(MODIFIED > A.MODIFIED OR (MODIFIED =
A.MODIFIED AND ID > A.ID)))
```

```

AND (A.ARTIFACT_EVENT_TYPE IS NULL OR NOT
A.ARTIFACT_EVENT_TYPE = 'removed' )
ORDER BY A.FORMATTED_ID

```

The example above is designed to handle cases where an artifact is moved into a state more than once. For example, a defect that is moved to “Complete”, subsequently moved back into “In Progress”, then moved to “Complete” again is represented with a row having the second timestamp for the “Complete” status.

Formatted_id	priority	type	severity	repository_url	StatusOpen	StatusInProgress	StatusReadyForTesting	StatusReadyForVerification	StatusComplete	StatusShipped	CurrentStatus
TKK2-Test-Project1	Low	Defect	3	HP ALM	2015-01-20 03:40:00	2015-01-24 23:10:00	2015-02-02 00:52:00	2015-04-27 03:09:00	2015-05-14 04:50:00	2015-05-23 06:37:00	Shipped
TKK2-Test-Project10	Medium	Defect	2	HP ALM	2015-02-05 01:22:00	2015-02-12 17:41:00	2015-02-24 22:13:00	2015-03-17 01:12:00	2015-03-30 17:58:00	2015-04-05 17:46:00	Shipped
TKK2-Test-Project11	High	Defect	1	HP ALM	2015-02-09 09:49:00	2015-02-13 10:00:00	2015-02-26 15:28:00	2015-03-09 08:46:00	2015-03-18 11:00:00	2015-03-27 15:58:00	Shipped
TKK2-Test-Project12	Low	Defect	3	JIRA	2015-02-06 14:09:00	2015-02-10 12:49:00	2015-02-27 12:26:00	2015-03-17 14:08:00	2015-03-29 11:39:00	2015-04-06 14:08:00	Shipped
TKK2-Test-Project13	Medium	Defect	2	HP ALM	2015-02-07 19:44:00	2015-02-11 21:33:00	2015-03-13 02:41:00	2015-04-15 19:54:00	2015-05-08 23:00:00	2015-05-17 18:06:00	Shipped
TKK2-Test-Project14	High	Defect	1	HP ALM	2015-02-07 18:43:00	2015-02-10 22:00:00	2015-02-18 22:23:00	2015-02-25 02:19:00	2015-03-07 01:31:00	2015-03-17 18:47:00	Shipped
TKK2-Test-Project15	High	Defect	1	HP ALM	2015-02-12 19:09:00	2015-03-05 18:22:00	2015-04-02 20:03:00	2015-05-01 18:26:00	2015-05-12 14:00:00	2015-05-20 19:48:00	Shipped
TKK2-Test-Project16	High	Defect	1	HP ALM	2015-02-15 19:00:00	2015-02-19 20:42:00	2015-03-14 20:20:00	2015-03-21 22:42:00	2015-04-04 17:16:00	2015-04-04 22:52:00	Shipped

Reports can be driven from the results of this SQL query, subtracting dates to produce cycle times for the desired transitions (e.g. “Open” to “Shipped”).

Status values in the SQL above correspond to the values present in the “Artifact” model; repository-specific status values can be mapped to the model values in the corresponding Collection mapping. If status values are added, removed or changed in the Artifact model, then the SQL will have to be modified accordingly.

Defect Count By State By Cycle Time

Defect Count By State By Cycle Time provides a count of defects by cycle time for each status of an artifact.

In this example, the cycle time is measured in days. Cycle time is only measured for status state transitions; Cycle time is not measured for the end state of an artifact.

We provide a basic defect model packaged with our product:

Basic Defect Model

Defect Model
Formatted ID
Project
Type
Created
Modified
Severity
Status
Summary

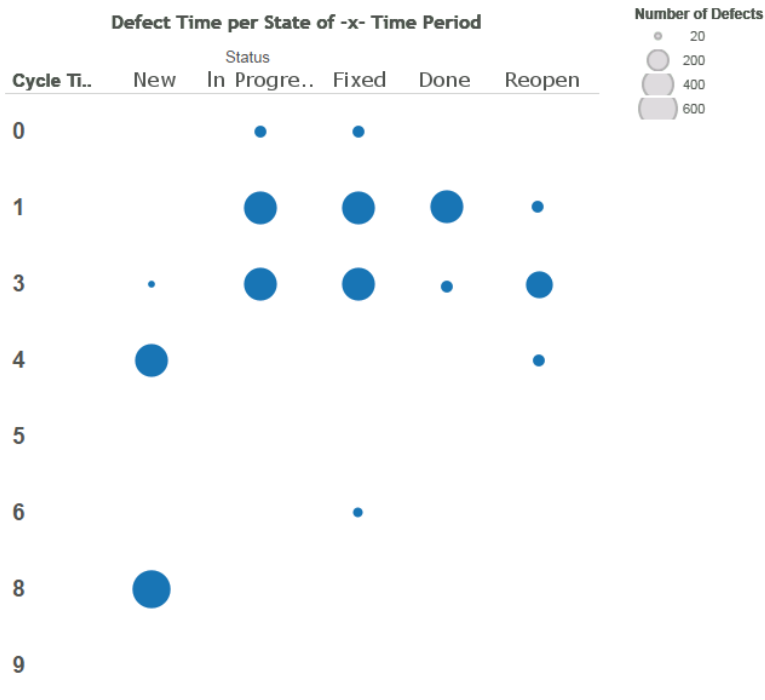
Summary-to-Description

Related Defects

Description

If you use this model, you can easily produce visualizations such as a bubble chart that can identify the volume of defects in each cycle time measured in days. This is simply a slightly different view into your overall cycle time.

Cycle Time Volume



SQL

```

SELECT status, COUNT(artifact_id), cycleTime
FROM (
    SELECT A.ARTIFACT_ID AS artifact_id,
    A.STATUS AS status, SUM(
    TIMESTAMPTDIFF(SQL_TSI_DAY,A.MODIFIED,B.MODIFIED) ) AS cycleTime FROM DEFECT A
    INNER JOIN DEFECT B ON A.ARTIFACT_ID =
    B.ARTIFACT_ID
    AND A.ID != B.ID
    AND A.STATUS != B.STATUS
    AND A.MODIFIED <= B.MODIFIED
    AND ((A.ARTIFACT_EVENT_TYPE IS NULL OR
    B.ARTIFACT_EVENT_TYPE IS NULL)
    OR NOT (A.ARTIFACT_EVENT_TYPE =
    'removed' OR B.ARTIFACT_EVENT_TYPE =
    'removed'))
    )
    WHERE NOT EXISTS (
    SELECT * FROM DEFECT C WHERE
    C.ARTIFACT_ID = A.ARTIFACT_ID AND C.ID !=
    A.ID AND C.ID != B.ID
    AND C.MODIFIED >= A.MODIFIED AND
    C.MODIFIED <= B.MODIFIED
    AND ((C.STATUS = A.STATUS OR C.STATUS
    = B.STATUS) OR (C.STATUS != A.STATUS AND
    C.STATUS != B.STATUS))
    )
    AND NOT EXISTS (
    SELECT * FROM DEFECT D WHERE
    D.ARTIFACT_ID = A.ARTIFACT_ID AND B.MODIFIED
    <= (
    SELECT MAX(MODIFIED) FROM DEFECT D
    WHERE D.ARTIFACT_ID = A.ARTIFACT_ID AND
    D.ARTIFACT_EVENT_TYPE = 'removed'
    )
    )
    GROUP BY A.ARTIFACT_ID, A.STATUS
) CT GROUP BY CT.status, CT.cycleTime
ORDER BY CT.status, CT.cycleTime

```

Step 5: Expand or Modify your Integration

Tasktop: 17.4 Release

Expanding the Scale of Your Integration

You've already configured your integration, and it's running great! Now

- Expanding the Scale of Your Integration
 - Adding Projects
 - Adding or Editing Fields

you'd like to increase the scale by adding additional projects from each of your repositories to your integration landscape, or by adding additional fields to your mapping. No problem - you can make these updates in just a few clicks!

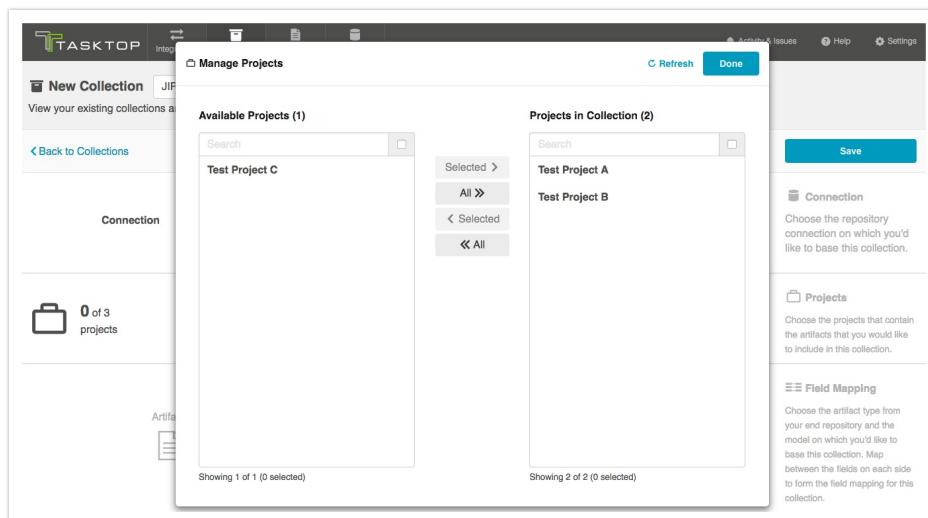
Below, we've included some tips and tricks on how to effectively scale your integration, as well as information on what to expect when you make modifications to your integration configuration after the integration has been activated.

Adding Projects

In order to add additional projects from one or more of your repositories to your integration landscape, simply navigate to each collection, and add additional projects as desired. Once that's saved, navigate to the integration, click on 'Artifact Routing' and route the projects appropriately - either creating new routes or adding to existing routes.

Once the new projects have been added and routed, Tasktop will detect the artifacts contained within the new project(s) at the change detection interval (configured on the Settings page) and flow data according to the configuration that you have already set.

On the Collection Configuration Screen:



On the Artifact Routing Screen (in the Integrations section):

Add Projects to New Routes:

ARTIFACT ROUTING: Smartbear - JIRA REQUIREMENTS

View and manage the artifact routes between these collections. Artifact routes specify where artifacts will be created in your collection(s) as part of this integration.

← Back to Integration Configuration Cancel Save

Smartbear Requirements Collection: Requirement to Requirement ↔ JIRA Requirements Collection: Story to Requirement

Artifact Routing
Create artifact routes to specify where artifacts will be created in your collection(s).

+ Route More Projects Clear Workspace

TCK Project

Sample Project ↔ Test Project C

Sample Project ↔ Test Project C

Artifacts that have no corresponding route are ignored.

ARTIFACT ROUTING: Smartbear - JIRA REQUIREMENTS

View and manage the artifact routes between these collections. Artifact routes specify where artifacts will be created in your collection(s) as part of this integration.

← Back to Integration Configuration Cancel Save

Smartbear Requirements Collection: Requirement to Requirement ↔ JIRA Requirements Collection: Story to Requirement

Artifact Routing
Create artifact routes to specify where artifacts will be created in your collection(s).

+ Route More Projects Clear Workspace

TCK Project

Sample Project ↔ Test Project C

Sample Project ↔ Test Project C

Artifacts that have no corresponding route are ignored.

JIRA Requirements
Collection: Story to Requirement

Search

- Test Project A
- Test Project B
- Test Project C

Showing 3 of 3 Projects (1 Containers selected)

C Refresh Cancel

Add Selected Add All

ARTIFACT ROUTING: Smartbear - JIRA REQUIREMENTS

View and manage the artifact routes between these collections. Artifact routes specify where artifacts will be created in your collection(s) as part of this integration.

← Back to Integration Configuration Cancel Save

Smartbear Requirements Collection: Requirement to Requirement ↔ JIRA Requirements Collection: Story to Requirement

Artifact Routing
Create artifact routes to specify where artifacts will be created in your collection(s).

+ Route More Projects Clear Workspace

TCK Project Test Project B

Sample Project ↔ Test Project C

Sample Project ↔ Test Project C

Artifacts that have no corresponding route are ignored.

Sample Project	.1	←	Test Project C	.1	⋮
TCK Project	.1	←	Test Project B	.1	⋮
Sample Project	.1	→	Test Project C	.1	⋮
TCK Project	.1	→	Test Project B	.1	⋮

Artifacts that have no corresponding route are ignored.

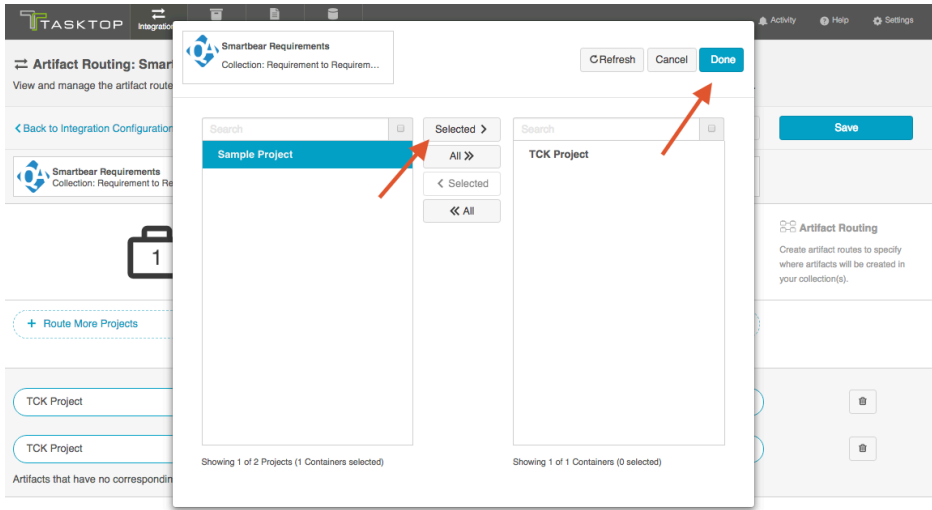
Add Projects to Existing Routes:

Click the numerical link on the right side of the pill to add additional projects to that route:

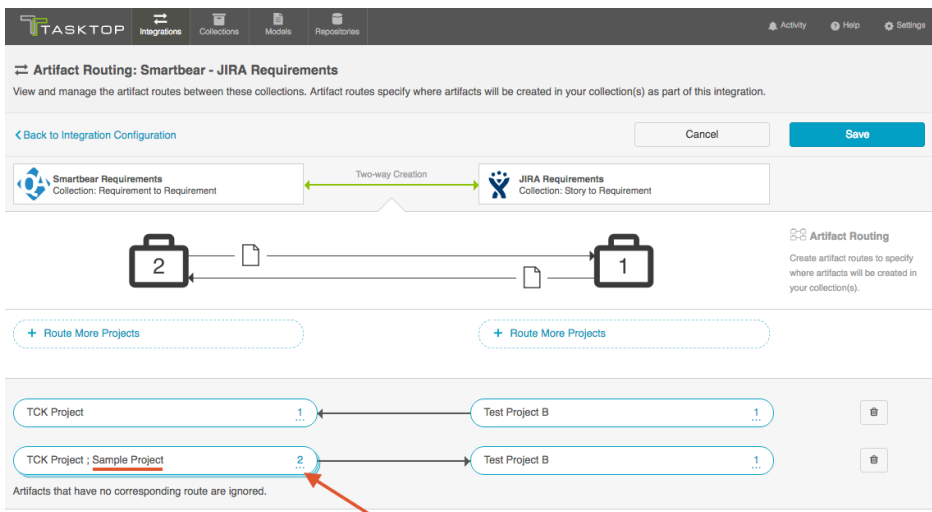
TCK Project	.1	←	Test Project B	.1	⋮
TCK Project	.1	→	Test Project B	.1	⋮

Artifacts that have no corresponding route are ignored.

Highlight the project you'd like to add, click 'Selected>' and then 'Done.'



You will now see the updated number of projects, and the additional project's name listed in the pill:



Note: Depending on how you set up your artifact routing, you may need to configure conditional artifact routing. This will be relevant if you route to more than one target project (as you will need to identify criteria by which the integration can determine which project to flow the artifact to). You can learn more about conditional artifact routing [here](#).

Adding or Editing Fields

If you'd like to add, remove, or edit a field in your model, Tasktop allows you to do so even after the Integration has begun to run. Once the field has been added to your model, navigate to your relevant collections and map that field as needed. You can then edit the field flow frequency from the Integration's field flow screen.

If you add a new field to your integration's field flow, the field will be synced automatically for newly created artifacts. Tasktop will detect these changes according to the change detection interval.

Warning: Note that if you add or edit a new field mapping on an integration

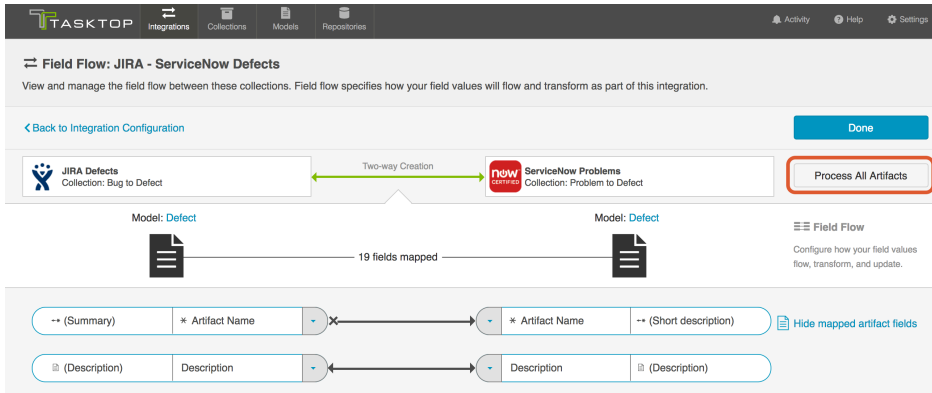
that has already begun running, Tasktop will not automatically apply those new field mappings to artifacts that had already been synced and that were created before that mapping had been added unless/until that field specifically changes on the artifacts. However, if you'd like to automatically sync the data in those fields for all artifacts, you can click the 'process all artifacts' button on the field flow page. This will push through all artifacts that have already been synced, and update any fields that are eligible based on your field flow configuration.

On the New Model Screen:

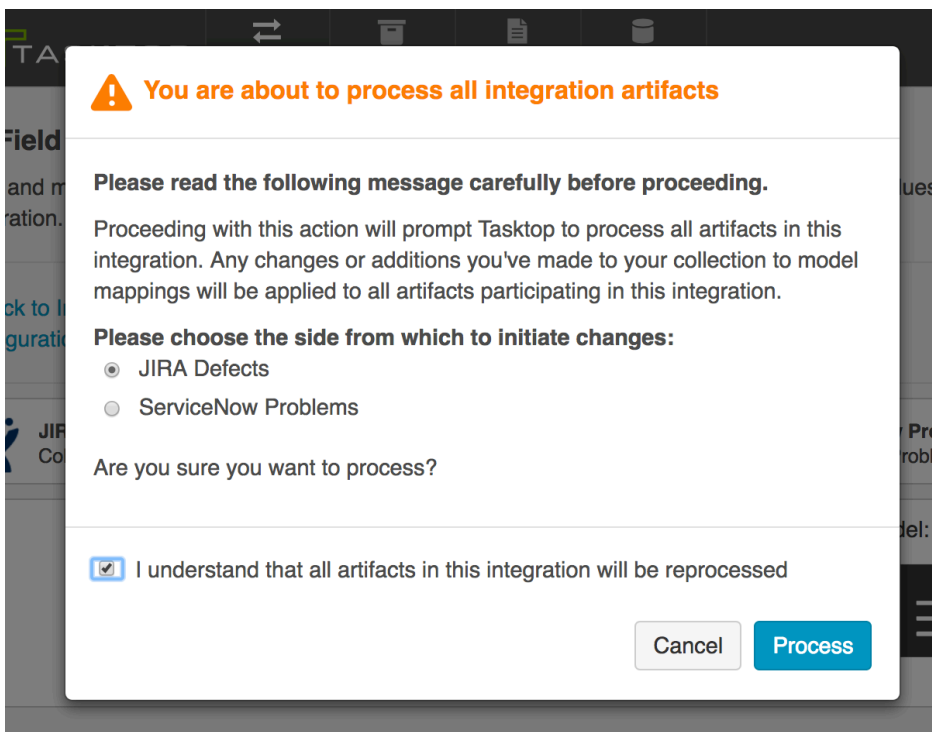
Smart Field	Label	Type	Required	
Summary	Summary	String	<input type="checkbox"/>	⊞ ↓
Description	Description	String	<input type="checkbox"/>	⊞ ↑

On the Collection Field Mapping Screen:

On the Integration Field Flow Screen



After clicking 'Process All Artifacts,' you will be prompted to choose the side from which to initiate changes:



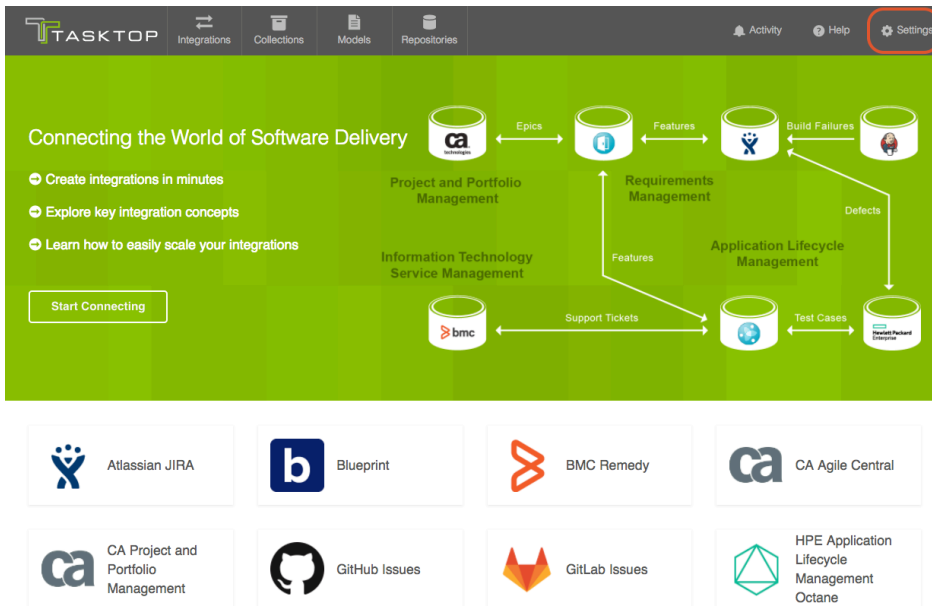
This will process all artifacts in the collection, and flow any eligible field updates to the target collection.

Settings

Tasktop: 17.4 Release

To access the 'Settings' page, click the Settings button in the upper right corner of your screen

- Polling Interval Configuration
- Logging
- Extensions
 - Custom Data Transformation
 - Payload Transformation
 - Person Reconciliation
 - State

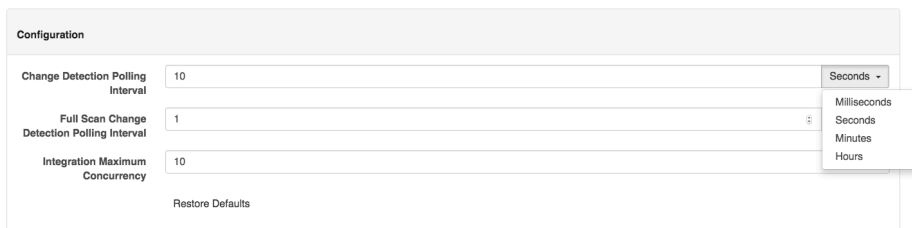


- Transition
- License
- Master Password Configuration
- Import Artifact Pair Information
- Storage Settings
 - Migrating from the Internal Database to an External Database
 - Migrating from an External Database to a Different External Database

From here, you'll be able to access things like polling interval configuration, extensions, license information, and more.

Polling Interval Configuration

The Configuration section allows the administrator to change the polling frequency of the connected repositories.



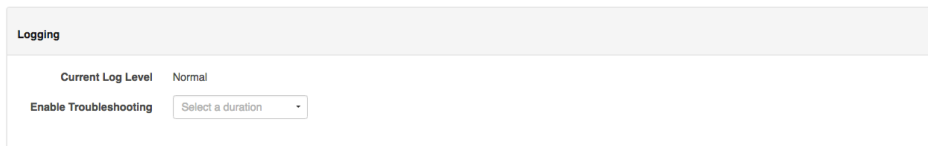
- **Change Detection Polling Interval:** The time between polling requests to detect *only changed artifacts*. This defaults to 1 minute, but can be customized as desired.
- **Full Scan Change Detection Polling Interval:** The time between polling requests to detect changed artifacts, in which *all* artifacts of a collection are scanned. This defaults to 10 hours, but can be customized as desired.
- **Integration Maximum Concurrency:** This limits the number of events processed concurrently by each integration. Increasing this value will enable more artifact changes to flow concurrently, whereas decreasing this value will reduce the level of concurrent changes. Changing this value has the potential to affect the load on the end-points of an integration, and may have an adverse effect on performance if set too high. The default setting (10) should be used unless advised to change by Tasktop Support.

Logging

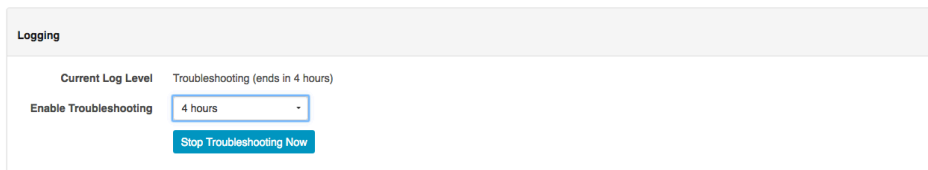
Tasktop logs various events that the application performs. These can be vital for troubleshooting purposes. There are two logging levels available. The first is "Normal" and is sufficient for most scenarios. In the event that more detailed logs are required, "Enable Troubleshooting" logging level is available. Due to the large volume of logs created during Troubleshooting logging, this option has a time limit with a maximum of 24 hours. If Troubleshooting level is selected, the Normal logging level can be enabled at any time by clicking the Stop Troubleshooting Now button.

Updating the logging levels immediately changes the logging granularity and Tasktop does not need to be restarted for the change to take effect.

Default Logging Enabled



Troubleshooting Logging Enabled



Downloading Logs

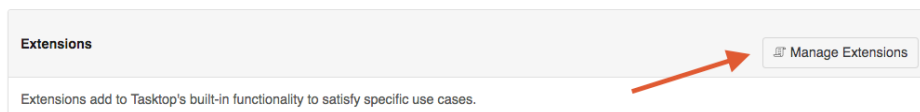
Please reference the [Troubleshooting](#) page for instructions on downloading the logs as part of the Error Report.

Extensions

Extensions add to Tasktop's basic functionality by facilitating processes such as custom data transformations, payload transformations, advanced person reconciliation, and state transitions.

You can create and save custom extensions for use in your integrations on the 'Settings' screen. To create and edit your extensions, click the 'Manage Extensions' button.

Below, you will find basic information about each extension type and how to configure it. You can also see example extensions, and learn technical implementation details in the [Extensions](#) section.



Extensions

View your existing extensions and create new ones. Extensions add to Tasktop's built-in functionality to satisfy specific use cases.

[← Back to Settings](#)

You haven't created any extensions.

[+ New Extension](#)

New Extension: Select Type

Select an extension type. Extensions add to Tasktop's functionality to satisfy specific use cases.

[← Back to Extensions](#)

[Done](#)



Custom Data Transformation



Payload Transformation



Person Reconciliation



State Transition

Select the extension type that describes the type of extension you need to create. Extensions add to Tasktop's functionality to satisfy specific use cases.

Custom Data Transformation

Custom Data Transformation Extensions enable you to map fields to one another which do not have out-of-the-box transforms. You can apply this extension when configuring mappings from the [Collection-to-Model mapping page](#).

Payload Transformation

Payload Transformation Extensions enable you to take the payload sent in by your Gateway Collection and transform it into a format that Tasktop can accept. Once you have saved your extension, you can select it on the [Gateway Collection screen](#).

Person Reconciliation

Person Reconciliation Extensions enable you to match 'person' fields from one repository to another. You can select the extension on the [Person Reconciliation screen](#) during the Collection configuration process.

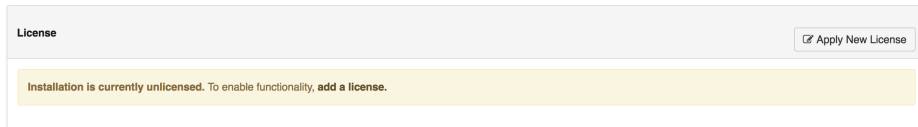
State Transition

State Transition Extensions enable you to transition artifacts from one state to another according to a set workflow. The extension can be applied from the [State Transition Sash](#) on the Collection Configuration screen.

License

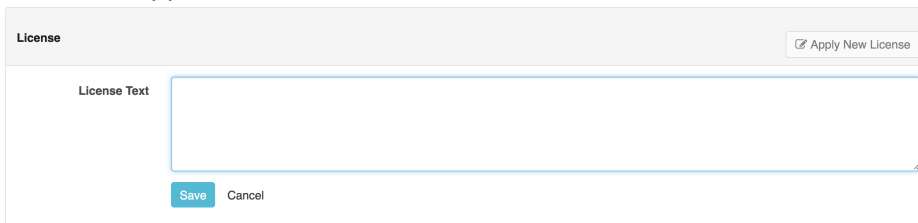
A license is required to run the application. Upon initial log-in, you will

see that your product is currently un-licensed:



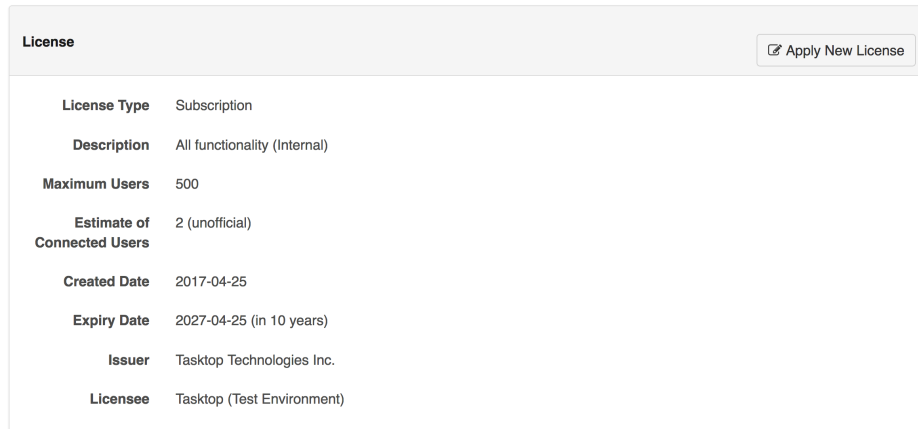
Click 'Apply New License' to enter your license.

The Master Password must be set and the License must be entered before the application can be used.

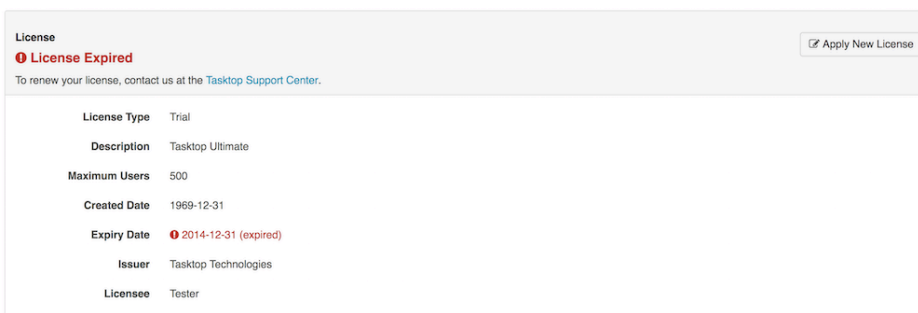


On the license panel you can see:

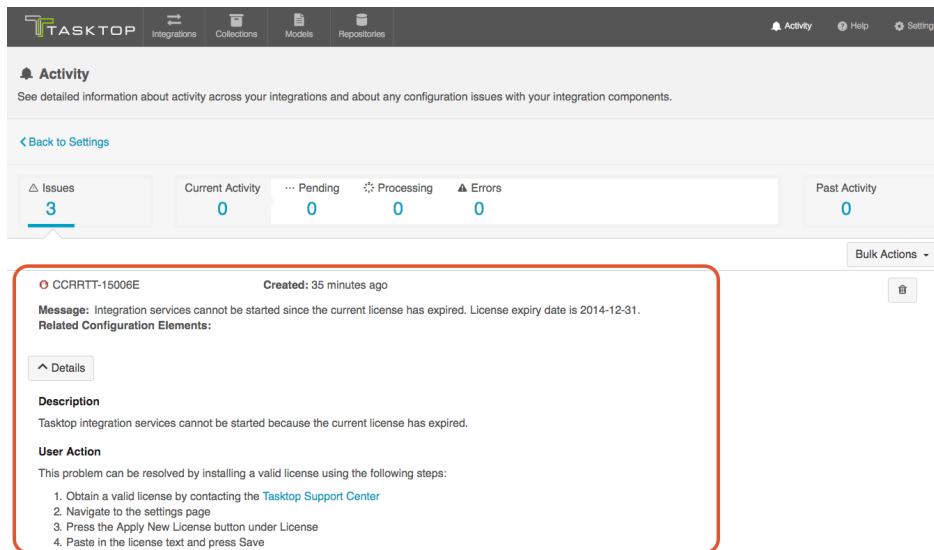
- License Type
- Description
- Maximum Users
- Estimate of Connected Users
- Created Date
- Expiration Date
- Issuer
- Licensee



You will also see a warning if your license is expired:



Should your license expire, in addition to seeing a warning on the Settings page, you'll also see that an issue is surfaced on the Activity screen:

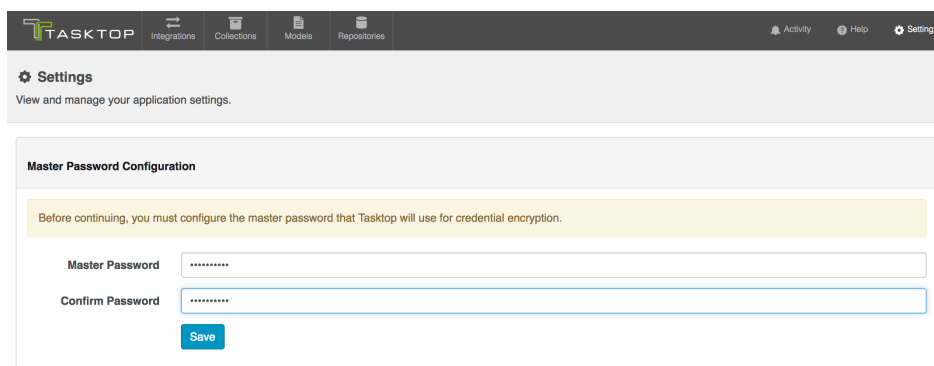


When your license is expired, you'll still be able to navigate within the Tasktop UI, but your integrations will be stopped from running. Note that though they will still display the Run or Stopped state they were in at the time your license expired, no artifacts will process in an integration until a new license is applied.

🟡 Please consult your license agreement or contact your account representative if you have any questions about your license settings or user counting policy.

Master Password Configuration

After installation, you will be prompted to set a Master Password.



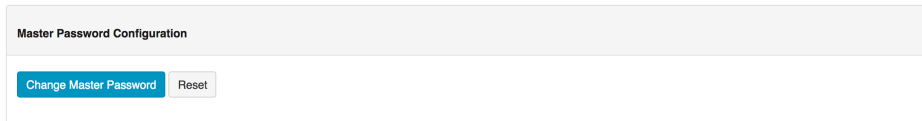
The Master Password is used to encrypt the credentials used in your repository connections and proxy settings. Tasktop Integration Hub will automatically use the stored Master Password to decrypt repository credentials.

Normally you will not need to re-enter your Master Password.

However, if the stored Master Password is missing, or if you'd like to change your Master Password from the Settings screen, you will need to enter your current Master Password.

The Master Password is encrypted and stored separately from the encrypted repository credentials. On Windows, the encrypted Master Password is stored in the Windows Registry, encrypted using the Windows Data Protection (DPAPI). On Linux, the encrypted Master Password is stored in the Home Directory of the User running Tasktop Integration Hub.

If desired, you can change or reset the Master Password from the 'Settings' page. In order to change the Master Password, you must enter your current Master Password. If resetting the Master Password, on the other hand, you will not need to enter your current Master Password, but previously encrypted repository passwords will be lost, and must be provided after resetting.



Import Artifact Pair Information

Importing artifact pairs allows Tasktop Integration Hub to know about existing artifact pairs that were created by Tasktop Sync. This prevents duplicate artifacts from being created when you switch from using Tasktop Sync to Tasktop Integration Hub to administer your integrations. Please contact Tasktop Support for additional information on how to use this capability.

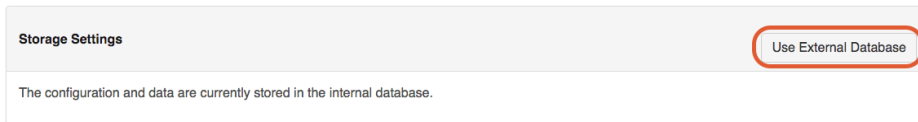
Storage Settings

Tasktop automatically stores operational data to a built-in database. However, for production environments, we strongly recommend that operational data is stored to an external database for improved maintainability. This will enable you to perform frequent back-ups without having to stop Tasktop Integration Hub, and ensure that your Tasktop Integration Hub practices are consistent with your existing disaster and recovery process.

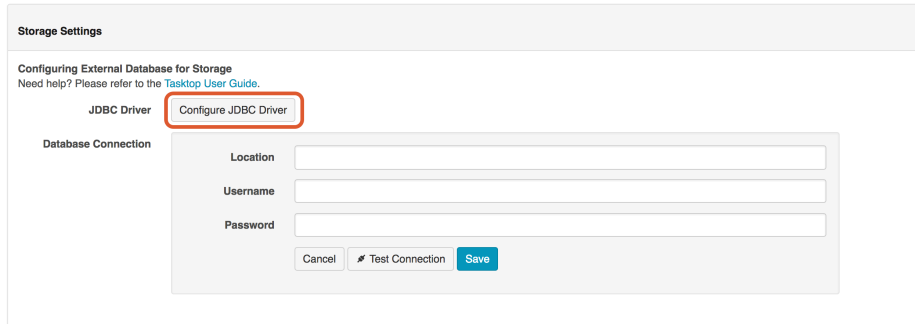
 Please see our [Installation & Hardware Requirements](#) section to see which databases are supported for storing operational data.

Migrating from the Internal Database to an External Database

To migrate your Tasktop operational data from the internal database to an external database, click the 'Use External Database' button.



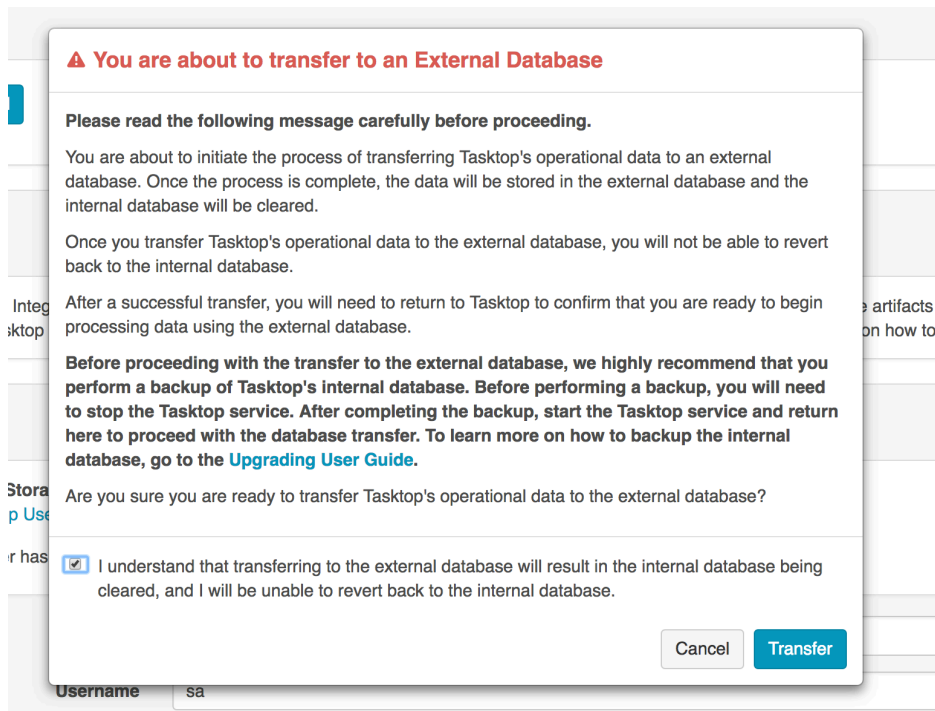
Next, click 'Configure JDBC Driver,' and upload a JDBC driver for your database.



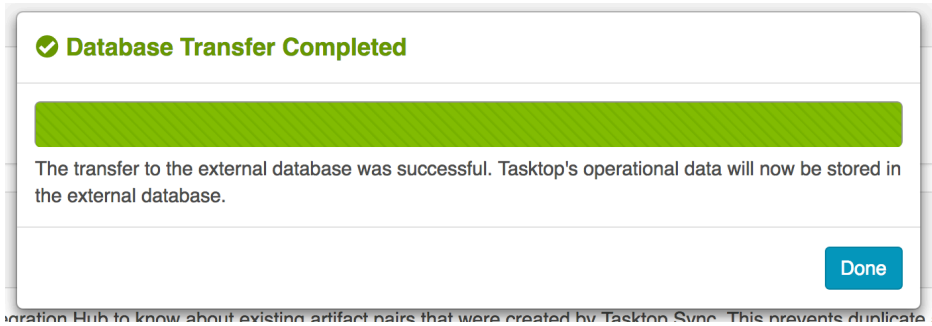
Once the JDBC driver is successfully uploaded, enter the location, username, and password for your database.

You can click 'Test Connection' to confirm that your credentials have been accepted by Tasktop. Once confirmed, click 'Save.'

You will see a warning message telling you that you are about to transfer to an External Database. Review the entire message, ensuring that you have performed the recommended data back-up, and if approved select 'I understand...' and then 'Transfer.'



You will get a 'Database Transfer Completed' message once the transfer is complete. You have now successfully transferred your operational data from Tasktop's internal database to your own external database.



Migrating from an External Database to a Different External Database

⚠ If you'd like to migrate your data from one external database to a different external database, please note that you will need to manually transfer the data from the current database to the new target database. If you do not manually transfer the data, Tasktop will not work properly once you switch to the target database settings. Tasktop will not automatically transfer this data for you.

However, if you are simply updating the location of your current external database and will continue using the same database, you do not need to transfer any data. Tasktop will continue to work properly.

Once you are ready to update the database information, simply update the Location, Username, and/or Password fields in the Database Connection setting, click 'Test Connection,' and then 'Save.' Read the warning message that pops up, ensuring you have taken all necessary steps, and then click 'I understand...' and 'Save.'

Upgrading

Tasktop: 17.4 Release

Before upgrading Tasktop, be sure to shut down Tasktop and afterwards [backup the internal database](#) (or if you are using an external database, follow the steps outlined below). The first time that Tasktop restarts after an update, the internal database will be migrated to the new version and it will no longer be possible to return to the prior version without the backup. Also make backups of the Tomcat and Catalina configuration files that have been customized. The upgrade process will overwrite these configuration files.

- [Linux](#)
- [Windows](#)
- [Recovering from an error during upgrade](#)

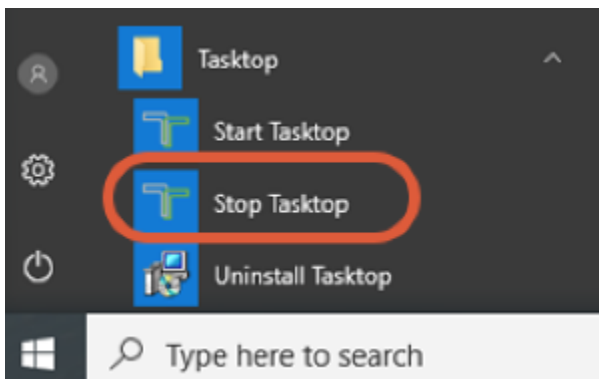
Linux

1. Ensure a copy of the old distribution archive is available in case a roll-back is required
2. Shutdown Tasktop and Keycloak
3. If you are using Keycloak's internal configuration database, back up the database (User's Tasktop data location/keycloak/standalone/data/keycloak.h2.db)

4. Move the old Tasktop installation to an archive folder
5. Unzip the new Tasktop distribution archive
6. If you are using Tasktop's internal configuration database, copy the `tasktop/db` folder from the old installation into the new installation folder `<install-location>/tasktop`
7. If you are using an external database for Tasktop's configuration, copy the `tasktop-db.json` file from the old installation into the new installation folder `<install-location>/tasktop`
8. Re-apply any customizations to the Tomcat and Catalina configuration
9. If you are using Keycloak's internal configuration database, restore the database (User's Tasktop data location/`keycloak/standalone/data/keycloak.h2.db`) after installation
10. If you are using an external database for Keycloak's configuration, reconfigure the external database as described in https://keycloak.gitbooks.io/documentation/server_installation/topics/database.html
11. If you have connected to the Microsoft TFS repository in the past, remove `<install-location>\Tasktop\libraries\microsoft-tfs` folder. Note that you will need to upload a new SDK file in the TFS repository settings once Tasktop is started back up.
12. Restart Tasktop
13. Open the errors page and resolve errors related to unsatisfied connector requirements

Windows

1. Ensure a copy of the old installer is available in case a roll-back is required
2. Click the 'Stop Tasktop' button on your desktop, and make sure services are stopped:



3. Shutdown Tasktop and Keycloak
4. If upgrading from 17.1.0 or earlier: Back up the Keycloak configuration database (User's Tasktop data location/`keycloak/standalone/data/keycloak.h2.db`)
5. If you are using Tasktop's internal configuration database, backup the database, located at `<ProgramData>\Tasktop\db`

6. If you are using an external database for Tasktop's configuration, perform the recommended back-up procedure associated with the database vendor.
7. Run the installer of the new version of Tasktop
8. Re-apply any customizations to the Tomcat and Catalina configuration
9. If upgrading from 17.1.0 or earlier: Restore the Keycloak configuration database (User's Tasktop data location/keycloak/standalone/data/keycloak.h2.db) after installation
10. If you are using an external database for Keycloak's configuration, reconfigure the external database as described in https://keycloak.gitbooks.io/documentation/server_installation/topics/database.html
11. If you have connected to the Microsoft TFS repository in the past, remove <ProgramData>\Tasktop\libraries\microsoft-tfs folder. Note that you will need to upload a new SDK file in the TFS repository settings once Tasktop is started back up.
12. Restart Tasktop and Keycloak
13. Open the errors page and resolve errors related to unsatisfied connector requirement

Recovering from an error during upgrade

If Tasktop fails to restart after an upgrade, or there are errors starting the integrations, then Tasktop will need to be returned to the previous version.

1. Shutdown Tasktop
2. Remove the new version and restore the archived version (Linux) or uninstall and run previous installer (Windows)
3. Delete the <install-location>/tasktop/db (Linux) or <ProgramData>\Tasktop\db (Windows) folder and replace it with the backed-up version
4. Restart Tasktop

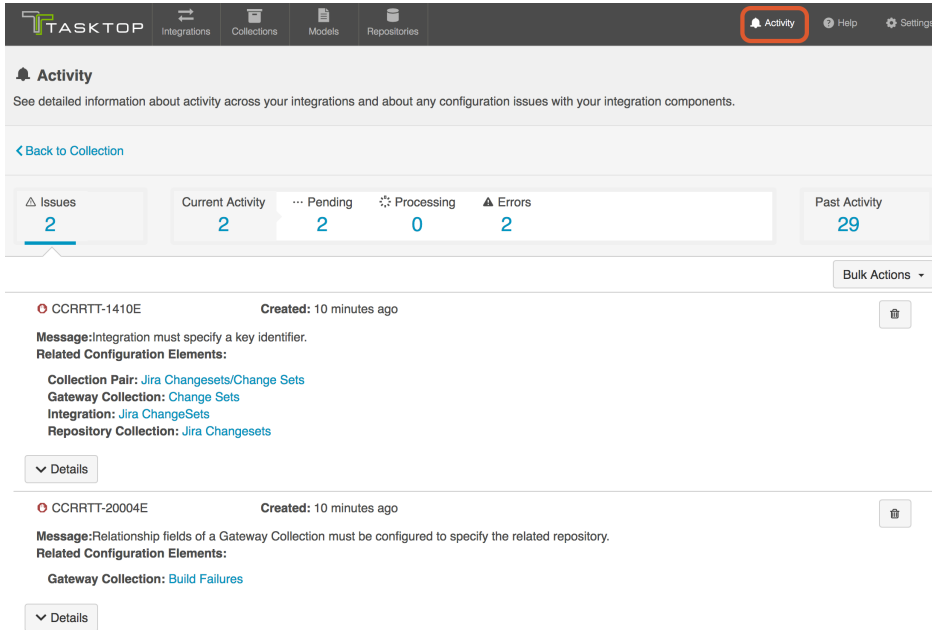
Troubleshooting

Tasktop: 17.4 Release

Activity Screen

Most problems can be solved by looking at the Activity screen and following steps described on the errors displayed there. The Activity screen can be seen by clicking on 'Activity' in the top right corner of the web application menu bar:

- Activity Screen
 - Issues
 - Current Activity
 - Pending
 - Processing
 - Error
 - Past Activity
- In-Application Errors
 - External Database Error
- Error Report

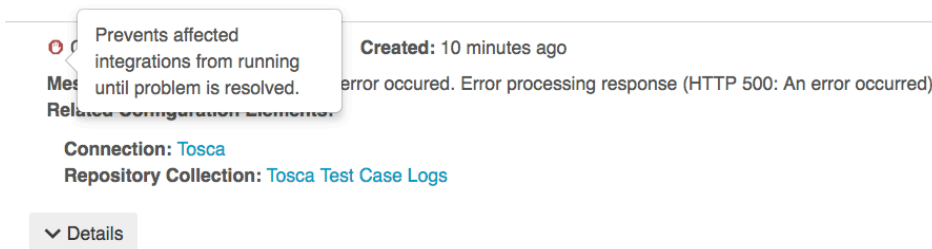


- Error Report Contents
- Configuring Logging

Issues

The *Issues* category shows issues that arise from invalid configuration of the integration components you have set up on Tasktop or from more global issues, such as having an invalid or expired license. These are things that can generally be resolved within the Tasktop application itself.

An additional warning icon appears when these issues are so fundamental that they will prevent integrations from running:



Current Activity

The *Current Activity* category shows events that are active in an integration and provides a good view to monitor what is happening in your integrations.

Current Activity encompasses the following:

- Pending: These are events that are queued up to be processed.
- Processing: These are events that are currently processing.
- Error: These are events that Tasktop tried to process but that, for one reason or another, were not successful.

You can take different actions on the events in these different subcategories, which are outlined in the sections below.

You can filter each type of current activity by entering search terms, by filtering on integration status (running or stopped), or by filtering on integration name.

Activity
See detailed information about activity across your integrations and about any configuration issues with your integration components.

[← Back to Collection](#)

Issues: 2 | Current Activity: 1 | Pending: 1 | Processing: 0 | Errors: 1 | Past Activity: 27

Search list results | Select Integration Status | Select Integration | Events: 1 | Bulk Actions

Previous 1 Next | Result 1 - 1 of 1

Processing Artifact: TPA-2 Bug A

Integration: JIRA Bugs -> ServiceNow Problems

Message: An unexpected connector error occurred. "Trivial" is not a valid option, allowed options are: "Critical", "High", "Medium", "Low". Please ensure that the value you're attempting to map to exists and that it is configured in the option mappings. The problem occurred while transforming values from fields Priority (priority) to Priority (priority).

Started: 2 minutes ago
Scheduled for Processing: 2 minutes ago
Retries: 1

Each category also allows you to take bulk actions:

Issues: 2 | Current Activity: 1 | Pending: 1 | Processing: 0 | Errors: 1 | Past Activity: 27

Search list results | All Integrations | Select Integration | Events: 1 | Bulk Actions

Previous 1 Next

Processing Artifact: TPA-2 Bug A

Integration: JIRA Bugs -> ServiceNow Problems

Message: An unexpected connector error occurred. "Trivial" is not a valid option, allowed options are: "Critical", "High", "Medium", "Low". Please ensure that the value you're attempting to map to exists and that it is configured in the option mappings. The problem occurred while transforming values from fields Priority (priority) to Priority (priority).

Project: Project: Test Project A, Issue Type: Bug

Repository: Jira Defects

Collection:

Details

Started: 8 minutes ago
Scheduled for Processing: 4 minutes ago
Retries: 4

Note: The number of events in the summary banner will update regularly, but the list of events themselves will need to be refreshed to show new activity.

This is to avoid items unexpectedly appearing and disappearing when you might be examining them.

The screenshot shows the Tasktop Activity interface. At the top, there are navigation tabs for Integrations, Collections, Models, and Repositories. The main section is titled 'Activity' and provides a summary of activity across integrations. A summary bar is highlighted with a red box, showing: 2 Issues, 0 Current Activity, 0 Pending, 0 Processing, and 0 Errors. Below this, there is a search bar and a list of events. One event is expanded, showing details for a processing artifact 'TPA-2 Bug A'. The message indicates an unexpected connector error. A red arrow points to a yellow 'Refresh' button at the bottom right of the artifact details.

Pending

On *Pending* Activity, you can take the following actions:

- **Prioritize:** Prioritize this pending event in the queue.
- **Cancel:** Remove this event from the pending queue. It will not be processed, though subsequent changes to artifacts will trigger another event.

Processing

The *Processing* tab shows activity that is currently processing. There are no actions that can be taken here.

Error

The *Error* tab shows any errors that have occurred.

You can take the following actions:

- **Prioritize:** Prioritize the retry of this error in the queue. This option is especially useful if you have made changes in your repository or in Tasktop that will likely clear up the error.
 - You will see this action if the event is already set to be retried, and is hence both in "error" and "pending" states simultaneously.
- **Retry:** Retry this error.
 - You will see this action if the event is not already set to be retried.
- **Cancel:** Remove this error from the list. It will not be retried, though subsequent changes to artifacts will trigger another event.
- **Recreate:** If a previously-sync'ed artifact has been deleted in one of your repositories, you have the option of recreating it from the Activity screen. This will keep the newly recreated artifact in sync with its corresponding artifact.

... Processing Artifact: TPA-2 Bug A PRB0041105 Bug A

Integration: JIRA Bugs -> ServiceNow Problems

Message: Unable to propagate artifact changes since the target artifact has been removed. Artifact PRB0041105 not found.

Collection Pair: ServiceNow Problems/Jira Defects

Project: Project: Test Project A, Issue Type: Bug

Repository: Jira Defects

Collection: ServiceNow Problems

Started: a few seconds ago
Scheduled for Processing: a few seconds ago
Retries: 1

... Processing Artifact: TPA-5 Bug B

Integration: JIRA Bugs -> ServiceNow Problems

Message: An unexpected connector error occurred. "Trivial" is not a valid option, allowed options are: "Critical", "High", "Medium", "Low". Please ensure that the value you're attempting to map to exists and that it is configured in the option mappings. The problem occurred while transforming values from fields Priority (priority) to Priority (priority).

Project: Project: Test Project A, Issue Type: Bug

Repository: Jira Defects

Collection:

Started: 2 minutes ago
Scheduled for Processing: a few seconds ago
Retries: 3

★ Note: Most errors will automatically be retried on a gradually decreasing interval (granted that Tasktop can locate the artifact that is to be changed). Retryable errors will be retried approximately 30 seconds after they are first encountered, and then on a gradually decreasing interval over time.

You can see information about errors being retried on the error itself. In the example below, you can see that the error has been retried 3 times. If an error will not be retried, this information will not be relevant and hence will not be displayed.

... Processing Artifact: TPA-5 Bug B

Integration: JIRA Bugs -> ServiceNow Problems

Message: An unexpected connector error occurred. "Trivial" is not a valid option, allowed options are: "Critical", "High", "Medium", "Low". Please ensure that the value you're attempting to map to exists and that it is configured in the option mappings. The problem occurred while transforming values from fields Priority (priority) to Priority (priority).

Project: Project: Test Project A, Issue Type: Bug

Repository: Jira Defects

Collection:

Started: 4 minutes ago
Scheduled for Processing: 3 minutes ago
Retries: 3

A complete listing of errors is available in [the appendix](#).

Past Activity

The *Past Activity* tab allows you to view all past integration activity, so that you can understand what has successfully completed.

Activity
See detailed information about activity across your integrations and about any configuration issues with your integration components.

Issues: 2 | Current Activity: 2 | Pending: 2 | Processing: 1 | Errors: 2 | **Past Activity: 29**

Search list results | Select Integration | Select Date Filter | Events: 29 | Bulk Actions

Previous 1 2 3 Next | Result 1 - 10 of 29

- Created Artifact: PRB0041111 Bug B
 - Integration: JIRA Bugs -> ServiceNow Problems
 - Source Artifact: TPA-5 Bug B
 - Target Artifact: PRB0041111 Bug B
 - Started: 13 minutes ago
 - Completed: 13 minutes ago
- Updated Artifact: PRB0041105 Bug A
 - Integration: JIRA Bugs -> ServiceNow Problems
 - Source Artifact: TPA-2 Bug A
 - Target Artifact: PRB0041105 Bug A
 - Started: 34 minutes ago
 - Retries: 12
 - Completed: 24 minutes ago

You can click the drop down arrow on each activity to see more details on the activity that has occurred

Updated Artifact: PRB0041105 Bug A

Integration: JIRA Bugs -> ServiceNow Problems | Started: an hour ago | Retries: 12 | Completed: 35 minutes ago

Source Artifact: TPA-2 Bug A

Target Artifact: PRB0041105 Bug A

Items	New Value	Original Value
Priority	1 - Critical	2 - High

💡 If past activity is indicating that a new artifact was created, you'll see that the Original Values listed are blank, and that the Activity type is 'Created Artifact' as opposed to 'Updated Artifact'

Created Artifact: PRB0041111 Bug B

Integration: JIRA Bugs -> ServiceNow Problems | Started: 26 minutes ago | Completed: 26 minutes ago

Source Artifact: TPA-5 Bug B

Target Artifact: PRB0041111 Bug B

Items	New Value	Original Value
Artifact Type	Problem	
Priority	3 - Moderate	
Problem state	Pending Change	
Short description	Bug B	

If you'd like to filter your results, you can use the search box on this page to refine your results. Additionally, you can use the integration filter to search by integration, or the date filter to search either by a fixed date range or by a set number of days in the past (which will dynamically update your results as days pass).

Activity
See detailed information about activity across your integrations and about any configuration issues with your integration components.

Issues: 2 | Current Activity: 2 | Pending: 0 | Processing: 0 | Errors: 2 | Past Activity: 29

Search list results | Select Integration | Select Date Filter | Events: 29 | Bulk Actions

Previous 1 2 3 Next | Result 1 - 10 of 29

- Created Artifact: PRB0041111 Bug B
 - Integration: JIRA Bugs -> ServiceNow Problems
 - Source Artifact: TPA-5 Bug B
 - Target Artifact: PRB0041111 Bug B
 - Started: 26 minutes ago
 - Completed: 26 minutes ago
- Updated Artifact: PRB0041105 Bug A
 - Integration: JIRA Bugs -> ServiceNow Problems
 - Source Artifact: TPA-2 Bug A
 - Target Artifact: PRB0041105 Bug A
 - Started: an hour ago
 - Retries: 12
 - Completed: 39 minutes ago

You can also use the Bulk Actions to refresh, or remove all past activity

that meets your current search filters. If you have not entered any search filters, all past activity will be refreshed or removed.

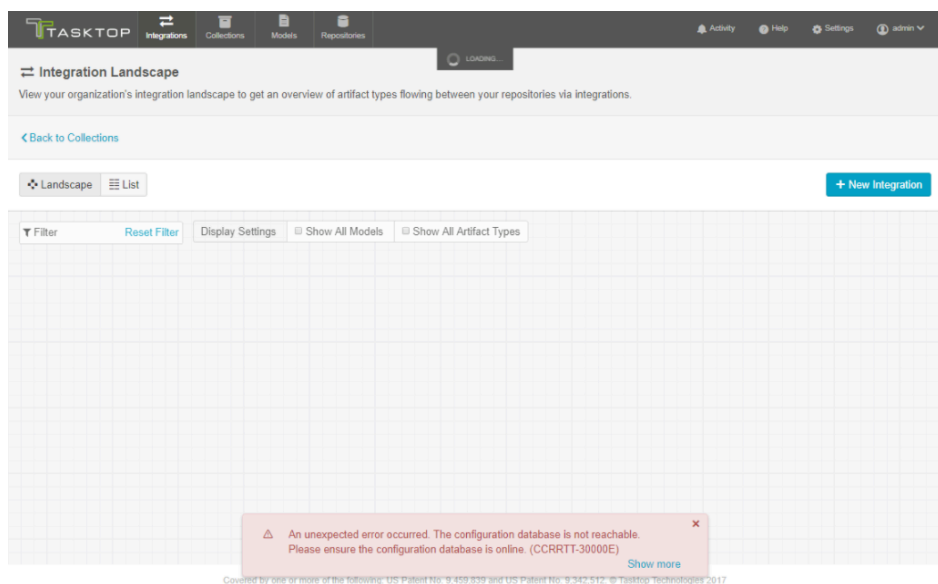
⚠ Note that Tasktop will store up to 100,000 entries on the Past Activity screen. Once 100,000 entries are met, older entries will be deleted as new entries come in. You can also opt to clear your entries when approaching 100,000 to have better visibility into more recent past activity.

In-Application Errors

There are some scenarios where you may see an error message within the application itself, rather than on the Activity Screen.

External Database Error

If you have exported your Tasktop configuration information to an external database (see information [here](#)), and your database is not reachable, you will notice that your configuration elements (i.e. repositories, collections, integrations, etc.) will not be visible, and an error message will appear. To resolve this error, please ensure that your external configuration database is online.



Error Report

In cases where the Activity screen is not enough to resolve a problem, an Error Report is available to provide additional information.

The Error Report can be downloaded from the Help screen. To download, click the "Download error report" link in the System Information section on the Help screen.

Help
Learn about getting help and providing feedback.

[< Back to Settings](#)

Help
See the [Tasktop Integration Hub User Guide](#) for detailed information.

Additional Resources
For additional resources and support, go to the [Tasktop Resource Center](#).

REST API
The Tasktop application provides REST API for retrieving platform errors. See the [REST API documentation](#) for details.

System Information
Build 17.3.0.v20170621-1704-SNAPSHOT [details](#)
[Download error report](#) for Tasktop support.

Error Report Contents

The downloaded Error Report file is named tasktop-state-DATE-TIME.zip. Once unzipped, there will be three folders. The folders and contents are listed below.

1. configuration
 - configuration.json
 - platform-details.json
2. logs
 - logs by day for past 14 days
3. metrics
 - metrics.json

File Name	Contents
configuration.json	Contains all the configuration of your application instance.
platform-details.json	Contains details about the specific build and license of the application
logs	A separate file is created for every day of logs. 14 days of logs are saved.
metrics.json	Contains various metrics of the application.

Configuring Logging

Tasktop logs various events that the application performs. These can be vital for troubleshooting purposes. Please see the [Logging](#) section of the Settings page for more details.

Extensions

Tasktop: 17.4 Release

Extensions add to Tasktop's built-in functionality to satisfy specific use cases, such as performing [state transitions](#) incorporating arbitrary business logic.

Extensions can be added to Tasktop by navigating to the 'Settings' screen, and selecting 'Manage Extensions.'

Extensions are created with a name and optional description so that they can be centrally managed and reused if needed.

Extension Language

Extensions are written in JavaScript, or more specifically [ECMAScript](#).

State Transitions

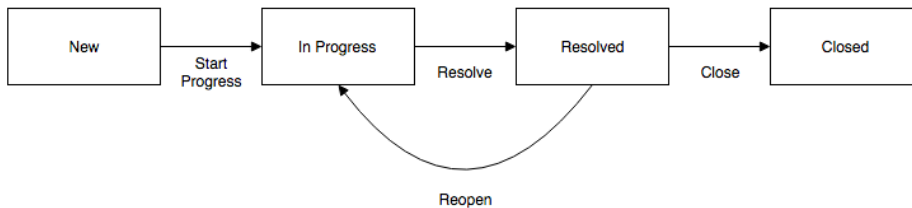
Artifact state transitions are used to transition an artifact from one status to another. To illustrate, we use the fictitious example of an artifact of type Defect with the following status values:

- New
- In Progress
- Resolved
- Closed

The status of a Defect cannot be modified directly. In this example, to move a defect from status "New" to "In Progress", the "Start Progress" transition is used.

Sometimes multiple status transitions are required. For example, to move a defect from "New" to "Closed", the following transitions are used in sequence "Start Progress", "Resolve", "Close".

The following diagram shows how state transitions are used to move a defect from one status to another:



Configuring State Transitions with Extensions

To perform state transitions, an extension can be used. Add a state transition extension from the Extensions screen, accessible from Settings. Once added, the extension can be applied from the [State Transition Sash](#) on the Collection Configuration screen.

- Extension Language
- State Transitions
 - Configuring State Transitions with Extensions
 - Authoring State Transition Extensions
- Payload Transformations
 - Configuring Gateway Collections with Extensions
 - Authoring Payload Transformation Extensions
 - Creating Multiple Artifacts From A Single Webhook Payload
- Custom Data Transformations
 - Creating a Custom Data Transformation Extension
 - Single Select and Multi Select in Custom Data Transformation Extensions
 - Rich Text Support in Custom Data Transformation Extensions
 - Web Links in Custom Data Transformation Extensions
 - Relationships in Custom

Authoring State Transition Extensions

State transition extensions are defined by a single function:

```
function  
transitionArtifact(context, transitions)
```

The function can return a single transition. For a given artifact, the extension may be called multiple times. Each time the extension is called, the transition that it returns is performed. State transition extensions are called repeatedly until they return `undefined`, indicating that no more transitions are needed.

To prevent errors, extensions are not called again if they cause an artifact to transition to the same status more than once.

A simple state transition extension could look something like this:

- Data Transformation Extensions
- Person Reconciliation
 - Configuration Person Reconciliation with Extensions
 - Authoring Person Reconciliation Extensions
 - Using LDAP or Active Directory
- Accessing Web Resources
 - HTTP Client API Reference
- Causing Extensions to Complete With An Error
- Troubleshooting Extensions
- Extensions and State
- Accessing Object Properties
 - Dot notation
 - Bracket notation

```

function
transitionArtifact(context,transitions) {

    if (context.sourceArtifact.status ===
'Resolved' &&
context.targetRepositoryArtifact.status !==
'Resolved') {
        var transition =
findTransitionWithLabel(transitions,'Resolve'
);
        transition.attributes.resolution =
'Fixed';

        return transition;

    }

}

function findTransitionWithLabel(transitions,
label) {

    for each(var transition in transitions) {

        if (transition.label === label) {

            return transition;

        }

    }

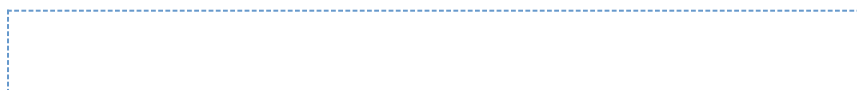
}

```

Two parameters are passed to the transitionArtifact function:

- context - a context object that provides state that the extension can use to determine which transitions are needed
 - context.sourceArtifact - a JavaScript object representation of the source artifact, whose structure matches the model configured in the integration
 - context.targetRepositoryArtifact - a JavaScript object representation of the target artifact, whose structure matches the structure of the artifact in the repository
- transitions - an array of transition objects

Below is an example of a context with a target artifact from JIRA:



```
{
  "sourceArtifact": {
    "summary": "a summary value",
    "priority": "Critical",
    "status": "Done"
  },
  "targetRepositoryArtifact": {
    "issuetype": "Bug",
    "components": null,
    "timespent": null,
    "formattedid": "TPC-144",
    "timeoriginalestimate": null,
    "project": "Test Project C",
    "description": null,
    "fixVersions": null,
    "resolution": null,
    "customfield_11500": null,
    "api-id": "JIRA",
    "attachment": null,
    "resolutiondate": null,
    "id": 14400,
    "summary": "a summary value",
    "watches": null,
    "created":
"2016-09-23T15:22:20.000+0000",
    "$closed": false,
    "reporter": "****",
    "priority": "Critical",
    "labels": null,
    "revision": null,
    "customfield_11601": null,
    "customfield_11600": null,
    "customfield_11501": null,
    "environment": null,
    "customfield_11504": null,
    "customfield_11602": null,
    "timeestimate": null,
    "versions": null,
    "duedate": null,
    "web-links": null,
    "location":
"http://jira.example.com/browse/TPC-144",
    "assignee": null,
    "worklog": null,
    "updated":
"2016-09-23T15:22:20.000+0000",
    "status": "To Do"
  }
}
```

Each transition object in the array appears as follows:

```
{
  id: 'an-id',
  label: 'A Label'
  attributes: {
    first-attribute: null,
    ...
  }
}
```

For example, transitions corresponding to the JIRA artifact example above are as follows:

```
[{
  "attributes": {
    "project": "Test Project C",
    "issuetype": "Bug"
  },
  "id": "11",
  "label": "To Do"
}, {
  "attributes": {
    "project": "Test Project C",
    "issuetype": "Bug"
  },
  "id": "21",
  "label": "In Progress"
}, {
  "attributes": {
    "project": "Test Project C",
    "issuetype": "Bug"
  },
  "id": "31",
  "label": "Done"
}]
```

Attributes of a transition are values that may be set when performing the transition. Attributes should not be set unless needed or required. The available attributes and whether or not they are required will vary depending on the type of repository of the collection.

Payload Transformations

Gateway collections can be used to accept a JSON payload via HTTP, enabling clients to use a REST API to publish artifacts in Tasktop.

Without further configuration, Gateway Collections require a JSON payload that matches the model of the collection.

By configuring a Gateway Collection with an extension, it is possible to accept arbitrarily complex JSON payloads, enabling integration with third party products that integrate with webhooks.

Examples of such third party webhook notifiers include:

- the [Jenkins Notification Plugin](#)
- [Microsoft VisualStudio Web Hooks](#)
- [GitHub Webhooks](#)

Configuring Gateway Collections with Extensions

To configure a Gateway Collection with an Extension, add a payload transformation extension from the Extensions screen, accessible from Settings. Once added, the extension can be referenced from the Gateway Collection screen.

Authoring Payload Transformation Extensions

Payload transformation extensions are defined by a single function:

```
function transformPayload(payload)
```

The function must return an array of 0 or more JSON objects matching the model of the gateway collection.

Given a model representing build jobs with the following fields:

- `created` - a date signifying the creation date
- `summary` - a brief one-line description
- `status` - a single-select indicating the build status

a simple payload transformation extension could look something like this:

```
function transformPayload(payload) {
  var createdTimestamp = new
Date(payload.build.completion_time).toISOString();
  var created =
createdTimestamp.substring(0,createdTimestamp
.indexOf('T'));
  return [
    {
      'created': created,
      'summary': payload.name + ':
'+payload.build.full_url,
      'status': payload.status
    }
  ];
}
```

The example above corresponds to the payload provided by the Jenkins Notification plugin, which provides JSON payloads as follows:

```
{
  "name": "Robot Lawnmower",
  "url": "job/Robot%20Lawnmower/",
  "build":
  {
    "full_url":
"http://build.example.com:8081/job/Robot%20La
wnmower/4/",
    "number": 4.0,
    "phase": "COMPLETED",
    "status": "FAILURE",
    "url": "job/Robot%20Lawnmower/4/",
    "scm":
    {
      },
    "causes":
    [
      "Started by user admin"
    ],
    "duration_string": "9 ms",
    "completion_time": 1.476313762942E12,
    "failing_since_build":
    {
      "full_url":
"http://build.example.com:8081/job/Robot%20La
wnmower/1/",
      "number": 1.0,
      "change_set":
      [
        ],
      "completion_time": 1.47631304791E12,
      "failing_since_time": "11 min"
    }
  }
}
```

Ignoring Webhook Payloads

For cases where the gateway collection is called and no corresponding action should be performed, the extension should return a 0-length array:

```
function transformPayload(payload) {
  ...

  if (nothingToDo) {
    return [];
  }
  ...
}
```

Creating Multiple Artifacts From A Single Webhook Payload

There may be cases when multiple artifacts should be created from a single webhook payload depending on the use case. For example, a [GitHub PushEvent](#) can contain multiple commits. To link each commit to an artifact separately, a payload transformation extension would be used as follows:

```
function transformPayload(payload) {
  var gatewayPayloads = [];
  for each (var commit in payload.commits) {

    gatewayPayloads.push(createCommitPayload(commit));
  }
  return gatewayPayloads;
}
```

Custom Data Transformations

In cases where specialized value transformations are needed for use in field mappings, such transformations can be added as custom data transformation extensions.

Creating a Custom Data Transformation Extension

Custom data transformation extensions are created from the Extensions screen, accessible from Settings. Created extensions can be selected when configuring a field mapping of a collection.

Custom data transformation extensions appear as follows:

```

var inputTypes = 'String';
var outputTypes = 'String';

function transform(context, input) {
  // returns the transformation result
}

```

All custom data transformation extensions must declare their input and output types as shown in the example above. Transformations are only available for a field mapping if the input types and output types match the fields selected in the mapping. In the case of a mapping with multiple source and target fields, the order of the declared input and output types must match the order of the source and target fields.

A simple split-and-trim value custom data transformation extension could look like this:

```

var inputTypes = 'String';
var outputTypes = ['String', 'String'];

function transform(context, input) {
  if (input) {
    var values = input.split('/');
    if (values.length != 2) {
      throw 'Unexpected value ' +
input;
    }
    return values.map(function(s) {
      return s.trim();
    });
  }
}

```

Single Select and Multi Select in Custom Data Transformation Extensions

Single Select and Multi Select values are specified using their labels. Extensions that accept a Single Select as the input type will receive a string containing the option's label. Extensions that specify a Single Select as the output type should return a string containing the option's label. To specify the empty option, return `undefined` from the extensions instead of a value. Extensions that accept a Multi Select as the input type will receive an array of strings of the option labels. Extensions that specify a Multi Select as the output type should return an array of strings with the option labels or an empty array to specify no options.

Rich Text Support in Custom Data Transformation Extensions

To perform Rich Text transformations, 'Rich Text' must be declared as input or output types of the extension.

A Rich Text input parameter is passed as a valid HTML string.

For Rich Text as output type, the extension is expected to return a valid HTML string.

To escape HTML characters, the following function is provided:

```
html.escape(string)
```

A simple String-to-Rich-Text value transformation could look like this:

```
var inputTypes = 'String';
var outputTypes = 'Rich Text';

function transform(context, input) {
  if (input) {
    return '<pre>' + html.escape(input) +
'</pre>';
  }
}
```

Web Links in Custom Data Transformation Extensions

To perform a web links transformation, web links must be declared as the input or output types of the extension. A web links field consists of a list of web link objects. A web link object consists of a location and other attributes.

The following is an example of a web link output:

```
[
  {
    label: 'Tasktop',
    location: 'http://www.tasktop.com'
  },
  {
    location: 'http://www.alt-tasktop.com'
  }
]
```

The label attribute is optional and if specified will be used to populate the label of the web link.

Relationships in Custom Data Transformation Extensions

Tasktop provides a JavaScript API for working with relationship fields. This API is able to retrieve, search and get associated artifacts for artifacts.

Artifact Service API Reference

- `artifacts.retrieveArtifact(relationship)` - retrieves the artifact for the provided relationship
- `artifacts.listSearchTypes()` - lists the valid search types for the targeted repository
- `artifacts.getSearchDefinition(searchTypeId)` - returns an object with the parameters that are required for the given search type id
- `artifacts.search(searchType, searchDefinition)` - searches the target repository with the given search type id and search definition
- `artifacts.getFormattedIdSearchDefinition()` - returns an object with the parameters that are required for a formatted-id search
- `artifacts.searchByFormattedId(searchDefinition)` - searches by formatted id with the provided search definition
- `artifacts.toContainer(relationship, summary)` - converts a relationship into a container, summary is optional
- `artifacts.toRelationship(container)` - converts a container into a relationship
- `artifacts.getAssociatedRelationship(relationship)` - finds the associated relationship for the given relationship. When mapping from model to collection the input value and source artifact relationship field values are from the source repository and must be converted to their associated value to be used in the target system. An exception is thrown if no artifact is found or multiple artifacts are found.
- `artifacts.getAssociatedContainer(container)` - finds the associated container for the given container. When mapping from model to collection the input value and source artifact container link field values are from the source repository and must be converted to their associated value to be used in the target system. An exception is thrown if no artifact is found or multiple artifacts are found.

A sample relationship transformation extension:

```

var inputTypes = 'Relationship';
var outputTypes = 'Relationship';

function transform(context, input) {
  if (input) {
    return
    findParentFolder(context.sourceArtifact);
  }
  return null;
}

function findParentFolder(artifact) {
  var parent =
  artifacts.retrieveArtifact(artifact['parent']
);
  if (parent['subtype'] === 'Folder') {
    return artifact['parent'];
  } else if (parent['subtype'] === null) {
    return null;
  }
  return findParentFolder(parent);
}

```

Looking at the above extension, we find the parent artifact and if that artifact is a folder we return that as the parent.

```

var inputTypes = 'Relationship';
var outputTypes = 'Relationship';

function transform(context, input) {
  var searchDefinition =
  artifacts.getFormattedIdSearchDefinition();

  searchDefinition['formatted-id'] = 'TPA-42';
  var results =
  artifacts.searchByFormattedId(searchDefinitio
n);
  if (results[0]) {
    return results[0];
  }
  return null;
}

```

The above extensions uses the formatted id search to find the correct artifact for the link.

The following extension uses a custom search to determine a relationship:


```

var inputTypes = 'Relationship';
var outputTypes = 'Relationship';

function transform(context, input) {
  var searchType = getCustomSearchType();
  var searchDefinition =
artifacts.getSearchDefinition(searchType);

  searchDefinition['domain'] = 'DEFAULT';
  searchDefinition['project'] = 'My Project';
  searchDefinition['summary'] =
context.sourceArtifact.summary;
  var results = artifacts.search(searchType,
searchDefinition);
  if (results[0]) {
    return results[0];
  }
  return null;
}

function getCustomSearchType() {
  var searchTypes =
artifacts.listSearchTypes();
  for (var i=0; i<searchTypes.length; i++) {
    if (searchTypes[i] === 'My Custom Search')
    {
      return searchTypes[i];
    }
  }
  return i;
}

```

Note that the returned search results are limited to a maximum of 1024 entries.

Containers and Relationships

A 'Container' can be used as input and output type in a Custom Data Transformation extension. Tasktop provides a JavaScript API for working with container fields.

The following two functions are provided to handle containers:

```
artifacts.toRelationship(container)
```

```
artifacts.toContainer(relationship[,  
summary])
```

All container objects provide a `summary` property.

- `.toContainer(relationship[, summary])` converts a relationship object into a container. The `summary` is provided as a `String` and is optional. If no `summary` is provided, the `summary` of the related artifact is used. An exception is thrown if the artifact or the `summary` field of the artifact can not be found.
- `.toRelationship(container)` converts a container into a relationship object to use with the `artifacts.retrieveArtifact(relationship)` API or return as result of the extension.

The following extension finds the first parent folder and returns that as the parent container.

```
var inputTypes = 'Relationship';  
var outputTypes = 'Container';  
  
function transform(context, input) {  
  if (input) {  
    var parentRelationship =  
      findParentFolder(context.sourceArtifact);  
    return  
      artifacts.toContainer(parentRelationship);  
  }  
  return null;  
}  
  
function findParentFolder(artifact){  
  var parent =  
    artifacts.retrieveArtifact(artifact['parent']  
);  
  if (parent['subtype'] === 'Folder') {  
    return artifact['parent'];  
  } else if (parent['subtype'] === null) {  
    return null;  
  }  
  return findParentFolder(parent);  
}
```

The next extension retrieves the parent of our parent container field and returns it as relationship.

```
var inputTypes = 'Container';
var outputTypes = 'Relationship';

function transform(context, input) {
  if (input) {
    var parentRelationship =
artifacts.toRelationship(input);
    var parentArtifact =
artifacts.retrieveArtifact(parentRelationship
);
    var container = parentArtifact['parent'];
    return
artifacts.toRelationship(container);
  }
  return null;
}
```

Note that only containers based on artifacts are supported.

Person Reconciliation

Integrations that create or update artifacts often need to deal with differences between the representation of persons in different systems. Without further configuration, a corresponding person will be determined by Tasktop's default matching algorithm (by user-id or email).

By creating an extension to map persons, the correct person can be used when creating or updating artifacts.

Configuration Person Reconciliation with Extensions

A person reconciliation extension can be created from the Extensions screen, accessible from Settings. Created extensions are selected in the Person Reconciliation section of the Collection screen. In most cases it makes sense to have one extension per repository, since each repository will have different requirements for mapping persons to and from the repository. Person reconciliation extensions apply to all person fields of an artifact, including person fields in comments and attachments.

Authoring Person Reconciliation Extensions

Person reconciliation extensions are defined by two functions:

```
mapPersonFromRepository(repositoryPerson)
```

```
mapPersonToRepository(modelPerson)
```

Both functions are expected to return a string value corresponding to the user id of the person. Returning `undefined` sets the person field to empty. In the case where a user can not be mapped and having the field empty is not an option, throw an exception as follows:

```
if (noMatchFoundCondition) {  
    throw 'some descriptive message';  
}
```

Such errors will cause processing of an artifact to result in an error with error code CCRRTT-17011E which will display under Activities & Issues.

```
mapPersonFromRepository(repositoryPerson)
```

`mapPersonFromRepository` is used to create a model representation of a person from a repository representation of a person, which occurs whenever a person is copied from a repository artifact to a model artifact. The return value of this function is used as the id of the person in the model artifact.

A single parameter is passed to the `mapPersonFromRepository` function:

- `repositoryPerson` - an object representing the person corresponding to the repository representation

An example person from JIRA looks like:

```
{  
  "person-id": "userA",  
  "person-email": "userA@test.tasktop.com",  
  "person-display-name": "User A",  
  "active": true  
}
```

```
mapPersonToRepository(modelPerson)
```

`mapPersonToRepository` is used to create a repository representation of a person from a model representation of a person, which occurs whenever a person is copied from a model artifact to a repository artifact. The return value of this function is used to lookup the corresponding person in the repository.

A single parameter is passed to the `mapPersonToRepository` function:

- `modelPerson` an object representing the person corresponding to the model representation

A `modelPerson` always has the following properties:

```
{  
  "user-id": "userId",  
  "display-name": "Jane Smith"  
}
```

Note that `display-name` could be empty.

Simple Person Reconciliation Example

A simple person reconciliation mapping extension could look like this:

```

function
mapPersonFromRepository(repositoryPerson) {
    if (repositoryPerson['email'] ===
'buildserver@mycompany.com') {
        return undefined;
    }
    var mapping = {
        'user1@mycompany.com':
'user.1',
        'user2@mycompany.com':
'user.2'
    };
    var result =
mapping[repositoryPerson['email']];
    if (!result) {
        throw 'no person found with
email='+repositoryPerson['email'];
    }
    return result;
}

function mapPersonToRepository(modelPerson) {
    var mapping = {
        'user.1':
'user1@mycompany.com',
        'user.2':
'user2@mycompany.com'
    };
    var result = mapping[modelPerson['id']];
    if (!result) {
        throw 'no person found with
id='+modelPerson['id'];
    }
    return result;
}

```

Person Reconciliation Extension Javascript API

Tasktop provides a JavaScript API for working with persons in a person reconciliation extension. This API includes two functions:

- `persons.listPersonSearchFields()` allows for the discovery of the searchable fields on a person. Not all fields from a person are searchable and vary between connectors.
- `persons.searchPerson(fieldId, fieldValue)` is used to search for person in a repository. This person can then be used to return the correct id for a user in a repository. `persons.searchPerson(fieldId, fieldValue)` will find exactly one person and will throw `aPersonNotFoundException` if no match is

found or `TooManyPersonsFoundException` if more than one person is found. These exceptions can be caught and handled by the extension.

Below is a person reconciliation extension that will take the id of a model person, retrieve the user by username and return the exact id from the repository. This is helpful for systems where the person's id is a number or some other non-human readable value.

```
function
mapPersonFromRepository(repositoryPerson) {
    return repositoryPerson['Username'];
}

function mapPersonToRepository(modelPerson) {
    // persons.listPersonSearchFields();
    determines the fields usable by
    .searchPerson(...)
    var repositoryPerson =
persons.searchPerson('Username',
modelPerson['id']);
    return repositoryPerson['ID'];
}
```

Using LDAP or Active Directory

LDAP (Lightweight Directory Access Protocol) and Active Directory can be used to lookup information required to map persons from one system to another. Tasktop provides a JavaScript API for accessing LDAP and Active Directory as follows:

```
function mapPersonToRepository(modelPerson) {
    ldap.connect('ldap://subdomain.mycompany.com
', 'cn=admin,dc=example,dc=mycompany,dc=com',
'mypassword');
    var results =
ldap.search('dc=example,dc=mycompany,dc=com',
'cn='+ldap.escape(modelPerson['id']))
    if (results.length == 0) {
        throw 'no person found with
id='+modelPerson['id'];
    }
    return results[0]['sn'];
}
```

Looking at the example above, three steps are involved:

1. establishing a connection
2. looking up the appropriate entries using a search

3. returning a value from the search results

The same approach is used for both LDAP and Active Directory.

The Tasktop JavaScript LDAP API is described as follows:

- `ldap` - the globally-visible object providing the LDAP API
- `ldap.connect(connectionUrl, principal, password)` - a means of establishing a connection with a connection URL, user principal and password
- `ldap.search(base, query, fields)` - a means of searching providing a base name of the context to search, a search query, and an optional list of fields to provide in the search results
- `ldap.escape(value)` - a means of escaping string literals to use in LDAP search queries or distinguished names

There is no need to close an LDAP connection; LDAP connections are managed implicitly by Tasktop.

Accessing Web Resources

Extensions may access resources using HTTP. For example, extensions may access a REST API which could provide data necessary for the extension.

Tasktop provides a fluent JavaScript API for making HTTP requests, inspired by the Java 9 HTTP client API. The API is used as follows:

```
var response = httpClient.request()
    .uri('http://example.com/my/rest/api')
    .parameter('first-param', 'first-value')
    .parameter('second-param', 'second-value')
    .header('my-special-header', 'header-value')
    .GET().response()

if (response.statusCode() == 200) {
    var responseJson =
JSON.parse(response.content());
    // do something with response data
}
```

HTTP Client API Reference

- `httpClient` - the globally-visible object providing the HTTP client API
- `httpClient.request()` - provides a `RequestBuilder` object
- `RequestBuilder.uri(uriString)` - specifies the URI of the request
- `RequestBuilder.parameter(key, value)` - adds a query parameter to the request with the given key and value
- `RequestBuilder.header(key, value)` - adds an HTTP

- header value to the request with the given key and value
- `RequestBuilder.GET()` - creates a `Request` object for an HTTP GET request
- `Request.response()` - creates a `Response` object with the result of the HTTP request
- `Response.statusCode()` - provides the HTTP status code of the response
- `Response.content()` - provides the body of the HTTP response as a string
- `Response.headers()` - provides the HTTP response headers as a JavaScript object with property names corresponding to HTTP header names, and values as arrays of values of the corresponding HTTP header

Example extension `Response.headers()` return value:

```
{
  "Transfer-Encoding": [
    "chunked"
  ],
  "Server": [
    "Jetty(9.2.13.v20150730)"
  ],
  "Vary": [
    "Accept-Encoding, User-Agent"
  ],
  "Content-Type": [
    "application/json;charset=UTF-8"
  ]
}
```

Causing Extensions to Complete With An Error

There are occasions where extensions should complete with an error. In such cases, simply use the JavaScript `throw` keyword as follows:

```
if (somethingUnexpected) throw 'some
descriptive message'
```

Such errors will cause processing of an artifact to result in an error with error code CRRRTT-17011E which will display under Activities & Issues.

Troubleshooting Extensions

Extension troubleshooting usually involves trial and error. To make the troubleshooting process easier, a global logging function is exposed as follows:

```
console.log(message)
```

`console.log` takes a single argument which is converted to a string.

For example:

```
function
transitionArtifact(context,transitions) {
    if (someUnexpectedCondition) {
        console.log('source artifact:
'+JSON.stringify(context.sourceArtifact));
        console.log('target artifact:
'+JSON.stringify(context.targetRepositoryArti
fact));
        console.log('transitions:
'+JSON.stringify(transitions));
        throw 'message describing that
something bad happened';
    }
}
```

The output of `console.log` goes to the Tasktop log file at `logs/extensions.log`

Extensions and State

Extensions should not rely on declared variables to retain state between invocations. Doing so is not supported and has undefined behavior.

For example:

```
// this is not supported:
```

```
var myGlobalState = // some state

function someFunction() {
    if (myGlobalState == someValue) {
        ...
    }
}
```

Accessing Object Properties

There are two ways to access object properties:

Dot notation

You can use the dot notation if the property name only contains

alpha-numeric and characters that are allowed in JavaScript variables such as '\$' or '_'.

For example:

```
person.email
```

Bracket notation

You must use the bracket notation if the property name contains characters that are not allowed in JavaScript variables such as a hyphen.

For example:

```
person[ 'user-id' ]
```

Business Continuity

Overview

This document describes the features of Tasktop Integration Hub intended to support a business continuity process. Tasktop Integration Hub maintains information critical to organizational business processes, and therefore should be included in a comprehensive business continuity plan that safeguards data and ensures business continuity in hardware and operational failure scenarios.

- [Overview](#)
- [Data Loss Prevention](#)
- [Downtime](#)
- [Backup](#)
- [Restore](#)
- [Failover](#)

Data Loss Prevention

An important aspect of disaster avoidance is avoidance of data loss. Tasktop Integration Hub should be configured to use a reliable external database such as Oracle or Microsoft SQL Server. Please see our [Installation & Hardware Requirements](#) section to see which databases are supported for storing operational data.

External databases should be set up with sufficient redundancy to maximize uptime and to reduce the probability of data loss due to hardware failure. For details on how to set up your external database, please see our [Settings page](#).

Downtime

When Tasktop service is unavailable, changes may be taking place in integrated repositories. Normal Tasktop operation ensures that data flows between these repositories in a timely manner. When the server is unavailable, however, information is no longer propagating between integrated systems.

This has the following impacts:

1. Synchronization integrations will not create or update artifacts in synchronized repositories
2. Enterprise Data Stream integrations will not record artifact changes from their integrated source repositories to their target databases, which may cause a loss of fidelity in reporting data
3. Gateway integrations cannot accept payloads from integrated gateway collections; this can result in data loss if the integrated tools cannot handle the downtime

Upon restarting Tasktop Integration Hub, integrations will resume with the following effects:

1. All Synchronization integrations will begin processing where they left off when the server became unavailable; there may be a backlog of changes to process, but no synchronizations will be lost
2. Enterprise Data Stream integrations will begin detecting artifact changes; any changes that occurred when service was unavailable will be detected, but multiple changes to the same field will have lost fidelity (only one change to that field will be reported)
3. Tasktop will begin accepting Gateway collection payloads, and if the integrated repositories are configured correctly to retry payloads, they will be processed as usual without data loss

Backup

A working backup strategy is a critical element of disaster recovery, since only backups can mitigate complete hardware failure and user error. A backup strategy that ensures correct and current backups is essential. Backups of the Hub database include both configuration and operational data.

Backup frequency should mirror your practices for all software tools your organization utilizes. At bare minimum, backup frequency should be daily, ideally with incremental backups performed more frequently.

In order to Backup Tasktop Integration Hub, follow the instructions below:

1. Ensure that you have migrated your operational data to an external database. For details on how to set up your external database, please see our [Settings page](#).
2. Backup the tasktop-db.json file
3. Backup the external database using that database's backup tools
4. Backup the Tomcat and Catalina configuration (Note: this only needs to occur if/when changes are made to the Tomcat and Catalina configuration).

Restore

In order to restore Tasktop Integration Hub, follow the instructions

below:

1. Restore the external database backup using the tools from that database
2. Restore the backed up Tomcat and Catalina configuration files from part 4 of the backup instructions

Failover

Tasktop Integration Hub supports failover for installations that use external databases by replicating a Tasktop install across multiple hosts with the same database connection. Only one instance of Tasktop Integration Hub may run at once; it is important to ensure that two instances of Tasktop Integration Hub are not run against the same external database; if this happens, Tasktop could create duplicate data in the integrated repositories. A failover instance of Tasktop should only be started when the primary instance has crashed and will not come back online.

Resources

Resources

Tasktop: 17.4 Release

Help and Support

To learn more about Tasktop, see [our website](#)

For help, contact us at the [Tasktop Support Center](#).

Feedback and Ideas

Have a suggestion or an idea for the product? Please contact us at feedback@tasktop.com.

Appendix A- Error Messages

Appendix A: Error Messages

Tasktop: 17.4 Release

The following is a complete list of error messages. Error messages are displayed in the [Activity view](#). More information about the Errors view can be found under [Troubleshooting](#).

- CCRRTT-0001E – An unexpected error occurred.
 - [Description](#)
 - [User Action](#)

- CCRRTT-0002E – The maximum number of allowable errors has been reached.
 - Description
 - User Action
- CCRRTT-0003E – The system has run out of memory.
 - Description
 - User Action
- CCRRTT-0004E – Configuration migration failed.
 - Description
 - User Action
- CCRRTT-0005E – There is a conflicting artifact association.
 - Description
 - User Action
- CCRRTT-0006W – Upgrade data migration cancelled.
 - Description
 - User Action
- CCRRTT-1000E – Unable to communicate with repository.
 - Description
 - User Action
- CCRRTT-1001E – Connector is missing requirements.
 - Description
 - User Action
- CCRRTT-1002E – An unexpected connector error occurred.
 - Description
 - User Action
- CCRRTT-1003E – An error occurred while executing an operation.
 - Description
 - User Action
- CCRRTT-1004E – Connection to LDAP directory failed.
 - Description
 - User Action
- CCRRTT-1005E – An unexpected error occurred while communicating with an LDAP directory.
 - Description
 - User Action
- CCRRTT-1101E – Connection credentials were not accepted by the repository.
 - Description
 - User Action
- CCRRTT-1102E – Connection HTTP proxy credentials were not accepted by the repository.
 - Description
 - User Action
- CCRRTT-1103E – Connection settings are invalid.
 - Description
 - User Action
- CCRRTT-1104W – Authentication state for repository connection has expired.
 - Description
 - User Action
- CCRRTT-1105E – Repository Collection project is invalid.
 - Description
 - User Action
- CCRRTT-1107E – Connection could not be established with a repository due to a failure during authentication.
 - Description
 - User Action

- CCRRTT-1108W – API call limit on repository has been exceeded.
 - Description
 - User Action
- CCRRTT-1109E – Repository Collection project configuration is outdated.
 - Description
 - User Action
- CCRRTT-1401E – Integration must specify at least one route.
 - Description
 - User Action
- CCRRTT-1402E – Integration must satisfy style constraints.
 - Description
 - User Action
- CCRRTT-1403E – Integration must have all collections attached to the same model.
 - Description
 - User Action
- CCRRTT-1404E – Collection must have a mapping to a model.
 - Description
 - User Action
- CCRRTT-1405E – Integration must have a source Collection.
 - Description
 - User Action
- CCRRTT-1406E – Integration must have a target Collection.
 - Description
 - User Action
- CCRRTT-1408E – Integration failed to lookup artifact.
 - Description
 - User Action
- CCRRTT-1409E – Integration has invalid filter.
 - Description
 - User Action
- CCRRTT-1410E – Integration must specify a key identifier.
 - Description
 - User Action
- CCRRTT-1411E – All specified routes of an integration must be configured.
 - Description
 - User Action
- CCRRTT-1412E – Integration has a conditional route with invalid configuration.
 - Description
 - User Action
- CCRRTT-1413E – Collection has invalid repository query.
 - Description
 - User Action
- CCRRTT-10004E – Enterprise Data Stream Integration must have exactly one target SQL Collection.
 - Description
 - User Action
- CCRRTT-10005E – Enterprise Data Stream Integration must have a source Collection.
 - Description
 - User Action
- CCRRTT-10006E – Enterprise Data Stream Integration target Collection must have appropriate mapping.
 - Description
 - User Action

- CCRRTT-10007E – Enterprise Data Stream Integration source Collection must provide the correct model.
 - Description
 - User Action
- CCRRTT-10008E – Enterprise Data Stream Integration target Collection must have exactly one project.
 - Description
 - User Action
- CCRRTT-15002E – Integration services cannot be started due to a problem with the license.
 - Description
 - User Action
- CCRRTT-15005E – Repository cannot be used due to a problem with the license.
 - Description
 - User Action
- CCRRTT-15006E – Integration services cannot be started since the current license has expired.
 - Description
 - User Action
- CCRRTT-15007E – Integration cannot be used with the configured repositories due to a restriction in the license.
 - Description
 - User Action
- CCRRTT-15008E – Collection cannot be used with the configured person reconciliation extension due to a restriction in the license.
 - Description
 - User Action
- CCRRTT-15009E – Collection cannot be used with the configured state transition extension due to a restriction in the license.
 - Description
 - User Action
- CCRRTT-15010E – Mapping cannot be used with the configured custom data transformation extension due to a restriction in the license.
 - Description
 - User Action
- CCRRTT-15011E – Your licensed user count has been exceeded.
 - Description
 - User Action
- CCRRTT-16001E – Services cannot be started until Tasktop security has been initialized.
 - Description
 - User Action
- CCRRTT-16002E – Error initializing password encryption.
 - Description
 - User Action
- CCRRTT-17001E – Mapping cannot be applied since it is not valid within the current context.
 - Description
 - User Action
- CCRRTT-17002E – Collection model mapping is invalid.
 - Description
 - User Action
- CCRRTT-17003E – Artifact could not be created or updated because one or more values cannot be accepted.
 - Description
 - User Action

- CCRRTT-17004W – Artifact cannot be processed since it is currently in use.
 - Description
 - User Action
- CCRRTT-17005E – Field flow is invalid.
 - Description
 - User Action
- CCRRTT-17006E – Artifact was created but some values could not be set.
 - Description
 - User Action
- CCRRTT-17007E – Conflict resolution strategy is invalid.
 - Description
 - User Action
- CCRRTT-17008E – Artifact could not be processed as it did not meet any of the configured conditions on the Conditional Artifact Routing page.
 - Description
 - User Action
- CCRRTT-17009E – Invalid state transition.
 - Description
 - User Action
- CCRRTT-17010E – Repeated state transition.
 - Description
 - User Action
- CCRRTT-17011E – Extension completed with an error.
 - Description
 - User Action
- CCRRTT-17013E – The state transition requires the selection of model fields.
 - Description
 - User Action
- CCRRTT-17014E – Relationship values could not be resolved during synchronization.
 - Description
 - User Action
- CCRRTT-17015E – Relationship values could not be resolved during synchronization.
 - Description
 - User Action
- CCRRTT-17016E – An unexpected error occurred when creating the artifact.
 - Description
 - User Action
- CCRRTT-17017E – The repository does not support artifact creation.
 - Description
 - User Action
- CCRRTT-17018E – Model does not have all fields required by the state transition.
 - Description
 - User Action
- CCRRTT-20000E – No integration is listening to the Gateway Collection.
 - Description
 - User Action
- CCRRTT-20001E – Time Tracking integration model must have a field of type time entries.
 - Description
 - User Action
- CCRRTT-20002E – Time Tracking integration Collection must have a field mapping to a field of type time entries in the Model.
 - Description

- User Action
- CCRRTT-20003W – Time Tracking integration target Collection does support impersonation of the Worker field.
 - Description
- CCRRTT-20004E – Relationship fields of a Gateway Collection must be configured to specify the related repository.
 - Description
 - User Action
- CCRRTT-20005E – Gateway collection must have a model.
 - Description
 - User Action
- CCRRTT-20006E – Gateway Collection cannot be used with the configured payload transformation extension due to a restriction in the license.
 - Description
 - User Action
- CCRRTT-30000E – An unexpected error occurred.
 - Description
 - User Action
- CCRRTT-30001E – Not found.
 - Description
 - User Action
- CCRRTT-30002E – The data provided was not valid.
 - Description
 - User Action
- CCRRTT-30003E – The connector kind was not found.
 - Description
 - User Action
- CCRRTT-30004E – The request entity was not valid JSON.
 - Description
 - User Action
- CCRRTT-30005E – Secure password storage must be initialized.
 - Description
 - User Action
- CCRRTT-30006E – Error communicating with {0} repository.
 - Description
 - User Action
- CCRRTT-30007E – Error processing request MIME attachment.
 - Description
 - User Action
- CCRRTT-30008E – Tasktop is stopped, see the Activity View and error log for more details.
 - Description
 - User Action
- CCRRTT-30009E – The database is not available.
 - Description
 - User Action
- CCRRTT-30010E – Connection settings are not valid.
 - Description
 - User Action
- CCRRTT-30011E – The database is locked for maintenance and cannot currently be used.
 - Description
 - User Action
- CCRRTT-50001E – Unable to propagate artifact changes since the target artifact has been removed.

- Description
- User Action
- CCRRTT-50002E – A conflict has occurred during synchronization.
 - Description
 - User Action
- CCRRTT-50005E – A conflict has occurred during synchronization.
 - Description
 - User Action

CCRRTT-0001E – An unexpected error occurred.

Description

An unexpected error has occurred.

User Action

Attempt to resolve error according to the specific error message.

CCRRTT-0002E – The maximum number of allowable errors has been reached.

Description

The maximum number of allowable errors has been reached. Any errors encountered after the maximum number will be discarded.

User Action

1. Open the errors page and resolve the listed errors

CCRRTT-0003E – The system has run out of memory.

Description

The system has run out of memory. Services have been stopped.

User Action

1. Increase the amount of memory available (see help docs).
2. Restart Tasktop.

CCRRTT-0004E – Configuration migration failed.

Description

Configuration could not be migrated to match an updated version of Tasktop due to one or more errors.

User Action

1. Investigate the cause of failure by viewing related errors under Issues on the Activities & Issues page.
2. Attempt to resolve error according to the specific error message and corresponding user actions.
3. Restart the Tasktop application.

CCRRTT-0005E – There is a conflicting artifact association.

Description

The artifact association could not be imported as an existing artifact association conflicts with it.

User Action

Contact support for assistance.

CCRRTT-0006W – Upgrade data migration cancelled.

Description

Data migration required to run an updated version of Tasktop was cancelled due to a configuration change or because Tasktop was shutdown.

User Action

None, data migration will be resumed automatically.

CCRRTT-1000E – Unable to communicate with repository.

Description

There was a network error when attempting to communicate with a repository.

User Action

1. Check the network connection between Tasktop and the repository.
2. Try connecting again later.

If the problem persists, contact your network administrator.

CCRRTT-1001E – Connector is missing requirements.

Description

The connector requirements are not met.

User Action

Read the connector-specific error message to determine which requirements are unsatisfied.

To provide 3rd party components such as a library or SDK, follow the following steps:

1. Navigate to the "Connections" page.
2. Select the connection for which the requirements were unsatisfied.
3. On the connection page, provide the required files.

CCRRTT-1002E – An unexpected connector error occurred.

Description

An unexpected connector exception has occurred.

User Action

Attempt to resolve error according to the specific error message.

CCRRTT-1003E – An error occurred while executing an operation.

Description

An exception has occurred during the execution of a connector operation.

User Action

Attempt to resolve error according to the specific error message.

CCRRTT-1004E – Connection to LDAP directory failed.

Description

An unexpected error has occurred while attempting to establish a connection with an LDAP directory.

User Action

Attempt to resolve error according to the specific error message.

CCRRTT-1005E – An unexpected error occurred while communicating with an LDAP directory.

Description

An unexpected error has occurred while communicating with an LDAP directory.

User Action

Attempt to resolve error according to the specific error message.

CCRRTT-1101E – Connection credentials were not accepted by the repository.

Description

There was an authentication error while attempting to communicate with a repository.

User Action

1. Verify that the credentials for the associated repository are correct in the settings.

If these steps do not resolve the error, ensure that the user has sufficient permissions in the target repository to create and edit artifacts.

CCRRTT-1102E – Connection HTTP proxy credentials were not accepted by the repository.

Description

There was an authentication error with the proxy server while attempting to communicate with a repository.

User Action

1. Verify that the proxy credentials for the associated repository are correct in the settings.

If these steps do not resolve the error, contact your network administrator for assistance.

CCRRTT-1103E – Connection settings are invalid.

Description

The connection settings are invalid.

User Action

1. Open the connection settings page for the repository that is in error.
2. Update the connection's settings to valid values.

If these steps do not resolve the error, contact support for additional assistance.

CCRRTT-1104W – Authentication state for repository connection has expired.

Description

The authentication state for a repository connection has expired.

User Action

Typically, the authentication state for a repository connection expires on a periodic basis and authentication will be retried automatically. If the error persists, verify that the repository credentials for the associated repository are correct.

CCRRTT-1105E – Repository Collection project is invalid.

Description

The Repository Collection project is not valid. This problem is usually caused by a project and/or type being deleted from the repository, but can also be caused by other problems such as a change in user permissions within the repository.

User Action

1. Determine the cause of the problem from the specific error message
2. Correct the problem on the repository and then click "Refresh Projects" on the Repository Collection, or
3. Remove the referenced project from the Repository Collection

CCRRTT-1107E – Connection could not be established with a repository due to a failure during authentication.

Description

There was an unexpected error while attempting to authenticate with a repository.

User Action

Attempt to resolve error according to the specific error message.

CCRRTT-1108W – API call limit on repository has been exceeded.

Description

The API limit imposed by the repository has been exceeded. This problem is usually caused during periods of heavy load.

User Action

This error will resolve itself automatically when the repository is no longer imposing a rate limit.

CCRRTT-1109E – Repository Collection project configuration is outdated.

Description

The Repository Collection project configuration is outdated.

User Action

1. Identify the outdated project configured from the specific error message
2. Remove the outdated project from the Repository Collection
3. Select "Manage Projects" and press the "Refresh" button in the Repository Collection
4. Add the project back to the Repository Collection

CCRRTT-1401E – Integration must specify at least one route.

Description

An integration must contain at least one route.

User Action

1. Navigate to the integration routing page
2. Add at least one route

CCRRTT-1402E – Integration must satisfy style constraints.

Description

An integration must satisfy the constraints of its style. This type of error should not happen when an integration is built using the UI.

See the detailed message for more details about the parts of the integration that are invalid.

User Action

1. Navigate to the integration page
2. Adjust the configuration to be valid (according to the messages)
3. If this integration was created via the web UI, consider contacting support

CCRRTT-1403E – Integration must have all collections attached to the same model.

Description

Collections used in an integration must all be attached to the same model.

User Action

1. Determine which model the integration should be using
2. Navigate to the integration and determine which collections are not using this model
3. Either remove the identified collections from the integration, or
4. For each identified collection, set the mapping to the correct model

CCRRTT-1404E – Collection must have a mapping to a model.

Description

Repository Collections used in an integration must have a mapping to a model.

User Action

1. Navigate to the collection
2. Select a Model to create a mapping

CRRRTT-1405E – Integration must have a source Collection.

Description

An integration must have a source collection.

User Action

1. Navigate to the Integration
2. Add a collection to be used as a source

CRRRTT-1406E – Integration must have a target Collection.

Description

An integration must have a target collection.

User Action

1. Navigate to the Integration
2. Add a collection to be used as a source

CRRRTT-1408E – Integration failed to lookup artifact.

Description

An integration failed to locate the artifact to be modified. This can be caused by:

- a missing formatted ID value on the source artifact,
- an invalid formatted ID value on the source artifact, or
- the absence of a target collection which contains an artifact matched by the formatted ID.

See the detailed message for more details about the parts of the lookup that failed.

User Action

1. Navigate to the integration page
2. Ensure the key field is configured correctly on the field flow page
3. Ensure the data on the source artifact is correct
4. Ensure a matching artifact is contained in a target collection

CCRRTT-1409E – Integration has invalid filter.

Description

The filter used in the integration has become invalid.

User Action

1. Navigate to the integration filter in error.
2. Resolve each error that appears in the filter.

CCRRTT-1410E – Integration must specify a key identifier.

Description

An integration must specify a key identifier for the given collections. Key identifiers are used to determine how to locate artifacts in a target collection. They do this by specifying the field on the source model that contains the target artifact formatted id.

User Action

1. Navigate to the integration page
2. Select the two collections missing a key identifier
3. Navigate to the field flow page and configure a key identifier

CCRRTT-1411E – All specified routes of an integration must be configured.

Description

All specified routes of an integration must be configured.

User Action

1. Navigate to the integration routing page
2. Configure all routes which require configuration

CCRRTT-1412E – Integration has a conditional route with invalid configuration.

Description

The conditional routing configuration of the integration has become invalid.

User Action

1. Navigate to the integration route in error.
2. Resolve each error that appears in the routing configuration.

CCRRTT-1413E – Collection has invalid repository query.

Description

The repository query used in the collection has become invalid.

User Action

1. Navigate to the collection.
2. Resolve the error by selecting a different repository query.

CCRRTT-10004E – Enterprise Data Stream Integration must have exactly one target SQL Collection.

Description

An Enterprise Data Stream Integration must reference a single SQL collection.

User Action

- Select a SQL Collection for the target of the Integration that is in error.

CCRRTT-10005E – Enterprise Data Stream Integration must have a source Collection.

Description

An Enterprise Data Stream Integration must reference at least one Collection to be used as a source of artifacts.

User Action

Select a source Collection for the Integration that is in error.

CCRRTT-10006E – Enterprise Data Stream Integration target Collection must have appropriate mapping.

Description

An Enterprise Data Stream Integration's data Collection must be mapped to a model. This corresponds to the model desired to be reported on.

User Action

Add mappings for the Collection used in the Enterprise Data Stream Integration.

1. navigate to the Collection
2. add a mapping to a model

CCRRTT-10007E – Enterprise Data Stream Integration source Collection must provide the correct model.

Description

An Enterprise Data Stream Integration source Collection must be mapped to the same model as the target Collection.

User Action

Add relationship to the model for the source Collection used in the Enterprise Data Stream Integration

1. navigate to the Integration
2. identify the model of the target Collection
3. navigate to the source Collection in error, and ensure that its model matches the model of the target Collection
 - if the source collection is a Repository Collection, add a mapping to the corresponding model
 - if the source collection is a Gateway Collection, ensure its model is set to the corresponding model

CCRRTT-10008E – Enterprise Data Stream Integration target Collection must have exactly one project.

Description

An Enterprise Data Stream Integration's Collection must have exactly one project.

User Action

1. Navigate to the Collection
2. Ensure it has exactly one project which corresponds to the database table

CCRRTT-15002E – Integration services cannot be started due to a problem with the license.

Description

Tasktop integration services cannot be started due to a problem with the license. This problem can be caused by running the software without a license, using features that are not included in the installed license, or by having an invalid or expired license.

User Action

This problem can be resolved by installing a valid license using the following steps:

1. Obtain a valid license by contacting the [Tasktop Support Center](#)
2. Navigate to the settings page
3. Press the Apply New License button under License

4. Paste in the license text and press Save

CCRRTT-15005E – Repository cannot be used due to a problem with the license.

Description

The repository connection cannot be used because connections to repositories of this type are not enabled by the license.

User Action

This problem can be resolved by installing a valid license using the following steps:

1. Obtain a valid license by contacting the [Tasktop Support Center](#)
2. Navigate to the settings page
3. Press the Edit button under License
4. Paste in the license text and press Save

CCRRTT-15006E – Integration services cannot be started since the current license has expired.

Description

Tasktop integration services cannot be started because the current license has expired.

User Action

This problem can be resolved by installing a valid license using the following steps:

1. Obtain a valid license by contacting the [Tasktop Support Center](#)
2. Navigate to the settings page
3. Press the Apply New License button under License
4. Paste in the license text and press Save

CCRRTT-15007E – Integration cannot be used with the configured repositories due to a restriction in the license.

Description

An integration cannot be run because it is configured with repository pairs which are invalid under the current license restrictions.

User Action

Perform one of the following:

- Delete the offending integration
- Disable the offending integration

- Update the offending integration to use repository pairs allowed under the current license restrictions

CCRRTT-15008E – Collection cannot be used with the configured person reconciliation extension due to a restriction in the license.

Description

A collection has been configured with a person mapping extension, which is not permitted by the current license.

User Action

Perform one of the following:

- Delete the offending collection
- Remove the person mapping extension from the offending collection

CCRRTT-15009E – Collection cannot be used with the configured state transition extension due to a restriction in the license.

Description

A collection has been configured with a state transition extension, which is not permitted by the current license.

User Action

Perform one of the following:

- Delete the offending collection
- Remove the state transition extension from the offending collection

CCRRTT-15010E – Mapping cannot be used with the configured custom data transformation extension due to a restriction in the license.

Description

A mapping has been configured with a value transformation extension, which is not permitted by the current license.

User Action

Perform one of the following:

- Delete the offending collection
- Remove the value transformation extension from the mapping in the offending collection

CCRRTT-15011E – Your licensed user count has been exceeded.

Description

Your licensed user count has been exceeded.

User Action

Please contact your sales representative.

CCRRTT-16001E – Services cannot be started until Tasktop security has been initialized.

Description

Tasktop integration services cannot be started because secure password storage has not been configured and initialized.

User Action

1. Navigate to the Settings page
2. Specify the Master Password under Secure Password Storage

CCRRTT-16002E – Error initializing password encryption.

Description

Secure password storage requires 256-bit AES encryption which is not available in the Java runtime environment.

User Action

This problem can be resolved by installing the Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy Files in the Java runtime environment. The download is available from [oracle.com](https://www.oracle.com/java/technologies/javase-downloads.html) including a README file with installation instructions.

Alternatively, the unencrypted level of the password store maybe used.

CCRRTT-17001E – Mapping cannot be applied since it is not valid within the current context.

Description

The mapping cannot be applied since the mapping is not valid for the artifacts in the current context.

User Action

1. Determine the source of the problem from the specific error message

2. Either update the mapping to match the artifacts and model in use, or
3. Update the corresponding artifact schema to match the mapping, for example by changing a field type

CCRRTT-17002E – Collection model mapping is invalid.

Description

The collection model mapping is not valid due to inconsistencies between the collection schema, the model schema and the mapping.

User Action

1. Determine the cause of the problem from the specific error message
2. Navigate to the mapping
3. Update the mapping to match the collection and model in use, or
4. Update the corresponding collection artifact schema to match the mapping, for example by changing a field type, or
5. Update the model to match the mapping, for example by adding a field, or changing a field type

CCRRTT-17003E – Artifact could not be created or updated because one or more values cannot be accepted.

Description

An artifact could not be updated or created because one or more of its values are not valid. See the specific error message for details.

User Action

1. Identify the fields and values that are in error from the specific error message
2. Correct the source data, either by
 - updating the source artifact, or
 - by making changes to the mapping, or
 - by making changes to the target system so that the provided data is valid, or
 - by providing a new artifact via a Gateway Collection

CCRRTT-17004W – Artifact cannot be processed since it is currently in use.

Description

Artifact cannot be processed since it is currently in use. This temporary problem occurs when Tasktop attempts to process changes to an artifact concurrently.

User Action

This error will resolve itself automatically, no user action required.

CCRRTT-17005E – Field flow is invalid.

Description

The field flow configuration is not valid due to inconsistencies between the the model schema and the field flow.

User Action

1. Determine the cause of the problem from the specific error message
2. Navigate to the integration
3. Select the collection pair
4. Navigate to the field flow
5. Update the field flow to match the model in use, or
6. Update the model to match the field flow, for example by adding a field

CCRRTT-17006E – Artifact was created but some values could not be set.

Description

An artifact was created by an integration but some values on the artifact could not be set. The resulting artifact has some field values that may not be correct.

User Action

1. Determine the cause from the specific error message
2. Either retry the corresponding activity, or
3. Verify the state of the created artifact and manually adjust values as necessary

CCRRTT-17007E – Conflict resolution strategy is invalid.

Description

The conflict resolution strategy configuration is invalid.

User Action

1. From the integration, navigate to the conflict resolution strategy
2. Select an option for the conflict resolution strategy

CCRRTT-17008E – Artifact could not be processed as it did not meet any of the configured conditions on the Conditional Artifact Routing page.

Description

Artifact could not be processed as it did not meet any of the configured conditions on the Conditional Artifact Routing page.

User Action

- Update the conditions configured on the Conditional Artifact Routing page to ensure the artifact's field value is accounted for, or
- Update fields on the artifact to ensure that it meets the conditions set on the Conditional Artifact Routing page, or
- Update specification for handling artifacts not matched by conditions configured on the Conditional Artifact Routing page to "Ignore" or "Default Route" instead of "Error".

CCRRTT-17009E – Invalid state transition.

Description

An extension provided invalid values when attempting to transition an artifact.

User Action

1. Identify the extension that produced invalid values
2. Identify the fields and values that are in error from the specific error message
3. Modify the extension to produce a valid transition

CCRRTT-17010E – Repeated state transition.

Description

An extension attempted to transition an artifact with the same transition more than once.

User Action

1. Identify the extension from the error message
2. Modify the extension to avoid repeated transitions of the same type for an artifact

CCRRTT-17011E – Extension completed with an error.

Description

An extension completed with an error. See the specific error message for details.

Scripts complete with errors for one of two reasons:

- the extension intentionally raised an error, for example to indicate that a business rule was not satisfied
- the extension itself has an error in its implementation

User Action

1. Determine from the specific error message the cause of the error
2. Either modify the extension to prevent the error from occurring, or
3. Modify the source or target artifact to satisfy the condition that caused the error

CCRRTT-17013E – The state transition requires the selection of model fields.

Description

A state transition extension is configured in a collection that has no model fields selected.

User Action

Either disable the state transition of the collection or select model fields for the state transition.

To select the fields for the state transition:

1. navigate to the collection
2. navigate to the collection state transitions via the "Edit state transition" link
3. add the model fields required by the state transition in "State Transition Fields"

To disable state transitions in the collection:

1. navigate to the collection
2. navigate to the collection state transitions via the "Edit state transition" link
3. select "None" for "State Transition"

CCRRTT-17014E – Relationship values could not be resolved during synchronization.

Description

One or more relationship links could not be resolved as part of a synchronization.

This problem occurs when two artifacts that link to each other are synchronized out of order. This commonly occurs when one artifact (A) links to another (B), but the linked-to artifact B has not yet been synchronized.

When the copy of artifact A (A') is created in the target repository, a link to a copy of B (B') cannot be created at that time since B' has not yet been created.

This problem usually resolves itself once B' is created; the link from A' to B' is created once B' becomes available.

User Action

- None; wait for the error to be resolved automatically, or
- Remove the unresolved link from the artifact being synchronized

CCRRTT-17015E – Relationship values could not be resolved during synchronization.

Description

One or more relationship links could not be resolved as part of a synchronization.

This commonly occurs when one artifact (A) links to another (B), but the linked-to artifact B has more than one corresponding copy in the target repository. This can be caused by having two separate synchronization integrations that cause B to be copied into the target repository.

User Action

- Remove the link from A to B, or
- Remove one of the two synchronization integrations

CRRRTT-17016E – An unexpected error occurred when creating the artifact.

Description

An unexpected error occurred when creating the artifact. The artifact may or may not have been created.

User Action

1. Do not retry the event without guidance from Tasktop Support,
2. Contact the Tasktop Support Center for assistance: "<https://links.tasktop.com/support>"

CRRRTT-17017E – The repository does not support artifact creation.

Description

The repository does not support artifact creation.

User Action

1. Navigate to the corresponding integration,
2. Disable artifact creation flow into the specified collection,
3. Remove all routes flowing into the specified collection.

CRRRTT-17018E – Model does not have all fields required by the state transition.

Description

A state transition extension is configured in a collection that requires fields that are not configured in the model.

User Action

Either remove the missing fields in the state transition configuration, or ensure that the model has the required fields.

To add the fields to the model:

1. navigate to the model
2. add the fields

To change the required fields of the state transition extension from the collection:

1. navigate to the collection
2. navigate to the collection state transitions via the "Edit state transition" link
3. modify the list of model fields

CCRRTT-20000E – No integration is listening to the Gateway Collection.

Description

A Gateway Collection has been used, but the collection is not configured as a source in an integration. The payload has been lost.

User Action

1. Use the Gateway Collection in an integration, or
2. Stop pushing to the collection (from the external source)

CCRRTT-20001E – Time Tracking integration model must have a field of type time entries.

Description

Model used in a Time Tracking integration must have a field of type Time Entries.

User Action

Either

1. Navigate to the model
2. Add a field of type Time Entries

Or

1. Create or select another model having a field of type Time Entries
2. Ensure that each collection used in the integration is using the selected model

CCRRTT-20002E – Time Tracking integration Collection must have a field mapping to a field of type time entries in the Model.

Description

Collections used in a Time Tracking integration must have a field mapped to the model Time Entries field.

User Action

1. Navigate to the collection model mapping
2. Add a field mapping to the model Time Entries field

CCRRTT-20003W – Time Tracking integration target Collection does support impersonation of the Worker field.

Description

The selected collection does not support worklog impersonation and so has limited use as the target in a Time Tracking integration.
The worklogs will be filed under the user of the target repository connection.

CCRRTT-20004E – Relationship fields of a Gateway Collection must be configured to specify the related repository.

Description

A Gateway Collection must configure the Relationship(s) fields to associate them with the repository having referenced artifacts.

User Action

1. Navigate to the Gateway collection
2. Locate the "Relationship Field Configuration" section in the UI
3. For each field, select the repository that is associated with that relationship.

CCRRTT-20005E – Gateway collection must have a model.

Description

A Gateway Collection must have a model configured.

User Action

1. Navigate to the Gateway collection
2. Select a model and save the changes

CCRRTT-20006E – Gateway Collection cannot be used with the configured payload transformation extension due to a restriction in the license.

Description

A gateway collection has been configured with a payload transformation extension, which is not permitted by the current license.

User Action

Perform one of the following:

- Delete the offending gateway collection
- Remove the payload transformation extension from the offending gateway collection

CCRRTT-30000E – An unexpected error occurred.

Description

An unexpected error has occurred. Check the specific error message for details.

User Action

Check the specific error message for details of the failure. If possible correct the problem described in the error message, or contact your administrator for assistance.

CCRRTT-30001E – Not found.

Description

The entity was not found because the entity no longer exists on the server.

User Action

Ensure that the provided entity id is correct, and if not correct the id and try again.

CCRRTT-30002E – The data provided was not valid.

Description

The data provided was not valid. See the specific error message for details.

User Action

Correct the problem described in the specific error message and try again.

CCRRTT-30003E – The connector kind was not found.

Description

The connector kind was not found.

User Action

Ensure that the connector kind is specified correctly and try again.

CCRRTT-30004E – The request entity was not valid JSON.

Description

The request entity was not valid JSON.

User Action

Ensure that the request payload is formatted as a valid JSON entity and try again.

CCRRTT-30005E – Secure password storage must be initialized.

Description

Secure password storage has not been initialized.

User Action

Configure secure password storage via the settings page.

CCRRTT-30006E – Error communicating with {0} repository.

Description

Error connecting to repository. See the specific error message for details.

User Action

Check the specific error message for details of the failure. If possible correct the problem described in the error message, or contact your administrator for assistance.

CCRRTT-30007E – Error processing request MIME attachment.

Description

The request MIME attachment could not be accepted either due to a bad request or an I/O failure.

This problem can be caused by insufficient disk space or lack of write permissions in the Tasktop application temporary directory.

User Action

1. Verify that the temporary directory of the Tasktop application is writable,
 - The Tasktop application must have write permissions to the directory
 - The directory must have sufficient available space
2. Try again

CCRRTT-30008E – Tasktop is stopped, see the Activity View and error log for more details.

Description

Tasktop has been stopped due to unrecoverable errors. See error log for more details.

User Action

Correct the problem described in the specific error message and restart.

CCRRTT-30009E – The database is not available.

Description

The configuration database is unavailable.

User Action

Ensure the configuration database is online and can be reached and ensure Tasktop's database settings are correct.

CCRRTT-30010E – Connection settings are not valid.

Description

The provided connection settings are not valid. See the specific error message for details.

User Action

Correct the problem described in the specific error message and try again.

CCRRTT-30011E – The database is locked for maintenance and cannot currently be used.

Description

The configuration database is locked for maintenance and cannot be used.

User Action

Wait for the ongoing maintenance to complete.

CCRRTT-50001E – Unable to propagate artifact changes since the target artifact has been removed.

Description

Changes to an artifact cannot be propagated to the corresponding artifact in the alternate repository of a synchronization integration since the target artifact has been removed.

User Action

- Use the "Recreate Artifact" action to have Tasktop recreate the artifact that was deleted in the end system and associate it with the still-existing artifact in the other repository (putting them in sync

- with one another), or
- Delete the associated artifact, or
- Move the associated artifact out of its collection such that the artifact is no longer synchronized, or
- Apply an artifact filter to ensure updates to the artifact will not be synchronized. To do so, make sure the artifact does not meet the filter criteria specified and make sure to configure the filter to apply to artifact updates

CCRRTT-50002E – A conflict has occurred during synchronization.

Description

A field conflict was detected when synchronizing artifacts. A field conflict occurs when the value of a field that is set to flow bidirectionally conflicts across your repositories.

The synchronization of these artifacts was halted with an error because a conflict resolution strategy of "Error Upon Conflict" was configured and the system was unable to propagate the value from either artifact without overwriting a change from the other artifact.

User Action

- Change the conflict resolution strategy to have one of the repositories dominate in case of a conflict, or
- Manually change the conflicting value on at least one of the artifacts such that there is no longer a conflict, or
- Change the field flow of the affected field to be unidirectional (in which case a conflict is not possible)

CCRRTT-50005E – A conflict has occurred during synchronization.

Description

A conflict was detected when synchronizing artifact containment. A conflict occurs when one or more containers of synchronized artifacts is changed for both artifacts.

User Action

- Change the container of one or both artifacts to its original value