

1. User Guide ..... 2

2. Installation Guide ..... 158

3. Installing Sync as a Windows Service ..... 167

# User Guide



Tasktop Sync allows teams to synchronize tasks across different defect and task tracking repositories. The Tasktop Sync server application places itself in between repositories to synchronize changes made to the defects or tasks in either repository.

This user guide explains how to set up and maintain a Tasktop Sync installation, including details on how to configure and run the software.

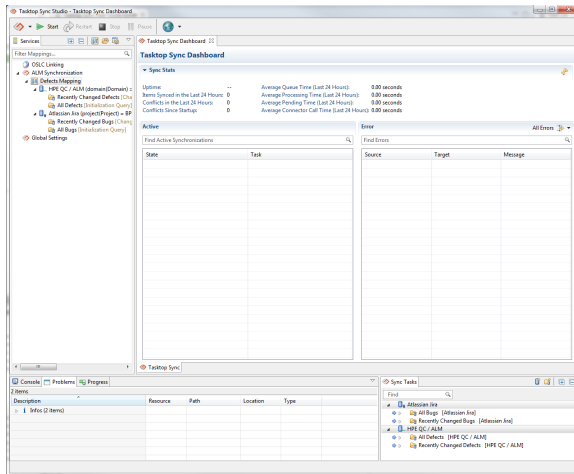
Copyright © 2012, 2013, 2014, 2015, 2016, 2017, 2018, 2019 Tasktop Technologies  
Covered by one or more of the following: US Patent No. 9,459,839 and US Patent No. 9,342,512.

## Introduction

For organizations needing to connect development, QA, and project management, Tasktop Sync provides the only enterprise-scale ALM middleware solution, built on the industry-standard Eclipse Mylyn ALM integration framework. Tasktop Sync provides two distinct capabilities: Task Linking and Task Synchronizing. In addition, Tasktop Sync also provides a Quick Start to help configure repositories.

Unlike previous approaches to ALM synchronization, Tasktop Sync provides real-time synchronization, automated conflict resolution, and support for over two dozen ALM tools. Building on Tasktop's Task Federation™ technology, Tasktop Sync's synchronization component ensures that each stakeholder has access to the data that they need within their tool of choice, even if the data resides across requirements management, Agile development, and traditional quality management systems. The [Task Synchronization Configuration](#) section explains in detail how to configure the Task Synchronization feature.

For those who want a lighter-weight solution, Tasktop Sync's Task Linking enables traceability by letting you find and link related tasks across repositories with minimal configuration. Task Linking is built on top of Open Services for Lifecycle Collaboration (OSLC) protocols, an industry-standard means for communicating between ALM components. Tasktop Sync supports the service provider, picker, links, and preview features of the OSLC Core 2.0 standard. This means that Task Linking allows users to select, link to, and preview related tasks in a foreign repository from a OSLC consumer's system. Not only can Task Linking easily communicate with OSLC consumers (like IBM(Rational Business Machines)© Rational Team Concert™ Rational DOORS Next Gen and IBM Rational Team Concert Change and Configuration Management), Tasktop Sync connectors can turn HPE QC / ALM, JIRA, and TFS repositories into OSLC producers. The [Task Linking Configuration](#) section explains how to configure the Task Linking feature.



## Terminology

Tasktop connects to many different products, and different vendors and products have very different terms for similar concepts. It is important to understand these terms to make sense of the rest of this manual.

**Task -;** A task is an item of work that someone needs to act upon. In other products, what we call "tasks" may be called "tasks", "defects", "bugs", "feature requests", "work items", "test cases", "trouble tickets", "records", etc. This can be confusing. For example, IBM Rational Team Concert manages "tasks", "work items", and "defects", all of which we call "tasks" in this document.

**Repository -;** A repository (or task repository) is a collection of tasks. In other products, what we call "repositories" or "task repositories" are also called "bug databases", "bug trackers", "issue tracking systems", "incident tracking systems", etc.

**Proxy Task -;** When Tasktop Sync synchronizes tasks between repositories, a proxy relationship between synchronized tasks is defined. A proxy task is the task (say, Task O) in the other repository that is synchronized to the current task (say Task C). Conversely, Task C is the proxy task of Task O.

**Task attribute -;** A task attribute is an aspect of a task that is stored in a repository. "Date" and "Priority" are common task attributes. In other products, what we call "task attributes" are also called "fields", "columns", etc.

**Task attribute values -;** A task attribute value is the specific value that a task attribute may have. For example, "March 14, 1:59am 2011" might be a task attribute value for the task attribute of "Date". "P1" might be a task attribute value for the task attribute of "Priority".

**Connector -;** A connector is the software that connects Tasktop Sync to a repository. Different vendors' repositories use different connectors.

**Workflow** -; In some Repositories, a task's status represents a stage in the task's workflow lifecycle. This workflow is configured within the repository, and links together statuses that a Task may transition between. For example, a Task with the status "To Do" may only be allowed to transition to the "In Progress" or "Rejected" status, and not the "Done" status.

**Transitions** -; In workflow management, a task's status cannot be updated directly, but must be changed using a transition. This transition is a repository operation that must be executed to perform the change to the task. Transitions are defined when configuring the workflow in the repository, and normally include the starting status and the ending status. Examples of transitions might include "Complete Task", or "Start Working".

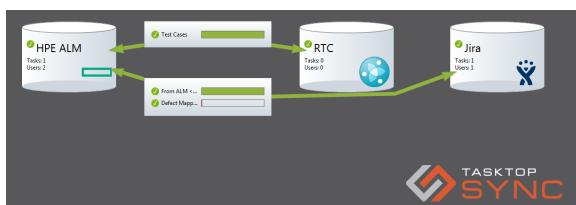
**Transition attributes** -; When executing a transition, some task attributes can, or must, be updated at the same time. For instance, it is common to include a **Resolution** value when executing a "Complete Task" transition. The transition attributes are defined in the repository workflow, and can be configured in Sync using the [Transition Attribute Mappings Table](#).

**Repository back-off** -; When defining repository connections, you must provide authentication credentials which will be used by Sync when communicating with the repository. If these credentials become incorrect, the communication will start failing. Some repositories will disable users that perform too many actions with invalid credentials. To prevent this, when Sync detects authentication errors it will institute a back-off, freezing all activity with the repository for a period of time.

## Features

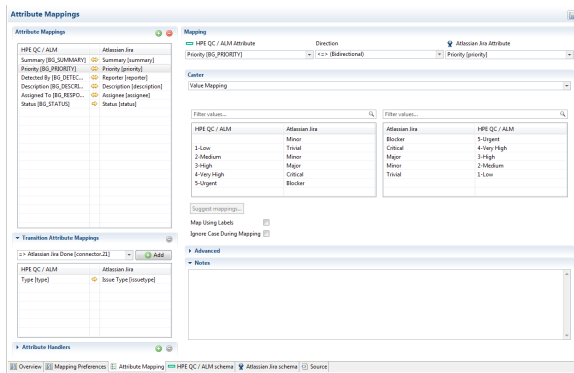
### Integration Visualizer

The integration visualizer provides the user with a broad overview of the ALM architecture configured in Tasktop Sync. This view not only allows the overall ALM architecture to be understood quickly, but it also provides an easily understood snapshot of the health of the architecture by tracking and display synchronization history and highlighting problems between different parts of the architecture.



### Task Mapping Editor

Tasktop Sync provides a comprehensive UI for configuring synchronization mappings between pairs of repositories and tasks. The task mapping editor allows users to quickly define new mappings, as well as add to and modify existing mappings.



## Automated Conflict Resolution

Tasktop Sync automatically handles conflicts in an intelligent fashion.

Conflicts can occur when two synchronized tasks are updated at the same time in each repository. If two users are collaborating on the same task and both decide to update it at the same time, they might submit incompatible changes. For example, one user could change the priority of the task to "High" while the other user changes it to "Medium". Most repositories provide built-in support for handling this case, but this support only works for tasks being accessed on the same repository. In a synchronized environment, the tasks can exist in different repositories; Tasktop Sync is in a position to handle conflicts that occur across repositories.

## Person Mappings

Heterogeneous ALM repositories are often accessed by different groups, with varying degrees of overlap. Even in the cases where the same user is accessing both systems being synchronized, the user might have different user IDs in each system. For example, a user might be called janesmith in one repository and jsmith in another. Synchronizing data between repositories involves translating these IDs between repositories so that identities are preserved.

Tasktop Sync provides a pluggable system for mapping person IDs. For simple person ID mappings, Tasktop Sync can be pointed at a file containing the mappings between person IDs. This mapping may contain direct ID-to-ID mappings as well as configuration for more sophisticated person mapping, including mapping using person metadata, and mapping using scripts. Additionally a default person ID can be provided on each repository that is used if a user exists in one system but not the other. For further details see [person mapping configuration](#)

## No Local State Lock-in

Tasktop Sync stores all important task-related data in your existing task repositories. There is no information stored in Tasktop Sync that is not retrievable from one of the repositories participating in synchronization. This means that utilizing Tasktop Sync does not cause your data to be stored in a new proprietary format, nor do you need to back up task data from the machine running Tasktop Sync.

## Copied Task Handling

Tasktop Sync handles synchronizing copied tasks in a way that eliminates the problematic scenario with multiple tasks having the same proxy link. Copied tasks will be treated as if they were new tasks; their incorrect proxy association will be removed, and then a new proxy task will be created.

## Automatic Proxy Link Repair

When enabled, Tasktop Sync supports the recovery of deleted or missing proxy links during synchronization. Tasktop Sync keeps a database of past synchronizations and will recover and rewrite the previous proxy associations. For example, this allows successful synchronizations of HPE ALM tasks that have reverted to a version before their proxy associations were stored.

## Configuration Templates

Configuration templates allow administrators of Tasktop Sync to easily expand and manage the use of Tasktop Sync at their organization by allowing them to apply the same configurations to multiple workspaces and projects. With this feature, administrators can:

- [Create a template from an existing task mapping](#)
- [Create new task mappings from a template](#)
- [Monitor template usage](#)
- [Change a template used by multiple task mappings](#)
- [Disconnect task mappings from templates](#)
- [Override a template](#)

## Task Relationship Management

ALM repositories can store relationships between their tasks. Normally, these relationships reflect a link between two different types of tasks, such as a test case linked with a defect found during execution of the test. Relationships may also exist amongst tasks of the same type, such as parent and child requirements. Tasktop Sync can synchronize these relationships, maintaining the links and hierarchy in two synchronized repositories.

## Synchronizing External Associations

Tasktop Sync handles synchronizing references to related tasks as web URLs. For example, when a defect is synchronized between your quality assurance repository and your engineering repository, a reference to a test can be synchronized as a URL, enabling your engineers to see the test in its original repository with a simple mouse click.

## Synchronizing Attachments

When enabled, Tasktop Sync can synchronize attachments between ALM repositories. Along with the attachment, certain attachment metadata may also be synchronized, including its description, author and title.

To synchronize attachments efficiently, Tasktop Sync must be able to compare attachments between repositories without downloading the entire binary. To do this, Tasktop Sync will use the filename and /or file size to identify the attachment. Since not all repositories provide access to both a filename and file size, Tasktop Sync will use the fields that are supported by both repositories to compare. So, for example, if one repository provides both filename and file size, yet the second provides only filename, Tasktop Sync will be forced to only use filename when comparing the attachments. If there are no overlapping fields provided, then Tasktop Sync cannot synchronize attachments between the repositories.

For details on configuring Attachment synchronization, see [Mapping Preferences](#).

## Synchronizing Comment and Attachment Authors

Tasktop Sync attempts to synchronize the authors of comments and attachments. If a [person mapping](#) is defined, it will be used to determine the author of the comment or attachment in the target repository. Tasktop Sync can optionally validate whether the authors of comments and attachments exist in the target repository before submission.

Not all systems support synchronization of comment or attachment authors. When synchronizing comments and attachments to systems which do not support synchronization of authors, the author will appear as the user defined in the repository connection. Tasktop Sync can also be configured to add the comment author's name to the synchronized comment's text. This feature can be useful to enable dialog across tasks in separate repositories.

## Status Workflow Management

Many ALM repositories support the notion of a task's status or state. For instance, a Defect task may have a set of predefined statuses: New, In Progress and Done. In some ALM repositories, changing a task's status cannot be done directly, but must be changed through executing a **transition**. This is commonly referred to as Workflow Management, and can be configured within the ALM repository.

Tasktop Sync can execute transitions using a **Status Transition Caster**. Details on how to configure the caster are [here](#).

## Architecture

This section describes the overall architecture of Tasktop Sync. Understanding the [fundamentals](#) section makes Tasktop Sync configuration easier. The following sections are more advanced discussions of the architecture of some of the critical features of Tasktop Sync.

# Fundamentals

At its core, Tasktop Sync maintains synchronization between a task in one repository, say Task A, with a task in another repository, say Task 1. In this case, Task 1 is called the "proxy task" of Task A, and vice versa.

The architecture of Tasktop Sync involves pairs of connectors to task repositories. These connectors provide a common interface to Tasktop Sync for accessing tasks from each repository. Tasktop Sync then uses these connectors in its core to synchronize tasks between the two connectors. This modular architecture is what enables Tasktop Sync to easily work with many different vendors' repositories and create general mappings between any pairs of these supported repositories.

The tasks which are synchronized between pairs of repositories are scoped first by queries, and then by matching a set of field values. The queries allow Tasktop Sync to bound the set of tasks which should be synchronized between repositories, but they also allow Tasktop Sync to build a local cache of task data to improve both throughput and latency of task synchronization, and to resolve conflicts between tasks at a fine-grained level.

Tasktop Sync uses these queries to issue requests to each repository in order to identify tasks which have changed and require synchronization. Synchronizations can be batched by varying the delay which Tasktop Sync uses between checking its queries for changed tasks.

Tasktop Sync is optimized to minimize latency, maximize throughput, and minimize load on your repositories.

- We have worked closely with our ALM partners to ensure that Tasktop Certified connectors /repositories are fully incremental: when issuing a query, Tasktop Certified connectors retrieve only the tasks which have changed *since the last time the query was issued*.
- Tasktop Sync maintains an internal cache of tasks, and only synchronizes when the new task differs from the old task on a configured attribute. If a task only changes in an attribute that is not part of the synchronization mapping, the process stops as soon the task is compared against the cache.
- Because Tasktop Sync synchronizes at an attribute level, non-conflicting changes propagate without conflicts, reducing the number of later synchronizations.

Together, this means that the synchronization speed does not depend upon the size of your repository or the number of users, only on the frequency of meaningful changes.

## Synchronization Conflict Handling

### Conflict Detection



The first step in handling conflicts is to detect them. Note that there's a difference between two tasks being out of sync with each other and being in conflict with each other. If an attribute of the two tasks is different because one of the tasks has been updated while the other wasn't, that is not a conflict. If the attributes are different because they have both been updated at the same time, that's a conflict. Furthermore, if an attribute of both tasks has been updated at the same time with the same change then they have in effect been synchronized by the users, and there is no conflict; a synchronization action would be redundant. If two tasks have changed, but the attributes which changed were different in the two tasks, again, there is not actually a conflict. For example, if one person changed the Severity of a task to "BLOCKER", while another person changed the owner to "mgarcia", this would not be a conflict.

If a synchronizer only has the information that two tasks were updated at the same time, but not information about which attributes caused the tasks to conflict, then that synchronizer will not be able to recognize non-conflicting attribute changes. Tasktop Sync, however, can detect which attributes have changed, and thus provides built-in support for attribute-level conflict detection.

## Conflict Resolution

Once a conflict has been detected, the next step is to decide what to do about the conflict. This involves selecting one of the changes as the new value of the attribute, thus overwriting the other value, or deciding that the sync is not allowed to go through at all, thus leaving both values in place. Note that with attribute-level conflict detection, only changes to the same attribute are declared as conflicts and chosen to be resolved. Non-conflicting changes to different attributes are merged and both tasks are updated with the other's changes.

Tasktop Sync has the following policies available for resolving conflicts:

- **A Dominates B:** Tasks from repository A will always overwrite conflicting data on tasks from repository B.
- **No Sync:** If a conflict is detected then an error is reported and the tasks are not synced. A user or administrator must manually resolve the conflict. This policy is mostly intended for catching configuration errors when a conflict is not expected to occur.

Conflict resolution policies can be defined at both the task level and the attribute level in the configuration file. Defining a conflict resolution policy at the task level applies that policy to every attribute mapping defined in that task's mapping that does not have its own resolution policy. Conflict resolution policies specified in an individual attribute mapping apply only to that attribute and override the policy specified in the containing task mapping.

## Conflict Notification

Once a conflict has been detected and resolved, it's often useful for users to be notified of the conflict and what action was taken to resolve it. At a minimum, Tasktop Sync reports all conflicts in a log file available to Tasktop Sync administrators. However, Tasktop Sync administrators are often not in the best position to act on this information, as they may not be involved in the task on which the conflict occurred. It's often desirable for the users who are subscribed to the task to be notified of the conflict so they can confirm that the action taken was appropriate, or at least be aware that two users were posting conflicting data to a task.

Tasktop Sync provides a conflict notification policy whereby the conflicts can be logged in the comment stream of the task in question. The comment indicates which attributes were involved in the conflict and which values were overwritten according to the conflict resolution policy.

## Synchronization States

A synchronization is defined as process of propagating a set of changes between two tasks. Each time Tasktop Sync detects changes on a mapped task, a new synchronization is scheduled.

The synchronization process runs through a set of states as changes are propagated between repositories. Many of these states are visible through the [Tasktop Sync Dashboard](#). The synchronization will progress through each state, and depending on the results accrued during processing, move onto another until the Done state is reached. Each state is described below.

### Queued

When a change is detected in a mapped repository task, or a new task that should be mapped is detected, a new synchronization is created and placed in the queue. At this stage, the synchronization is in Queued state, waiting to be processed.

A synchronization can remain in Queued state due to a number of restrictions, such as:

- The task(s) of the new synchronization are already involved in an existing synchronization.
- There is already the maximum number of concurrent synchronizations active.

The synchronization will remain in Queued state until no more restrictions exist. From Queued state, a synchronization can enter Error state, if an error is encountered, or Processing state. If Tasktop Sync is stopped, any synchronization in queued state will remain there until Tasktop Sync is restarted.

### Processing

A synchronization is in Processing state while the actual synchronization of tasks is being performed. During Processing, both repositories are contacted, and changes to the tasks are propagated.

Normally, at the end of the Processing state, the synchronization enters Done state and is complete. If an error occurs, it will enter Error state.

A synchronization can also enter Pending state if there are more changes to synchronize. See [Pending](#) for details. If Tasktop Sync is stopped, any synchronizations in Processing state will be completed.

## Error

When a synchronization encounters an error it cannot automatically recover from, it moves the synchronization to Error state. These synchronizations are visible under the [Tasktop Sync Dashboard](#).

The synchronization will remain in Error state until either a new change is detected in the task(s), returning the synchronization to Queued state, or the Tasktop Sync administrator manual forces the state to change. This can be done by right-clicking on the synchronization in the Tasktop Sync Dashboard, and choosing one of the following:

- Queue for Synchronization: Send the synchronization back to Queued state.
- Clear Selected Errors: Clear the error and move to Done state.

More details in these steps are located under the [Tasktop Sync Dashboard](#).

A synchronization in Error state will remain in this state even when Tasktop Sync is stopped.

## Pending

During Processing state, if there are more changes known to be propagated, the synchronization is moved to Pending state. This condition can happen for a number of reasons:

- There is a recoverable network error.
- A linked task does not yet exist in the target repository.
- It will take multiple submissions to propagate all the changes.

The synchronization will remain in Pending state for approximately one minute, and then moved back to Queued state to start the synchronization again. If the condition(s) that triggered the original Pending state still exists, the synchronization will then re-enter Pending State. This can happen up to 60 times, at which point the synchronization will enter Error state.

If Tasktop Sync is stopped, any synchronizations in Pending state will be returned to Queued state when Tasktop Sync is restarted.

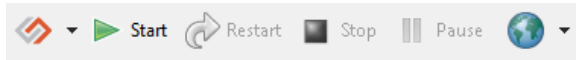
## Done

When a synchronization has finished propagating all the mapped changes across repositories, the synchronization is Done. At this stage, the synchronization will no longer appear in the Tasktop Sync Dashboard, and no more work will be scheduled.

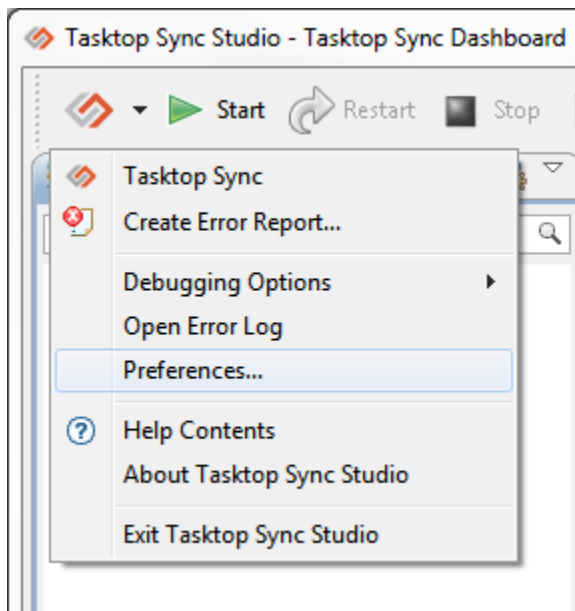
# The Tasktop Sync Interface

This section explains the layout of Tasktop Sync and how you can use the provided views to manipulate Tasktop Sync. To configure Tasktop Sync please see the [configuration](#) section.

## The Tasktop Sync Toolbar



Along the very top of the Dashboard window is the Tasktop Sync toolbar which provides ways to control and configure Tasktop Sync.



The far left "Tasktop Sync" button provides access to general Tasktop Sync actions.

These action include:

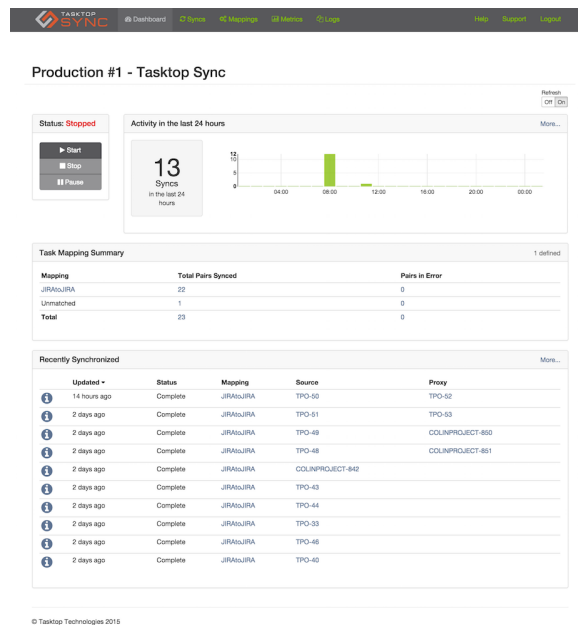
- Opening the Tasktop Sync Dashboard
- Creating an Error Report
- Changing the amount of logging performed during synchronization
- Opening the Error Log
- Opening Tasktop Sync's preferences
- Opening Tasktop Sync's help documentation
- Opening Tasktop Sync's About page
- Exiting Tasktop Sync

Immediately to the right of the Tasktop Sync button are buttons to start, restart, stop and pause /resume Tasktop Sync. When the pause button is clicked, Tasktop Sync will finish any operations that have already been started, but will not begin any new processing.

On the far right is a drop-down menu with controls for Tasktop Sync's web server. From here the web server may be started, stopped or restarted. Additionally the configuration for Tasktop Sync's web server and OSLC adapter may be opened.

## The Tasktop Sync Dashboard

In the main editor view is the Tasktop Sync Dashboard itself.



## Dashboard Statistics

At the top of the Tasktop Sync Dashboard is the statistics section, which displays the following statistics:

- **Average Queue Time:** The average time that a task spends in a processing queue. This is the time from the moment the task gets detected to the moment it starts being processed.
- **Average Processing Time:** The average time it takes to synchronize a task. This is counted from the moment when the task starts being processed to the moment that the synchronization ends.
- **Average Pending Time:** The average time that a task spends in a pending queue. This is calculated across all tasks.
- **Average Connector Call Time:** The average time that Tasktop Sync spends communicating with repositories during a single synchronization. This includes fetching the tasks and submitting them.

## The Active and Error Queues

Below the statistics section are the active and error queues. The active queue displays all of the active synchronizations. The error queue displays all synchronizations that didn't complete successfully and require manual intervention. Right-clicking on any of the entries in the error queue and selecting "Open Source Data" or "Open Target Data" will display the attributes of the source or target task, respectively, as well as the values associated with those attributes. Double-clicking on an entry in the error queue will open the [Synchronization Log Viewer](#) of that task, which provides more detailed information about the synchronization history of the task. Both the list of task attributes and the Synchronization Log may be helpful in determining why a synchronization failed. Entries in the error queue can be manually retried by right-clicking on the entry and selecting "Queue for Synchronization". Otherwise, the synchronizer will not automatically retry the synchronization until one of the participating tasks changes. Both the active queue and the error queue may be searched using the provided search boxes.

Active	
Find Active Synchronizations	
State	Task
Processing	http://example.com/qcbi...
Pending	http://example.com/qcbi...
Queued	http://example.com/qcbi...

? Unknown Attachment

## Error Queue Filters

In addition to using the search box, the error queue can also be filtered by task mapping by using the drop-down menu located at the top right-hand corner of the queue:

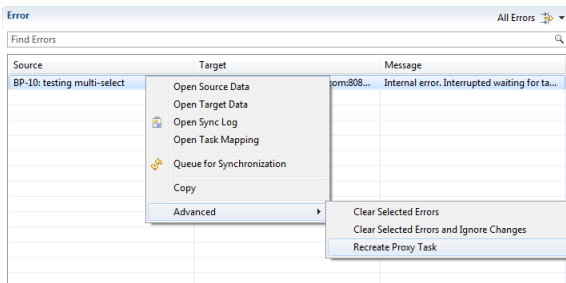
Error		
Find Errors		
Source	Target	Message
168347: test	DEF42: test	Internal error
171538: D3		Configuration error Could n...
208818: One last test		Sync Connector related erro...
208820: Sync with QC		Sync Connector related erro...
208824: One last bug for SY...		Sync Connector related erro...

When a mapping is selected, errors that did not originate from that mapping will not be shown. Errors that cannot be matched to a specific mapping will still be displayed, but will be shaded gray to indicate that they are unmatched. The selected filter can be toggled off and on by clicking on the filter icon next to the drop-down menu.

Error		
Find Errors		
Source	Target	Message
171538: D3		Configuration error Could n...
208818: One last test		Sync Connector related erro...
208820: Sync with QC		Sync Connector related erro...
208824: One last bug for SYN...		Sync Connector related erro...
209400: What happens when ...		Sync Connector related erro...
DEF24: test 6/10		Internal error {0}
DEF34: And again		Configuration error Found ...

## Proxy Task Recreation

If one of the tasks in a synchronized pair has been deleted, the other task will end up in the error queue the next time it is synchronized. This occurs because Tasktop Sync is unable to locate the changed task's proxy task. Tasktop Sync will not automatically recreate deleted proxy tasks, however this can be accomplished manually. Right click on the task in the error view whose proxy task has been deleted, and select "Recreate Proxy Task" under the "Advanced" sub-menu.



Tasktop Sync will prompt you to confirm the proxy recreation. Once confirmed the selected task will have its proxy task recreated at the next opportunity.

## Ignoring New Changes for Selected Errors

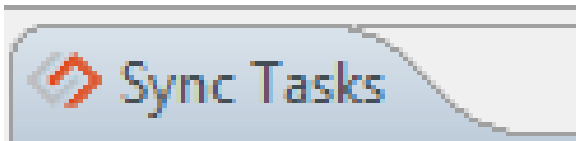
A synchronized pair of tasks in the error queue can be set so that their future changes are **ignored**. To do this, right click on the task pair in the error view, and select "Clear Selected Errors and Ignore Changes" under the "Advanced" sub-menu. The selected pair will also be cleared from the error queue.

## The Diagnostic Views

Below the Tasktop Sync Dashboard is a panel with several views:

- **Console:** This view shows the detailed log output from Tasktop Sync while it is running. It is useful for monitoring the work Tasktop Sync is performing in real time, as well as diagnosing problems that have occurred during synchronization. (Note that all of the messages in this view are saved by default in the [Tasktop Sync Log](#) on disk.)
- **Problems:** This view shows a list of errors and warnings that relate to your configuration of Tasktop Sync, which gives you an overview of problems with your current configuration.
- **Progress:** This view shows a list of operations currently being executed by Tasktop Sync with an indication of the progress of each operation.
- **Error Log:** This view shows real-time messages at a higher level than the console view. Messages shown in this view will be of higher importance than those in the console view and will typically indicate that there have been errors during synchronization. This view can be opened by selecting "Open Error Log" from the Tasktop Sync button in Tasktop Sync's toolbar.

## The Sync Tasks View



The Sync Tasks view lists all of the defined repository connections in Tasktop Sync, along with the queries associated with each. Each query can be expanded to list all of the tasks it returns. Note that the presence of a repository in this view does not necessarily mean it participates in synchronization; you must also configure mappings for these repositories if you want tasks to synchronize.

Along the top of the view are buttons to create a new [repository connection](#) , or a [new query](#) , as well as buttons to expand or collapse the contents of the view.

Many actions can be triggered from the Sync Tasks view by right clicking on repositories, queries, or tasks.

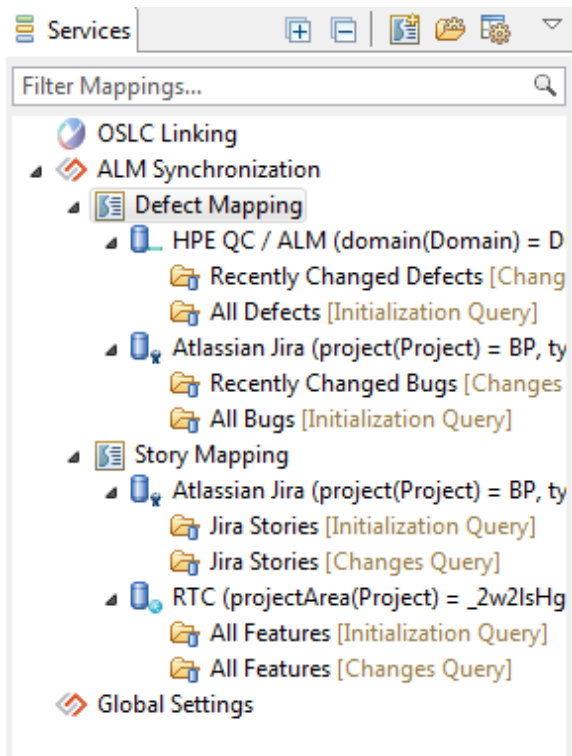
Repositories may have their schemas regenerated, or their configuration updated by right clicking on them in the Sync Tasks view. For more details on repository configuration and schemas see [Repository Configuration Changes](#). Additionally a repository may have its [URL changed](#) by right clicking on them in the Sync Tasks view.

Tasks and their synchronization history may be inspected from this view. Right-clicking on any task in this list and selecting "Open Synchronization Log" will open the [Synchronization Log Viewer](#) , which displays the synchronization history of the selected task. Double-clicking any task in the list, or right-clicking on a task and selecting "Open Task Data", will display the attributes of the selected task and the values associated with those attributes. Both the list of task attributes and the Synchronization Log may be helpful in determining why a particular task is or is not synchronizing.

## The Services View

Finally, on the left of the screen, there is the Tasktop Sync **Services** View. This is where you will see a summary of the repositories and queries you have configured to actively participate in linking and synchronization. This list may be different than the lists in the Sync Tasks views if you have not configured Tasktop Sync to make use of all the repositories or queries you have defined.





## The Services Toolbar

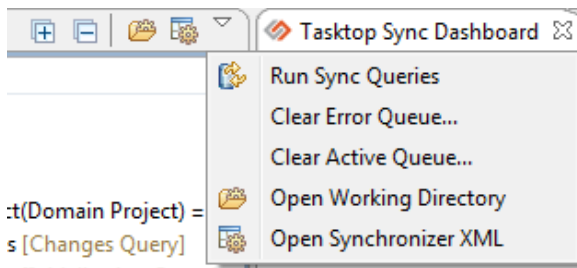
Along the top of the Service View is the toolbar. It contains the actions "Expand All" and "Collapse All", which expand or collapse all of the nodes in the Services View, respectively. Individual nodes are expanded or collapsed by clicking the triangle to the left of each node, or right-clicking the node and clicking either "Expand" or "Collapse". The toolbar also contains the actions "Open Working Directory", which opens a file explorer pointing to the Tasktop Sync working directory, and "Open Synchronizer XML", which opens a text editor containing the `synchronizer.xml` file.

## The Services Filter

Directly below the toolbar is the "Filter Mappings..." text box. If you have a large number of task mappings or templates, you can filter out any that do not contain your search term. Entering text into this box will hide any mappings or templates whose name, repository connector kind, repository URL or repository label do not match your search. For detailed searches, this filter also supports regular expressions.

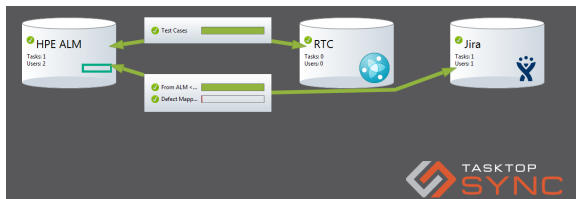
## The Services Menu

Along the far right of the toolbar is the drop-down Services Menu, which contains a few more advanced actions. The "Run Sync Queries" action forces Tasktop Sync to run all repositories' changes queries. The "Clear Error Queue" action sets the state of all entries in the error queue to Aborted, and the "Clear Active Queue" action sets the state of all synchronizations that are currently active or pending to Aborted. Since these synchronizations are set to Aborted, they will not be retried until a change is made to the source or the proxy task. It is also possible to open the working directory and the configuration file from the Services Menu by clicking on the "Open Working Directory" and "Open Synchronizer XML" icons, respectively.

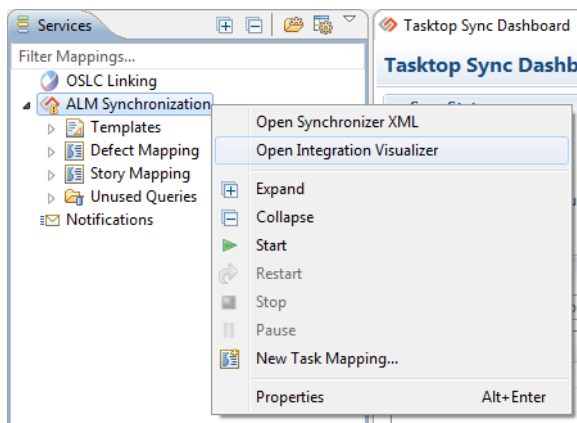


## Integration Visualizer

The Integration Visualizer provides a compact view of how sync mappings unify and synchronize the activities between multiple repositories. The various nodes and links between them are automatically generated per sync configuration and will expand as your ALM architecture evolves.



To access the Integration Visualizer, right-click "ALM Synchronization" in the Services view and select "Open Integration Visualizer".



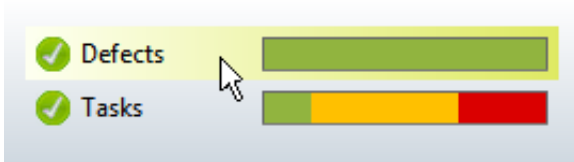
## Repository Node



Repositories are represented by their cylinders in the Integration Visualizer. Double-clicking the cylinder will open the Repository configuration wizard, which allows the editing of information regarding that repository.

Displayed on the node are statistics describing the synchronized tasks and users of a repository. The "Tasks" statistics displays the number of tasks that have had synchronizations completed, are being processed, have errored out, or have been aborted by all mappings connected to that repository. The "Users" statistics describes all users that have reported, been assigned to, or have posted comments or attachments to synced tasks from that repository.

## Task Mappings Node



Between pairs of repositories with mappings, the task mapping node displays all the mappings in which the repositories are related. Double-clicking on any task mapping will open that mapping's [Task Mapping Editor](#).

The different colours on the bars for each task mapping represent the ratios of the states that synchronizations of that mapping ended in.

Colour of Task Mapping Bar	Interpretation
Green	Successful completed synchronizations.
Yellow	Synchronizations that have been manually aborted.
Red	Synchronizations that resulted in an error and are thus unsuccessful.

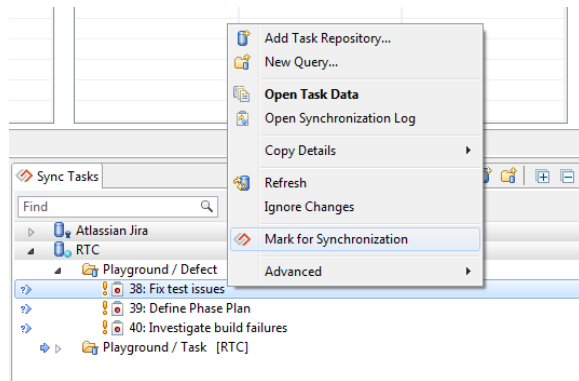
## Starting the Synchronizer

From the Tasktop Sync toolbar or tray icon's context menu, start the Synchronizer by pressing the Start Button in the toolbar. Tasktop Sync will begin synchronizing tasks between configured repositories.

The Synchronizer maintains a persistent store of all tasks that have incoming changes that have not yet been synchronized (including those with errors). When the server is restarted it will reload the queue and continue where it left off.

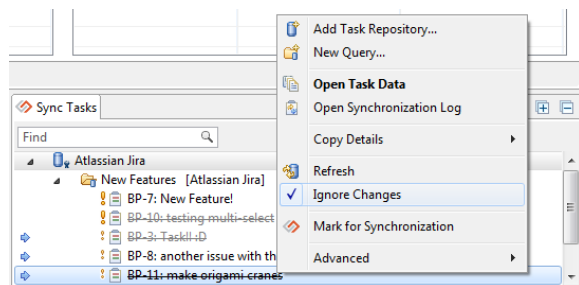
## Marking for Synchronization

While setting up Tasktop Sync mappings and during troubleshooting, it can be useful to mark one or more selected task for synchronization. To do this, open the Sync Tasks view, select one or more tasks, right-click, and select "Mark for Synchronization". This will cause the selected tasks to be added to the Tasktop Sync active queue for processing.



## Ignoring New Changes

Specific, unwanted tasks can be configured so that their changes will not be synchronized. To do this, open the Sync Tasks view, select one or more tasks, right-click, and check "Ignore Changes". The selected tasks and their proxies will now have their changes ignored during synchronization. Unchecking the option will allow previously ignored tasks to be synchronized and will perform a [Mark for Synchronization](#). Ignored tasks have a strikethrough and are greyed out.



## Synchronizing Attachments Again

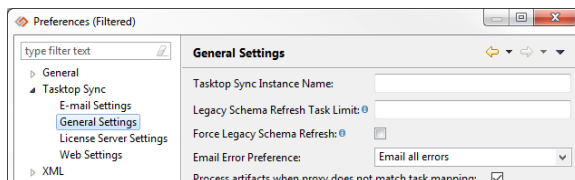
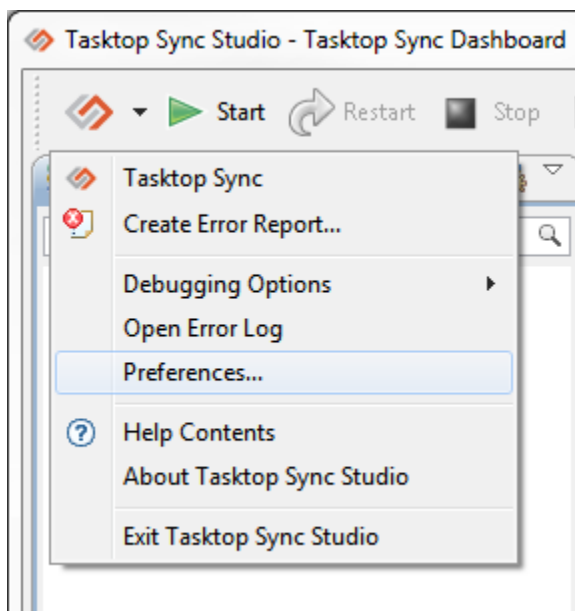
If an attachment was marked as invalid while attempting synchronization, it will not be synchronized in the future. To manually re-synchronize a task's attachments and clear their error state, open the Sync Tasks view, select one or more tasks, right-click, open the "Advanced" sub-menu, and select "Synchronize Attachments Again". This will cause all of the selected tasks' attachments to be synchronized as well as a [Marking for Synchronization](#) to be performed.

## Marking Comments as Synchronized

If there are new comments you do not wish to synchronize, then they can be marked as synchronized. To mark comments as synchronized, open the Sync Tasks view, select one or more tasks, right-click, open the "Advanced" sub-menu, and select "Mark Comments as Synchronized". This will cause all of the comments on the selected tasks to be marked as synchronized. The next time a task is processed all comments which were marked as synchronized will not be considered for synchronization.

## General Settings

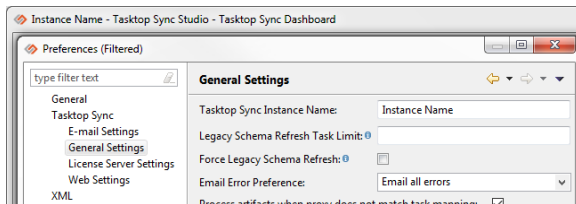
General settings for Tasktop Sync can be found under "General Settings" within the "Preferences" menu.



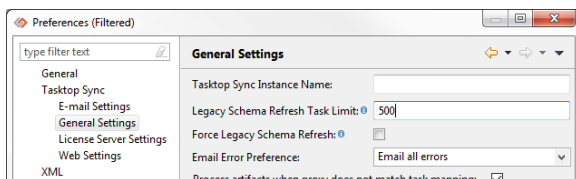
## Instance Name

Tasktop Sync can be assigned an instance name. This will appear in the application's title bar, in the web user interface, and in the subject of any emails Tasktop Sync may send. Information on emails can be found in the [Notifications](#) section. This setting may be useful when running multiple instances of Tasktop Sync.

The "Tasktop Sync Instance Name:" field sets the instance name, as shown below:

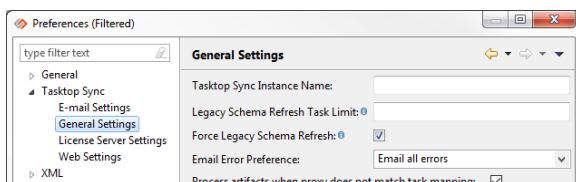


## Legacy Schema Refresh Task Limit



Some repositories require that all tasks be scanned when generating repository schema. This method is referred to as [Legacy Schema Refresh](#). The Legacy Schema Refresh Task Limit setting limits the number of tasks which are refreshed and scanned during Legacy Schema Refresh. This can reduce the time required for repository schema generation, but has the consequence of possibly failing to retrieve the latest configuration if the limit is set too low. If this not set or set to 0, then there will be no limit. For task repositories not using Legacy Schema Refresh, this setting has no effect. For more information about repository schema generation, see [Repository Configuration Changes](#).

## Force Legacy Schema Refresh

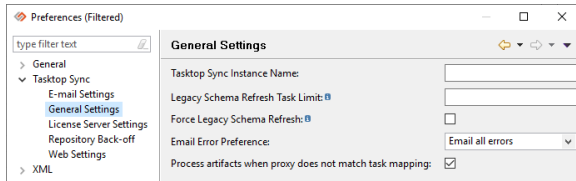


For task repositories that can provide repository schema details from a single task, you can still force Tasktop Sync use the slower [Legacy Schema Refresh](#) , forcing Tasktop Sync to review all tasks (or up to the limit defined in [Legacy Schema Refresh Task Limit](#) ) when refreshing the repository schema. To force Tasktop Sync to use Legacy Schema Refreshing on all repositories, click the checkbox beside "Force Legacy Schema Refresh". For more information about repository schema generation, see [Repository Configuration Changes](#).

## Email Error Preference

For details on this field, see [Notifications section](#) below.

## Process artifacts when proxy does not match task mapping



Tasktop Sync can be configured to process artifacts if the proxy task does not belong to the current task mapping. If this option is disabled, proxy tasks will be validated against the task mapping and will not be processed if they do not belong to it. The option is enabled by default.

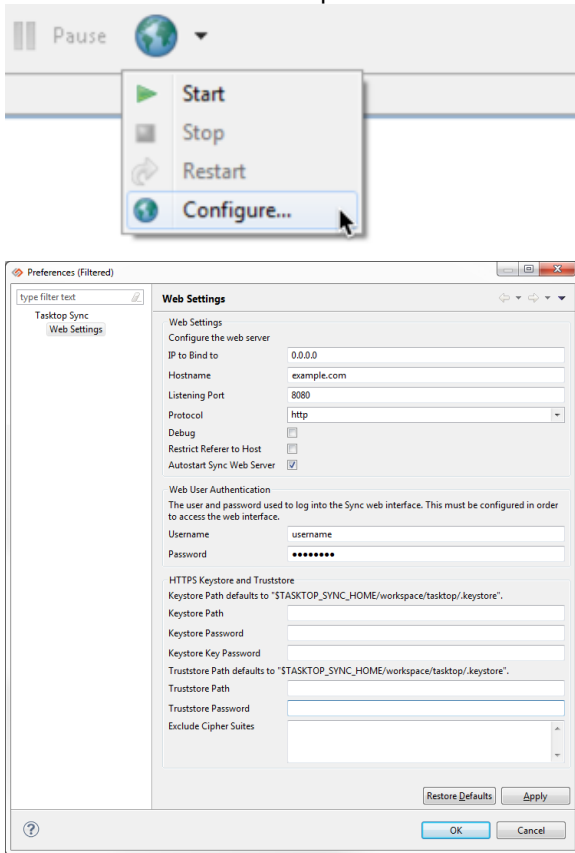
# The Tasktop Sync Web UI

## Web UI Configuration

The Tasktop Sync Web UI provides access to remote monitoring facilities for a Tasktop Sync installation. To enable the monitoring web UI you must configure the web username and password in the Tasktop Sync dashboard.

1. Start Tasktop Sync.
  2. Click on the pull-down menu to the right of the globe icon in the toolbar and select `Configure`.
- ...

3. Enter the username and password to the dialog that pops up.



For full details on configuring the Tasktop Sync Web UI see the section on [Web Container Configuration](#).

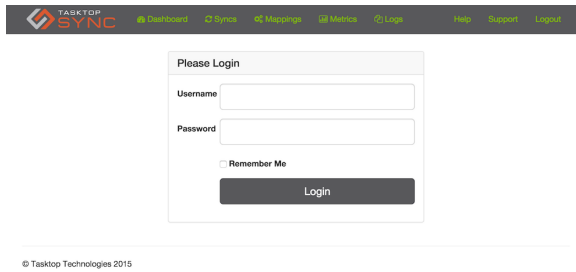
Once configured, see the next section for instructions on accessing the Web UI.

## Accessing the Web UI

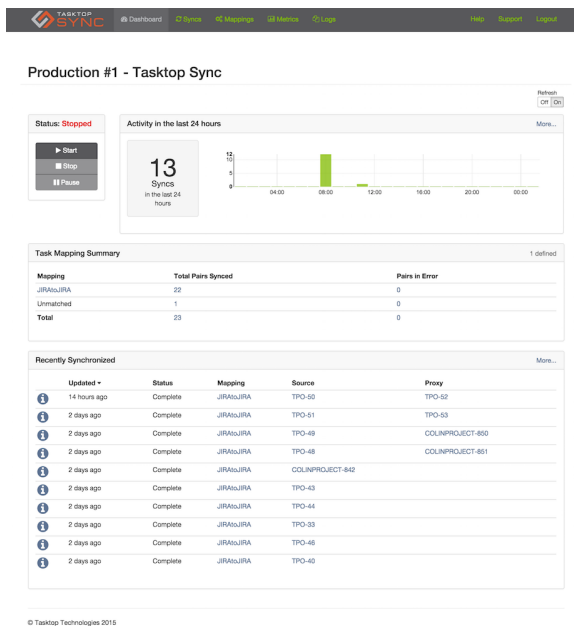
Tasktop Sync includes a remote monitoring interface that is accessible from any standard web browser. To access this UI you must be on a machine that can connect to the web services port and you must know the username and password. The [Web Container Configuration](#) section describes how to set the port and configure the username and password. The default port is 8080. You will also need to know the hostname or IP address of the machine running Sync. For the purposes of this example, we'll assume that the hostname of the machine running Sync is "sync.example.com" and the port is 8080.

To access the monitoring interface, enter the hostname, port and "/sync" into your web browser (i.e. <http://sync.example.com:8080/sync>). This will load the login page.





Enter the username and password and click `Login`. This will load the Sync Dashboard page.

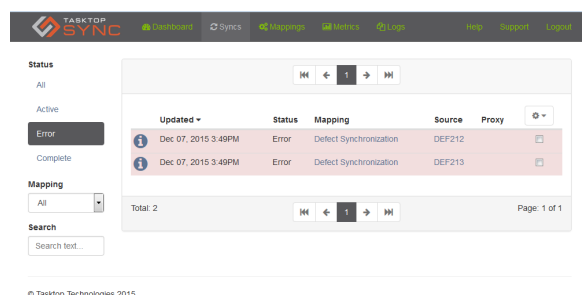


The header of each page in the web UI contains links to five sections of the website:

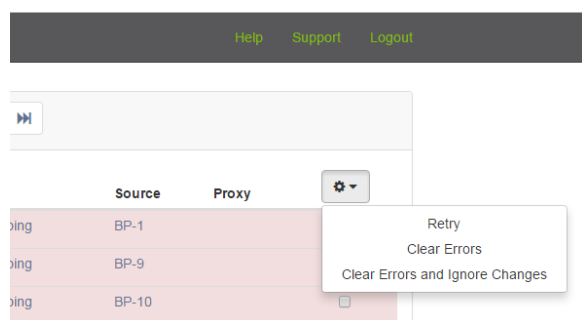
- **Dashboard:** This is the homepage of the web UI, which provides an overview of the daily synchronization activity and displays the most recent synchronizations. It also shows the current status of the synchronizer.
- **Syncs:** This tab contains a list of all the synchronizations along with each one's current status. Clicking on any of the items will display further details for that synchronization pair.
- **Mappings:** This tab contains a list of all the configured integrations. Clicking on the title of any of these will display more information about that mapping.
- **Metrics:** This tab provides an overview of the synchronization history over a user-specified period of time. See the [Usage Metrics](#) section for more information about this page.
- **Logs:** This tab provides access to all of the Sync log files. Each log file can be downloaded individually, or an archive of all the current log files can be downloaded by clicking the "All logs zipped" link.

## Reviewing Synchronizations in the Web UI

All synchronizations, including those in error, are visible and manageable from the Tasktop Sync web UI under the "Syncs" tab. The list of synchronizations can be filtered to those of a specific status by clicking on the desired status on the left hand side of the page. For example clicking "Error" will cause only synchronizations in error to be shown. Additional filtering by mapping, or text search, can be performed on the left hand side of the page.



When synchronizations are selected by clicking the checkboxes in the rightmost column of the list of synchronizations, a menu can be opened by clicking on the gear in the top right corner of the screen. This menu contains actions that can be taken with the selected synchronizations. The three possible actions are "Retry", "Clear Errors", and "Clear Errors and Ignore Changes".



Clicking "Retry" will queue the selected synchronizations to be processed by Tasktop Sync. Clicking "Clear Errors" will remove the selected synchronizations in error from the error queue and not mark them for processing. When an error is cleared the tasks will not be synchronized, and if the cause of the error is not addressed then it may occur again the next time one of the synchronized tasks is processed. For more details on Tasktop Sync errors see the documentation for [Error State](#). Clicking "Clear Errors and Ignore Changes" will perform the same action as well as [ignoring new changes](#) for the selected synchronizations.

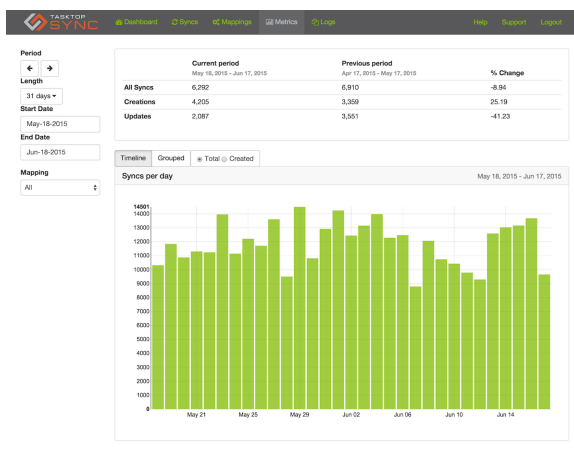
The "Clear Errors" and "Clear Errors and Ignore Changes" options are only available when synchronizations in error are selected. If the selected synchronizations include at least one synchronization not in error, only the "Retry" option will be available.

The details of a synchronization can be viewed by clicking on the information icon in the leftmost column of the list of synchronizations. This will open a page with details on the synchronization including the last time the synchronization was processed, what mapping was used in processing the synchronization, links to the tasks being synchronized, as well as all logs for the synchronization.

When viewing a specific synchronization it may be retried by clicking the gear icon in the top right corner of the screen, and clicking "Retry". Similarly if the synchronization being viewed is in error there will be the "Clear Error" and "Clear Error and Ignore Changes" options in the menu. Clicking "Clear Error" will remove the synchronization in error from Tasktop Sync's error queue. Clicking "Clear Error and Ignore Changes" will perform the same action as well as [ignoring new changes](#) for the selected synchronization.

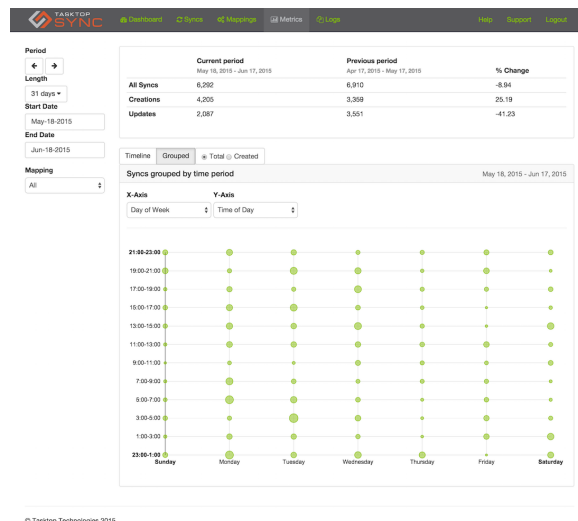
## Usage Metrics

Visibility into the synchronization usage occurring in your organization's integrations is key to being able to manage those integrations well. Understanding things like high-volume time periods, repositories that have the most artifacts being synced, and integrations that produce the highest load of synchronizations allows you to properly plan and manage your entire integrations landscape. To allow you to better understand this information, Tasktop Sync provides macro views and statistics that are dynamic and interactive through the web UI's Metrics tab. Here you will find an overview of the number of created and updated synchronizations for the current user-specified period, as well as a comparison of these statistics to those from the previous period. The length, start date, and end date of the period can all be modified in the panel on the left. For example, in the picture below we have chosen the period to start on October 27, 2013, and have a length of one month (31 days).



By default, the Metrics page will display the information as a bar chart describing the number of synchronizations (either total or created) that occurred per day for the selected period. By clicking on

the "Grouped" tab, you can display this information as a bubble chart of synchronizations grouped per time period. For example, the graph in the picture below plots the current period's synchronizations by time of day for each day of the week.



# Task Synchronization Configuration

## Creating a Task Mapping

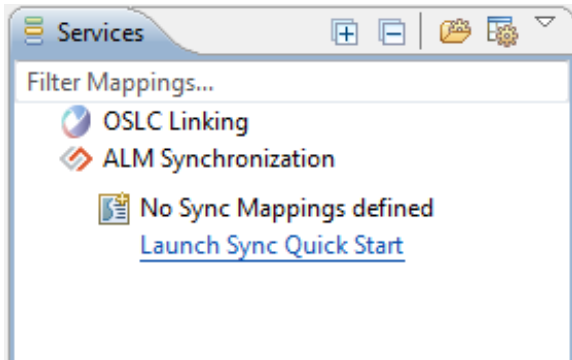
Creating a task mapping in Tasktop Sync requires that three primary items be configured: a connection to each participating repository, a set of queries defining the scope of the synchronization, and a mapping that defines how synchronization between two repositories should be handled.

Tasktop Sync provides a repository creation wizard as well as a thorough task mapping editor to assist you in creating all three of these requirements. However, some synchronization setups require complicated definitions. To support the full range of synchronization complexity, Tasktop Sync also allows manual configuration of the synchronization mapping through its configuration file. For details on manual editing of the configuration file see the [Manual Configuration](#) section.

However, even if you require the ability to later edit the configuration files by hand, it is recommended that you initially create your task mappings using the Quick Start Wizard and task editor.

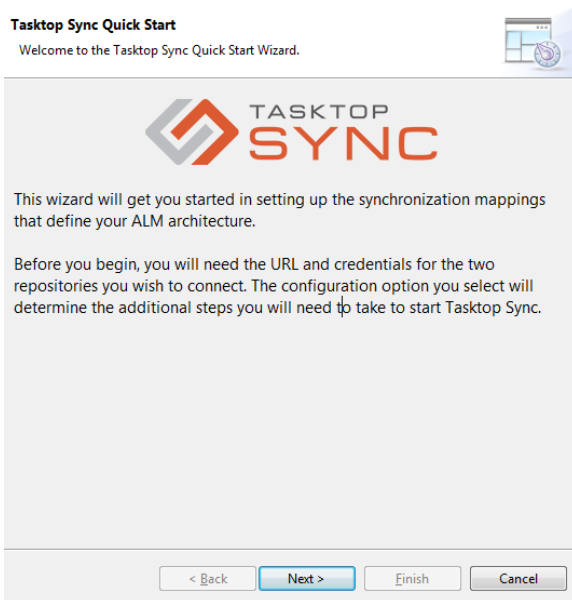
In order to use the Quick Start Wizard, you will need to create a test task in each of the repositories you want to sync (these tasks can be deleted after you are done with the Quick Start Wizard). You can use pre-existing tasks, but that is generally a bad idea if you are using a production system.

If you have not yet defined any mappings, you will see a "Launch Sync Quick Start" link in the upper left Services view:



If that link is available, click on it to start the Quick Start Wizard. Otherwise you can right-click the "ALM Synchronization" node in the Services view and select "New Task Mapping."

If you have not yet defined any mappings, you will be presented with a welcome page which reminds you that you need credentials for the two repositories to sync. This page will not appear if you already have mappings defined.

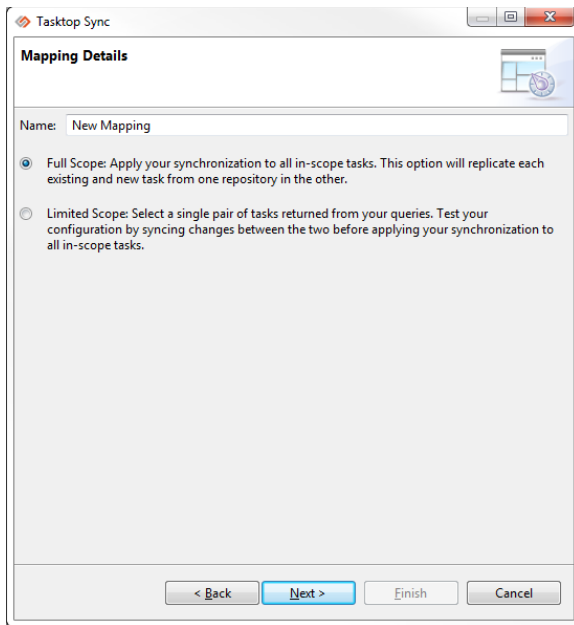


The next page will ask you for some details about the mapping to configure. In addition to the mapping name (used for display in the UI), you will be asked to choose your desired type of mapping.

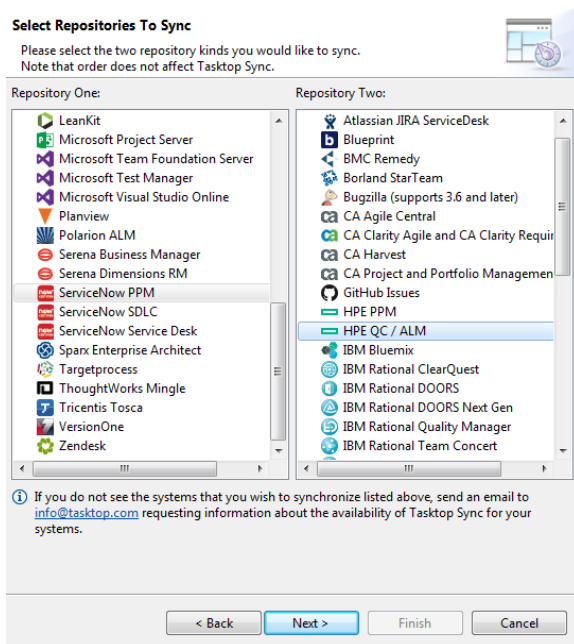
A "Full Scope" mapping will apply your synchronization to all existing tasks returned by your queries, replicating each existing task from one repository in the other and creating any new tasks added to one in the other. See [Create Queries](#) for more information.

A "Limited Scope" mapping asks you to select only a single pair of existing tasks returned from your queries to test your configuration between the two before applying your synchronization to all in-scope tasks.

Note that if you start with a scope limited to a single pair of tasks, you can convert it later to use queries to cover many tasks.



Next, you will see a page which allows you to specify which kind of repositories you would like to sync. The order in which you specify the two repositories doesn't affect the synchronization at all.



After you select which type of repository to use for Repository One and Repository Two, you will see two pages where you will enter your credentials for each repository. These will be different depending upon which repository type you select. Here is a screen shot of the credentials page for an HPE QC / ALM connector:

## Full Scope

If you chose "Full Scope", the next page will ask you some details about the source and scope of the first repository. This includes the two queries that define the mapping source, the scope of the tasks, and the proxy storage configuration.

For each repository, you need to configure queries. See [Create Queries](#) for more information about query scope. You may choose existing queries, or create new queries.

For each of the scope fields, select the value that matches your tasks within your queries.

You need to configure proxy storage. See [Proxy Association Storage](#) for more information.

This must be repeated on the next page for the second repository.

After you press **Finish**, Tasktop Sync will open the task mapping editor for your new mapping. The mapping will be configured to synchronize the summary attribute (and only the summary attribute) between the sets of tasks you defined.

## Limited Scope

If you chose "Limited Scope", the next two pages will ask you to choose queries from each repository. You only need to fill in the initialization query and the scope for each repository; the changes query and proxy storage configuration are optional.

**Configure Mapping Source and Scope for RTC 3.0.1**

RTC 3.0.1

Initialization Query 0 Tasks New...

Changes Query 0 Tasks New...

Project

Task Type:

Proxy Store Type Attribute

Proxy attribute: <Not Set>

Refresh

< Back Next > Finish Cancel

After completing the source and scope page for each repository, you must choose a task from each repository. The results of each initialization query are shown on the page; click on a single task in each list. To help you locate the desired task, you can enter some filter criteria to narrow down the list.

**Query Results**

Select an individual task from each side to only synchronize those tasks.

HPE QC / ALM RTC

All Defects All Features

Filter Tasks Define

DEF1: another issue!  
DEF3: Fix the problem with the matrix!  
DEF46: the matrix is breaking down!  
DEF5: Summary  
DEF6: New Defect

1: Define Phase Plan  
2: Define the initial Product Backlog  
3: Define a new build  
5: Define phases  
6: Define categories and releases for work items  
7: Define team members  
8: Define permissions

< Back Next > Finish Cancel

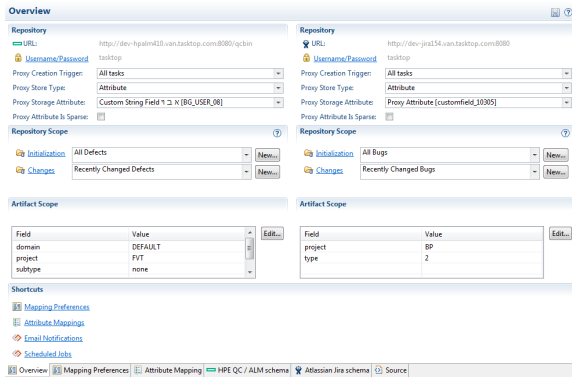
After you press **Finish**, Tasktop Sync will open the task mapping editor for your new mapping. The mapping will be configured to synchronize the summary attribute (and only the summary attribute) between the two test tasks that you created.

In order to start synchronizing, Tasktop Sync must be started. See [The Toolbar](#) for how to start Sync.

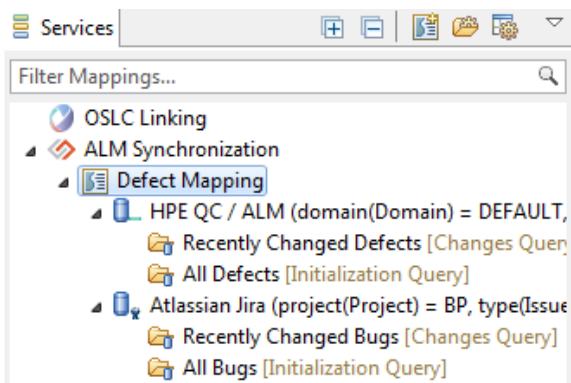


# Editing a Task Mapping: The Task Mapping Editor

The Task Mapping Editor displays the mapping configuration in a user-friendly format. It shows you information about the repositories, queries, and mappings. The editor also provides direct links for viewing repository schemas, editing repository scope, and editing query parameters.



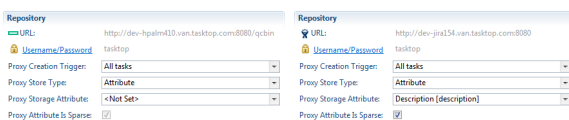
To access the Task Mapping Editor, double-click on the mapping title in the Services view or right-click the mapping title and select **Properties**.



## Task Mapping Editor Layout

### Repositories

General repository information is displayed in the upper sections of the Task Mapping editor. The repository information includes the repository URL, credentials, proxy creation trigger, and the proxy storage configuration. The name of the repository type will be in the header of this section, and the icon for the repository type will be next to the URL link.

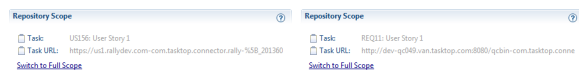


This section of the editor links to the repository "Properties" editor through the `URL` and `Username / Password` links. If a username or password has not been supplied, a yellow warning sign will be displayed next to the text box.

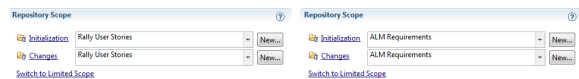
## Repository Scope

The `Repository Scope` section defines the set of tasks which are candidates for synchronization between two repositories. If you created your mapping with the Quick Start Wizard's [Limited Scope](#), the scope will initially be a single task for each repository. The URL of this task will be displayed in the Task ID field shown in this section.

To expand the scope of your synchronization you will have to configure Tasktop Sync to use a set of queries as its definition of the source. You can convert to using queries as the defining source by clicking on the "Switch to Full Scope" hyperlink shown underneath the Task ID field.



Once you have expanded the scope to use queries, this section will be replaced by two fields for selecting queries. If you chose your queries in Quick Start's [Limited Scope](#), then some or all of the queries will already be selected.



The "Initialization" query is used to initialize or reinitialize the synchronization between tasks. This query should capture all of the tasks that you wish to synchronize from each of the repositories.

The "Changes" query is used to capture the subset of the initialization query tasks which Tasktop Sync should check for changes that require synchronization. This query is run by Tasktop Sync frequently and so should ideally capture a smaller set of tasks than the initialization query. Typically this query captures tasks that have been modified in the last few days.

Note that Tasktop Sync will attempt to synchronize a task even if the proxy task doesn't fall into the queries on the other side of the mapping.

You can change these queries by clicking on the textbox and selecting a pre-existing query for that repository; clicking on the `Changes` or `Initialization` links will bring up the `Edit Query` dialog. Clicking on the `New...` button will allow the creation of a query that is immediately defined as your changes or initialization query.

## Artifact Scope

The `Artifact Scope` section displays the fields used to match your tasks within the repository. When tasks are received from your source queries, their attributes must match the scope field values

defined here; otherwise, they will be ignored. Each repository will have its defined list of scope fields, and their values. These can be defined by using [Quick Start](#), or they can be edited by clicking the `Edit` button. Consult the Tasktop Sync Connectors guide for more information.

It is possible that the specification of a system's scope may change when Tasktop Sync is updated. Tasktop Sync's configuration must be updated to align with the new scope definition. Until the configuration is updated Tasktop Sync will not run. Details on how to handle this scenario are found on the Tasktop Sync [FAQ](#).

Artifact Scope		
Field	Value	<input type="button" value="Edit"/>
Item Type	BUG	

Artifact Scope		
Field	Value	<input type="button" value="Edit"/>
Item Type	Bug	
project	DEMO	

## Mapping Preferences

The Mapping Preferences editor defines further mapping configuration options, including attachment, comment, and time worked synchronization, as well as person mappings.

The screenshot shows the 'Mapping Preferences' dialog with two repository configurations. The left repository is 'http://dev-jira54.xen.tasktop.com:8080' and the right is 'https://devon058.zendesk.com'. Both have 'tasktop' as the 'Auto-Comment User'. The left repository has 'Do not validate' for 'Author Validation Policy', while the right has 'Use auto-comment user on validation failure'. Both have 'Synchronize Incoming Time Worked' checked. Under 'Comment Synchronization', both have 'Synchronize Incoming Comments' checked, 'Add Author Header To Comments' checked, 'Comment Content Type' set to 'JIRA Confluence', 'Comment Visibility To Synchronize' set to 'All Comments', and 'New Comment Visibility' set to 'Submit Comments Privately'. Under 'Attachment Synchronization', both have 'Synchronize Incoming Attachments' checked. The left has 'Large Attachment Handling' set to 'Allow all attachments' and 'Maximum Attachment Size (Bytes)' set to '4096'. The right has 'Large Attachment Handling' set to 'Ignore oversized attachments' and 'Maximum Attachment Size (Bytes)' set to '4096'. Both have 'Invalid Attachment Handling' set to 'Fail on attachment error'. At the bottom, there is a 'Person Mapping' section with a 'Browse...' button, and 'Conflict Notification Policy' set to 'Log' and 'Conflict Resolution Policy' set to 'Do not synchronize'.

## Repository Preferences

The screenshot shows the 'Repository Preferences' dialog with two repository configurations. The left repository is 'tasktop' and the right is 'admin'. Both have 'tasktop' as the 'Auto-Comment User'. Both have 'Do not validate' for 'Author Validation Policy'. Both have 'Synchronize Incoming Time Worked' checked.

The "auto-comment user" is the user who will appear as the author of Tasktop Sync's automatically generated comments.

The "author validation policy" is employed when synchronizing both comments and attachments. It is used to decide whether to validate the authors of comments and attachments, and what to do when author validation fails. When "Do not validate" is selected the mapped authors of comments and attachments will not be validated. When "Use auto-comment user on validation failure" is selected the mapped authors of comments will be validated to determine if they exist in the target repository, if they do not exist the auto-comment user will be used as the author on submission.

To synchronize worklogs, enable "Synchronize Incoming Time Worked". If enabled, a user's time worked entries will be synchronized to the associated repository.

## Comment Synchronization

To synchronize comments between tasks, click one or both checkboxes labeled "Synchronize Incoming Comments". Each checkbox represents the target repository to copy comments into.

Aside from enabling or disabling comment synchronization, the "comment content-type" can also be defined for this repository. The "comment content-type" will be used during comment synchronization to convert rich text comments from the format used in the source repository to the format used in the target repository.

The "Add Author Header to Comments" can also be enabled or disabled. The "author header" is a short line of text appended to the beginning of a comment which states the author of the comment from the source repository. For example, if this setting is enabled and the user `John Smith` in the source repository created the comment `This is a comment`, then the comment will be appear as the following in the target repository:

(Comment by John Smith)

This is a comment

Enabling this setting will cause the author header to be added to every new comment synchronized into this repository. Previously synchronized comments will not be modified.

The "Comment Visibility To Synchronize" setting allows the synchronization of specific comment visibilities (e.g. only synchronizing private comments). This setting can only be used by repositories which support the notion of public and private comments. When this setting is set to "All Comments", no filtering is done and both public and private comments will be synchronized out of the repository. When this setting is set to "Only Private Comments", only private comments will be synchronized out of this repository. When this setting is set to "Only Public Comments", only public comments will be synchronized out of this repository.

The "New Comment Visibility" settings specifies the visibility of the comments that Sync will create in that system. This setting can only be used by repositories which support the notion of public and private comments. When this setting is set to "Retain Comment Visibility" newly created comments will have the same visibility as in the comment's originating system. When this settings is set to "Submit Comments Privately" newly created comments will all have a visibility of private regardless of their visibility in the originating system. Sync can similarly assign public visibility to all new comments by selecting "Submit Comments Publicly". It is recommended that a setting other than "Retain Comment Visibility" be selected when synchronizing comments from a system which does not support public and private comments to one that does.

## Attachment Synchronization

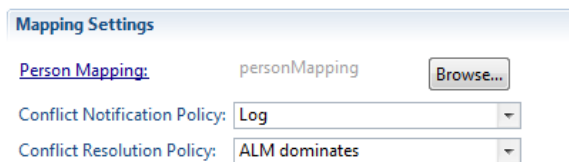
To synchronize attachments between tasks, click one or both checkboxes labeled "Synchronize Incoming Attachments". Each checkbox represents the target repository to copy attachments into.

Beyond enabling or disabling attachment synchronization, a "Large Attachment Handling" filter may be defined for this repository. Defining a "Large Attachment Handling" filter will place an upper limit on the size of attachments which will be synchronized into the repository. The attachment filter is configurable with a custom maximum file size and a handling strategy, either to fail a synchronization when an attachment is oversized or to skip the synchronization of the oversized attachments allowing the rest of the synchronization to proceed. The criteria for an attachment being oversized can be set by changing the "Maximum Attachment Size (Bytes)" value.

Additionally, you can configure an "Invalid Attachment Handling" filter to handle invalid attachments. The attachment filter is configurable to either fail on attachment error or to ignore attachment errors during synchronization. After problems are resolved, skipped attachments can be re-synchronized by [synchronizing attachments again](#).

## General Preferences

The `General Preferences` defines configuration options that apply to the entire task mapping. These options include an optional person mapping file (see [person mappings](#)) and conflict resolution and notification policies.



The screenshot shows a dialog box titled "Mapping Settings". It contains three configuration options: "Person Mapping:" with a text field containing "personMapping" and a "Browse..." button; "Conflict Notification Policy:" with a dropdown menu set to "Log"; and "Conflict Resolution Policy:" with a dropdown menu set to "ALM dominates".

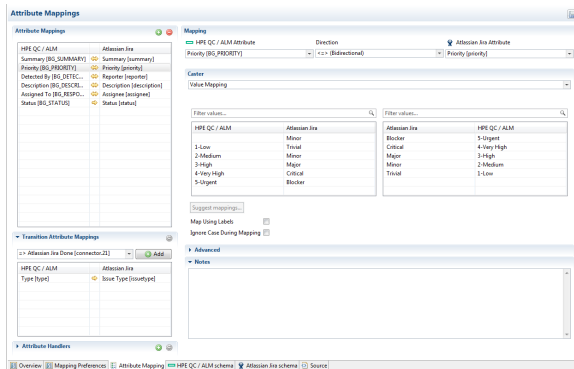
The filename found in the Person Mapping box will also display at the top of the Services view as a link to the person mapping file.

Conflict notification can either be set as `Log` or `Comment`. If `Comment` is selected, when Tasktop Sync detects a conflict it will be reported as a comment in the target task where the conflict was detected. If `Log` is selected, detected conflicts will be reported in the Tasktop Sync console and on-disk logs. It is recommended that you use `Log` for conflict notification.

The `Conflict Resolution Policy` box will display the default conflict resolution policy of this task mapping. When a conflict is detected, Tasktop Sync uses this policy to determine what value should be recorded for the attribute in question. If `Do not synchronize` is specified then no action is taken to resolve the conflict, and each repository will keep its own value. Otherwise, one of the repositories can be specified as the dominant repository. In this case, the value from the dominant repository will take precedence in resolving the conflict. The conflict resolution policy can be overridden on an attribute-by-attribute basis in the [attribute mapping editor](#).

## Attribute Mappings Editor

The attribute mappings editor is used to configure the specific attribute-level mappings that are used when synchronizing tasks between repositories. Before you can configure attribute mappings, you must have a refreshed schema for each repository (see [the schema view](#) ). Each part of the attribute mapping editor is described below.

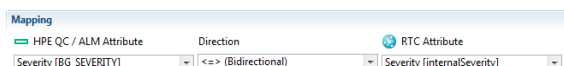


## Mappings Table

Attribute Mappings	
HPE QC / ALM	RTC
Summary [BG_S...	Summary [sum...
Priority [BG_PRI...	Priority [internal...
Severity [BG_SE...	Severity [interna...
Modified [BG_V...	Modified By [m...
Detected on Dat...	Creation Date [c...
Description [BG...	Description [des...
Assigned To [B...	Owned By [own...

The Mappings table displays all the attributes currently being synchronized. The arrows between the attributes displayed visualize the direction in which the attributes are synced; synchronization can be unidirectional from one repository to another, bidirectional, or non-existent if the mapping should not be applied. The plus and minus buttons on the top right allow the addition and removal of attribute mappings.

## Mapping



This area allows you to define the attributes being mapped. The direction of synchronization can be configured in the Direction box.

## Caster

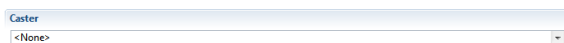
Similar attributes on different repositories often come in different formats, resulting in the need for values to be transformed to the proper format for a given repository. The caster section allows you to configure casters which can perform these transformations.

Casters can also be used for maintaining task relationships. When a task has a relationship to another task in the repository, such as a related defect or a parent requirement, this relationship is exposed as an attribute. For example, an HPE ALM defect has an attribute named "Linked Entities", which contains a list of related defects, requirements, and test cases. When synchronizing this defect, you may want to synchronize this relationship as well. Tasktop Sync can mirror this relationship to a target repository, if all tasks involved are being synchronized, or it can synchronize the relationship as a web URL, linking back to the source repository.

Once you have selected the attributes being mapped, and the desired direction of the synchronization, the list of casters will be presented. This list is dependent on the mapping details, and will vary depending on the attribute types. (To determine the attribute type, click on the repository [schema](#) page, and click on the desired attribute.)

Tasktop Sync provides several different casters by default. This section will explain how to configure and use these casters.

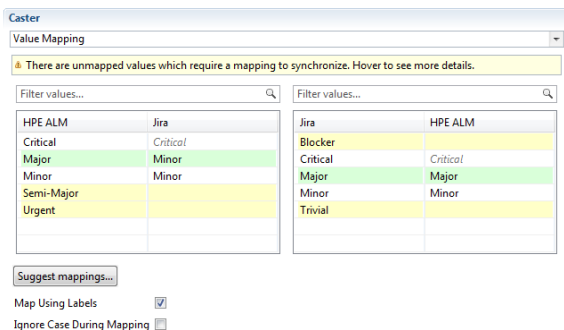
## No Caster



Caster  
<None>

The simplest cases, such as syncing `Summary`, require no casters at all, and `<None>` is the default value.

## Value Mapping



Caster  
Value Mapping

⚠ There are unmapped values which require a mapping to synchronize. Hover to see more details.

Filter values... Filter values...

HPE ALM	Jira
Critical	Critical
Major	Minor
Minor	Minor
Semi-Major	
Urgent	

Jira	HPE ALM
Blocker	
Critical	Critical
Major	Major
Minor	Minor
Trivial	

Suggest mappings...

Map Using Labels ☒

Ignore Case During Mapping ☐

Some enumeration attributes, such as "Severity" and "Priority" attributes, often have different values between repositories. To map a finite set of values to another finite set of values, such as with enumeration attributes, use the value mapping caster.

Because each mapped attribute may have a different number of values, the value mapping caster allows you to provide a mapping for each direction of the value. Use the tables provided to configure the mappings between the repositories. Note that you should take care to not confuse which direction applies to which table. In the example above, the table on the left shows the mapping of HPE ALM values to Jira values, and the table on the right shows the mapping of Jira values to HPE ALM values.

So, for example. "Major" in HPE ALM becomes "Minor" in Jira. But "Major" in Jira becomes "Major" in HPE ALM.

Tasktop Sync will automatically map two options if they share the same id, when synchronizing by id, or the same label, when synchronizing by label. These mappings do not require manual configuration, and are displayed in the tables as italicized, grey text. The "Critical" value in the example above is an example of such a mapping. Note that these mappings can be overridden by manually configuring the value mapping for the automatically mapped value.

The tables will also display values which are unmapped and cannot be automatically synchronized. These mappings are highlighted in yellow. The values "Semi-Major" and "Urgent" in the above example are such mappings. Additionally, mappings which have been configured but have not been saved are highlighted in green. The "Major" value in the above example is such a mapping.

If there are similar values on both repositories, you can click the "Suggest mappings..." button to automatically fill in mappings. This will generate mappings between values which have similar labels, but will not overwrite any existing mappings.

The value mapping caster can also be configured to map using labels. This setting can be enabled by checking the "Map Using Labels" checkbox. When this setting is enabled the caster will consider only the user facing labels on the attribute and not the internal IDs. This can be useful for sharing value mappings between configuration templates for systems whose IDs are project or type specific.

For example. If a field in Jira has an option "Normal" with ID "1" and IBM RTC has an option "Medium" with ID "\_FggUIOfEEeG6psipjwbkrQ" the value mapping caster can be configured to map "Normal" to "Medium" instead of mapping "1" to "\_FggUIOfEEeG6psipjwbkrQ" when the map by labels attribute is enabled. The value mapping caster may then be stored in a configuration template which can be re-used to create multiple task mappings involving multiple Jira and RTC projects even if the option IDs change, as long as the labels are consistent across all projects.

This mode has the additional functionality of automatically mapping values which have the same label.

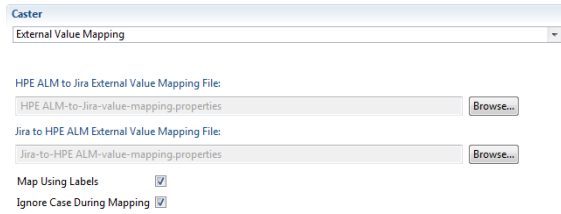
For example. If a field in Jira has an option "Normal" with ID "1" and IBM RTC has an option "Normal" with ID "\_FggUIOfEEeG6psipjwbkrQ" the value mapping caster will map these automatically without an explicit mapping when set to map using labels. However if an explicit mapping exists for this option it will be applied instead of the implicit label mapping.

The value mapping caster can also be configured to ignore case when mapping. This setting is enabled by checking the "Ignore Case During Mapping" checkbox. Enabling this setting will cause the case of the incoming value to be ignored when mapping.

For example, consider the a mapping where the value with id "1" and label "Normal" is mapped to the value with id "a" and label "Medium". If "Ignore Case When Mapping" is disabled, and "Map Using Label" is enabled, then an incoming value with label "Normal" will be mapped to "Medium", whereas an incoming value with label "normal" will not. If "Ignore Case When Mapping" and "Map Using Label" are both enabled, then an incoming value with label "Normal", an incoming value with label "normal", and an incoming value with label "nORmaL" will all be mapped to "Medium".



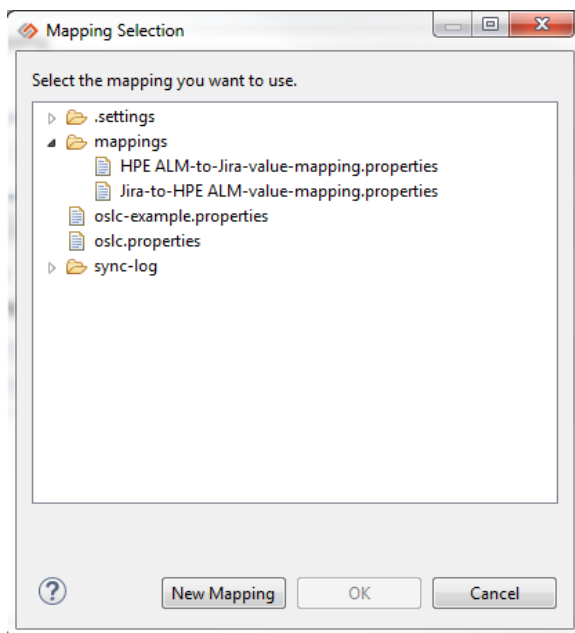
## External Value Mapping



The screenshot shows the 'Caster' configuration window for 'External Value Mapping'. It features two file selection fields: 'HPE ALM to Jira External Value Mapping File' and 'Jira to HPE ALM External Value Mapping File', each with a 'Browse...' button. Below these are two checked checkboxes: 'Map Using Labels' and 'Ignore Case During Mapping'.

A value mapping can also be defined using a properties file. The properties file will contain the desired mappings, and can be loaded into Tasktop Sync through the user interface. The "Map Using Labels" checkbox indicates whether the mappings within the properties files are specified using the option label or the option id. The "Ignore Case During Mapping" checkbox indicates whether the caster should ignore case when mapping. This will function in the exact same way as the Value Mapping caster. For more information, see the [Value Mapping](#) section.

Note: In order for the properties files to be used, they must reside within the "tasktop" directory within the workspace.



A new mapping file can be generated by clicking the "New Mapping" button. This will generate a file which contains mappings for values which have the same label or id, depending on whether "Map Using Labels" is checked.

The properties files for the mappings specified in the [Value Mapping](#) section would be defined as follows:

The "HPE ALM-Jira-value-mapping.properties" file:

```
1-Low=Trivial
2-Medium=Minor
3-High=Major
4-Very\ High=Critical
5-Urgent=Blocker
```

The "Jira-HPE ALM-value-mapping.properties" file:

```
Blocker=5-Urgent
Critical=3-High
Major=2-Medium
Minor=1-Low
Trivial=1-Low
```

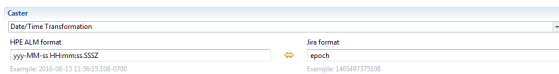
Note that two different mapping files are required (one for each direction) because the mappings are not one-to-one.

The contents of the properties files follow the Java properties file format. Some details of the format are follows:

- Whitespace in property names must be escaped with a '\'. e.g. 4-Very High must be written as 4-Very\ High when it is a property name.
- Whitespace in property values need *not* escaped with a '\'. e.g. 4-Very High may be written as is when it is a property value.
- The '\' character must also be escaped with a '\'. e.g. Blocker\Showstopper must be written as Blocker\\Showstopper in the properties file.

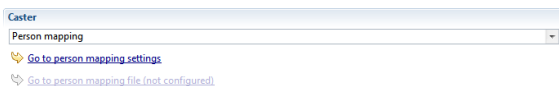
Further details on the Java properties file format may be found in the [Oracle Java documentation](#)

## Date/Time Transformation



For repositories with different date/time formats with attributes such as Date Modified or Date Created, a Date/Time Transformation may need to be used. Pressing **Ctrl-space** will enable content-assist when filling out the formats. See the [Date-Time Caster](#) section for more details.

## Person Mapping



For fields such as Assignee, Reporter or any other field involving the user identity of a person, the Person Mapping caster can be used if user IDs are different in each repository. This caster must be used in conjunction with the person mapping file defined in the [General Preferences](#).

Clicking on **Go to person mapping settings** goes to the [General Preferences](#) section of the task mapping editor, to define which person mapping file is being used.

Clicking on **Go to person mapping file** will go to the Person mapping file currently being used. The person mapping caster uses mapping specified in this file. Read about [person mapping files](#) to learn how they are configured.

## Text Markup Transformation

The screenshot shows the 'Caster' configuration window. The main dropdown is set to 'Text Markup Transformation'. Below it, there are two sections: 'HPE ALM format' with a dropdown set to 'HTML', and 'Jira format' with a dropdown set to 'JIRAConfluence'. A small icon is visible between the two format sections.

For fields such as `Description`, different repositories may use different markup to store formatting. When this is the case, use the Text Markup Transformation caster. Select from the drop-down menu to specify the text content-type of each repository.

In the case that a repository does not support rich text, we suggest setting the format to `Textile`, as it is a rich text format which is easily readable as plain text.

If both repositories use HTML markup, we suggest using the Text Markup Transformation with `HTML` selected for the text content type for both repositories. This will enable a transformation that handles differences in HTML generated by repositories from different vendors.

## Literal Value

The screenshot shows the 'Caster' configuration window. The main dropdown is set to 'Literal Value'. Below it, there are two input fields: 'Fixed value for HPE ALM' and 'Fixed value for Jira'. Both fields have a small 'Type: Short Rich Text' label below them.

If you want attributes to be set to a specific fixed value, use the Literal Value caster. Read [here](#) for more information about the usage of this caster. If the attribute being written to with a literal caster comes from an enumeration, then contest-assist, available via `Ctrl-space`, is available to get a list of options to select from. This can be useful for setting defaults on enumerated types when syncing repositories with required fields which do not conveniently map to a field in the other repository.

## Multi-value to CSV String

The screenshot shows the 'Caster' configuration window. The main dropdown is set to 'Multi-Value to CSV mapping'.

If you have a multi-select attribute in one repository and want to synchronize the labels of the selected options to a string field on the other, you can use the Multi-value to CSV String caster. This caster supports bidirectional synchronization between a multi-select field and plain-text string field. When synchronizing to the string field, the value will be a comma delimited list of the labels from the multi-select field (e.g. `one,two`) and when synchronized to the multi-select, the correct options will be selected based on the labels provided in the string field. When using this caster, the none of the values of the multi-select field may contain a comma.

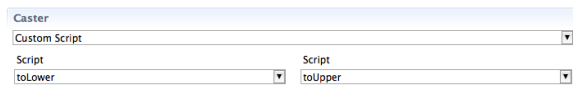
## Value to String

The screenshot shows the 'Caster' configuration window. The main dropdown is set to 'Value to String mapping'.

If you have a single-select attribute in one repository and want to synchronize the label of the selected option to a string field on the other, you can use the Value to String caster. This is also useful when you just want a 1-way mapping of the label from a single-select field to a string field for reference in the other system. This caster supports bidirectional synchronization between a single-select field and plain-text string field. When synchronizing to the string field, the label of the value in the single-select will be

written. When synchronizing to the single-select field, the correct value will be selected that matches the label specified in the string field.

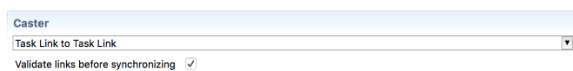
## Custom Script



The screenshot shows a configuration window for a 'Caster'. It has a dropdown menu set to 'Custom Script'. Below this, there are two 'Script' labels, each followed by a dropdown menu. The first dropdown is set to 'toLower' and the second is set to 'toUpperCase'.

If you have defined custom casters, then you can use these as casters for your attributes. Read [here](#) for more information about how to define custom casters. To assign your custom caster, first select the caster "Custom Script". You can then assign a custom caster to each attribute. These casters will be used when writing to the associated attribute.

## Task Link to Task Link



The screenshot shows a configuration window for a 'Caster'. It has a dropdown menu set to 'Task Link to Task Link'. Below this dropdown, there is a checkbox labeled 'Validate links before synchronizing' which is checked.

When synchronizing task relationships, use the Task Link to Task Link caster. This caster will copy the relationship from one repository to the other, mirroring the task hierarchy in each. For example, if you are synchronizing both test cases and defects between two repositories, you can maintain the link between them in both repositories.

This caster can only be selected for a mapping that meets the following condition:

1. Both attributes in the mapping must be of the type "taskDependency". (To determine the attribute type, click on the repository [schema](#) page, and click on the desired attribute.)

If this condition is not met, the caster will not appear in the Caster selection list.

### Validate links before synchronizing

There are conditions where relationships in the source task should not all be synchronized to the target task, and you only want to copy over a subset of relationships. There are two primary examples of these conditions: when the referenced task in the source will not synchronize to the target repository, or when the target task's link field has restrictions on the types of relationships it accepts.

Another example of this condition: when an artifact has a relationship (link) to another artifact in a project that is inaccessible, this blocks the entire artifact from syncing to the target artifact. The fix in version 4.19 skips the inaccessible artifact and continues to sync the rest of the artifacts when "validate links before synchronizing" is checked.

For example, if your relationship references both test cases and defects, but Tasktop Sync only synchronizes test cases, then any tasks that reference defects will result in an error (since the defects do not have proxy associations in the target system). To further the example, even if both test cases and defects are synchronized, but the target repository only allows links to defects, then any tasks that reference test cases will result in an error.

To stop this from happening, check the box beside `validate links before synchronizing`. When this option is checked, Tasktop Sync will only synchronize relationships that meet the following conditions:

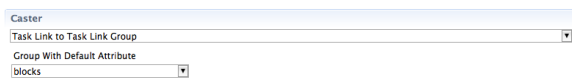
1. The referenced task matches a task mapping synchronizing to the target repository (see "Task Matching Criteria" below).
2. The referenced task does not match a task mapping synchronizing to the target repository, but is in an **accessible** project.
3. If there is a restriction on the target attribute, the referenced task, when synchronized, meets these restrictions.

All references to tasks that do not match the conditions will be ignored.

Task Matching Criteria: A task "matches" a task mapping if the following criteria are met:

1. The repository URL of the task and the task mapping are the same.
2. The task attributes match the values defined in the task mapping [Artifact Scope](#).
3. If specified, the task meets the task mapping [conditions](#).
4. The task is returned by the task mapping's changes query

## Task Link to Task Link Group



To synchronize multiple task relation attributes from one repository to a single task relation attribute in another repository, use the Task Link to Task Link Group caster. For example, if one repository defines many different forms of relationships in different attributes, such as "blocked by" and "duplicates", and your target repository has only a single attribute for these relationships, "links", you can use the Task Link to Task Link Group caster to combine all relationships in both "blocked by" and "duplicates" into the target of "links".

This caster works as a compliment to the Task Link to Task Link caster, allowing you to combine two or more attribute into a single mapping to a target attribute. This caster can only be used to combine the mapping with an existing mapping with the Task Link to Task Link caster, so you must define this mapping first. To continue the previous example, you would first define a mapping between "blocked by" and "links" with a Task Link to Task Link caster. You then create a second mapping between "duplicates" and "links" using the Task Link to Task Link Group caster. In a final step, in the configuration of the Task Link to Task Link Group caster, you combine the two mappings by assigning "blocked by" as the default attribute.

An important feature of the Task Link to Task Link Group caster is that it only supports unidirectional mappings. The associated Task Link to Task Link caster can be bidirectional. This directs Sync on which attribute to add new task relations found in the target repository attribute. To once again continue on the example, if a new task relation was added to "links", this task relation would be synced to "blocked by", since it was mapped with the bidirectional Task Link to Task Link caster.

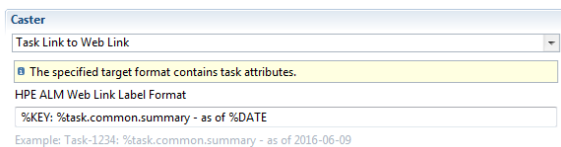
This caster can only be selected for a mapping that meets the following conditions:

1. Both attributes in the mapping must be of the type "taskDependency". (To determine the attribute type, click on the repository [schema](#) page, and click on the desired attribute.)
2. The mapping must be unidirectional from the repository with multiple attributes to the repository with the single attribute.

If these conditions are not met, the caster will not appear in the Caster selection list.

Note: The relationship can only be synchronized if the tasks on both sides of the relationship are being synchronized between the two repositories. If only one of the tasks is being synchronized, the synchronization will result in an error.

## Task Link to Web Link



The screenshot shows a configuration window for a 'Caster'. The title bar says 'Caster'. Below it is a dropdown menu with 'Task Link to Web Link' selected. A yellow warning box contains the text 'The specified target format contains task attributes.' Below the warning is a text field with the label 'HPE ALM Web Link Label Format' and the value '%KEY: %task.common.summary - as of %DATE'. At the bottom, there is an example text: 'Example: Task-1234: %task.common.summary - as of 2016-06-09'.

To synchronize an internally referenced task as a web URL, use the Task Link to Web Link caster. With this caster, you can synchronize the association to referenced tasks, without synchronizing the referenced tasks themselves. What will be synchronized to the target task will be a web URL, which can be opened in a browser to view the referenced task.

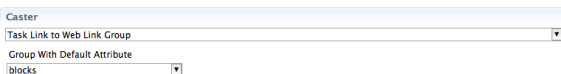
This caster also supports formatting of the target link's label. Pressing `Ctrl-space` will enable content-assist when filling out the format. See [Task Link To Web Link Caster](#) in the advanced [Transforming Task Attribute Values with Casters](#) section for more details.

This caster can only be selected for a mapping that meets the following conditions:

1. One attribute in the mapping must be of the type "taskDependency". (To determine the attribute type, click on the repository [schema](#) page, and click on the desired attribute.)
2. One attribute in the mapping must be of the type "externalLink".
3. The mapping must be unidirectional from the "taskDependency" attribute to the "externalLink" attribute.

If these conditions are not met, the caster will not appear in the Caster selection list.

## Task Link to Web Link Group



The screenshot shows a configuration window for a 'Caster'. The title bar says 'Caster'. Below it is a dropdown menu with 'Task Link to Web Link Group' selected. Below that is another dropdown menu with 'Group With Default Attribute' selected. At the bottom, there is a text field with the label 'blocks'.

To synchronize multiple internal task relation attributes from one repository to a single web URL attribute in another repository, use the Task Link to Web Link Group caster. For example, if one repository defines many different forms of relationships in different attributes, such as "blocked by" and "duplicates", and your target repository has only a single attribute for storing web URL link, "web

links", you can use the Task Link to Web Link Group caster to combine all internal relationships in both "blocked by" and "duplicates" as URLs into the target of "web links". What will be synchronized to the target task will be web URLs, which can be opened in a browser to view the referenced task.

This caster works as a compliment to the Task Link to Web Link caster, allowing you to combine two or more attribute into a single mapping to a target attribute. This caster can only be used to combine the mapping with an existing mapping with the Task Link to Web Link caster, so you must define this mapping first. To continue the previous example, you would first define a mapping between "blocked by" and "web links" with a Task Link to Web Link caster. You then create a second mapping between "duplicates" and "web links" using the Task Link to Web Link Group caster. In a final step, in the configuration of the Task Link to Web Link Group caster, you combine the two mappings by assigning "blocked by" as the default attribute.

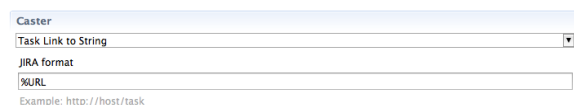
This caster also supports formatting of the target link's label. The configuration for this caster is shared with the configuration for the referenced default attribute mapping.

This caster can only be selected for a mapping that meets the following conditions:

1. The source attribute in the mapping must be of the type "taskDependency". (To determine the attribute type, click on the repository [schema](#) page, and click on the desired attribute.)
2. The target attribute in the mapping must be of the type "externalLink".
3. The mapping must be unidirectional from the "taskDependency" attribute to the "externalLink" attribute.

If these conditions are not met, the caster will not appear in the Caster selection list.

## Task Link to String



The screenshot shows a configuration window for the 'Task Link to String' caster. It includes a 'Caster' dropdown menu with 'Task Link to String' selected, a 'JIRA format' dropdown menu with '%URL' selected, and a text input field containing the example 'http://host/task'.

If you need to synchronize an internally referenced task as a web URL, but your target repository does not support attached web URLs, use the Task Link to String caster. With this caster, you can synchronize one or more associations to referenced tasks as web URLs, formatted into text attributes, in your target repository.

This caster also supports formatting of the target text attribute. Pressing `Ctrl-space` will enable content-assist when filling out the format. See [Task Link To String Caster](#) in the advanced [Transforming Task Attribute Values with Casters](#) section for more details.

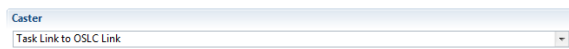
This caster can only be selected for a mapping that meets the following conditions:

1. One attribute in the mapping must be of the type "taskDependency". (To determine the attribute type, click on the repository [schema](#) page, and click on the desired attribute.)
2. One attribute in the mapping must support text entry.
3. The mapping must be unidirectional from the "taskDependency" attribute to the text attribute.

If these conditions are not met, the caster will not appear in the Caster selection list.

Note: Since the resulting text will concatenate multiple web URLs, it can be quite large. It is recommended that the target text attribute support strings of at least 1024 characters.

## Task Link to OSLC Link



When synchronizing task links to a system which uses OSLC linking, such as IBM Rational Team Concert, Tasktop Sync can be configured to synchronize Task Links to OSLC Links. This caster will convert the internal Task Link on the source task into a link referencing the linked task via Tasktop Sync's built-in OSLC Adapter.

To use this caster, Tasktop Sync's web server and OSLC adapter must be fully configured. For details on configuring Tasktop Sync's OSLC Adapter, see the sections on [Task Linking Configuration](#) and [Web Container Configuration](#). For Tasktop Sync to provide the linked task via OSLC, the linked task must appear in a query which is configured to be exposed by the Tasktop OSLC Adapter.

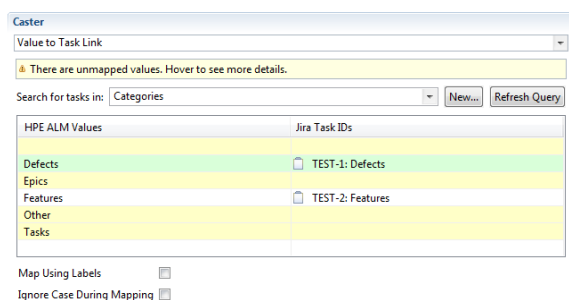
If any of the above requirements are not met the Caster may fail to cast the Task Link, or the OSLC Link may not be able to be resolved by Tasktop Sync's OSLC Adapter.

This caster can only be selected for a mapping which meets the following conditions:

1. One attribute in the mapping must be of the type "taskDependency". (To determine the attribute type, click on the repository [schema](#) page, and click on the desired attribute.)
2. One attribute in the mapping must be of the type "externalLink".
3. The mapping must be unidirectional from the "taskDependency" attribute to the "externalLink" attribute.

If these conditions are not met, the caster will not appear in the Caster selection list.

## Value to Task Link



If you need to synchronize a task relationship to a data enumeration, such as a project from CA PPM, to a single select field in Jira, use the Value to Task Link caster. This caster enables you to map an attribute value in one repository to a task link in another.

The mappings for this caster are configured through a table, where the values from the data enumeration are located on the left column, and the tasks associated with the values can be configured



by entering task IDs into the right column. To help you locate that desired tasks, you can assign a query under "Search for tasks in". Select or create a query that will return all the possible tasks you wish to map to. Selecting this query will enable content assist, allowing you to select tasks from a list instead of manually typing in task IDs. In addition, having a query selected will cause the table to display the task key and summary in the right column instead of the task ID. It is recommended to select a query as configuring the caster with content assist is easier than typing in task IDs. Note that the mapping configured will always be one-to-one.

The table will highlight rows which have not been mapped in yellow, and rows with unsaved changes in green. In the above example, "Epics" and "Tasks" are examples of values which are not mapped, and "Defects" is an example of an unsaved mapping.

In addition, the caster can be configured to "Map Using Labels" or to "Ignore Case During Mapping". When "Map Using Labels" is enabled, the underlying configuration in the XML will contain the labels of the values instead of the IDs. When "Ignore Case During Mapping" is enabled, the casing of the values will be ignored. For example, when "Ignore Case During Mapping" is disabled, and "Task" is mapped to a task with ID "1", then only "Task" will be mapped to task "1", and "task" will not. However, when "Ignore Case During Mapping" is enabled, "task", "TASK", and "Task" will all map to task "1".

This caster can only be selected for a mapping which meets the following conditions:

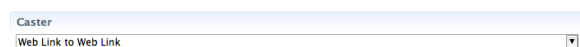
1. One attribute in the mapping must be of the type "Single Select" or "Multi Select".
2. One attribute in the mapping must of of the type "Task Dependency"

If these conditions are not met, the caster will not appear in the Caster selection list.

Note: To determine the attribute type, click on the repository [schema](#) page, and click on the desired attribute.

Note: An attribute which supports multiple values may be synchronized with an attribute which only supports single values as long as the attribute supporting multiple values contains at most one value. An error will occur when attempting to synchronize multiple values into an attribute which only supports single values.

## Web Link to Web Link



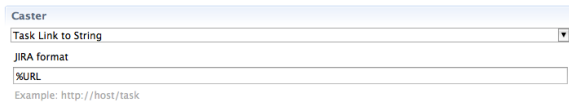
To synchronize referenced web URLs between repositories, use the Web Link to Web Link caster. With this caster, you can synchronize one or more attached web URLs between two tasks. This synchronization can be either bidirectional (web URLs attached to either task will be synchronized to the other) or unidirectional (only web URLs attached to to the task in one of the repositories will be added to the other).

This caster can only be selected for a mapping that meets the following condition:

1. Both attributes in the mapping must be of the type "externalLink". (To determine the attribute type, click on the repository [schema](#) page, and click on the desired attribute.)

If this condition is not met, the caster will not appear in the Caster selection list.

## Web Link to String



Caster  
Task Link to String  
JIRA format  
%URL  
Example: http://host/task

If you need to synchronize web URLs from one repository to another, but your target repository does not support attached web URLs, use the Web Link to String caster. With this caster, you can synchronize one or more web URLs, formatted into text attributes, into your target repository.

This caster also supports formatting of the target text attribute. Pressing `Ctrl-space` will enable content-assist when filling out the format. See [Web Link To String Caster](#) in the advanced [Transforming Task Attribute Values with Casters](#) section for more details.

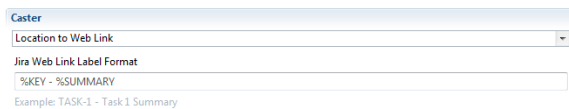
This caster can only be selected for a mapping that meets the following conditions:

1. One attribute in the mapping must be of the type "externalLink". (To determine the attribute type, click on the repository [schema](#) page, and click on the desired attribute.)
2. One attribute in the mapping must support text entry.
3. The mapping must be unidirectional from the "externalLink" attribute to the text attribute.

If these conditions are not met, the caster will not appear in the Caster selection list.

Note: Since the resulting text will concatenate multiple web URLs, it can be quite large. It is recommended that the target text attribute support strings of at least 1024 characters.

## Location to Web Link



Caster  
Location to Web Link  
Jira Web Link Label Format  
%KEY - %SUMMARY  
Example: TASK-1 - Task 1 Summary

If you need to have the URL of the source task as a web link on the target task, use the Location to Web Link caster. This caster will create a web link, which points to the source task, on the target task.

This caster also supports formatting the label of the created web link. Pressing `Ctrl-space` will enable content-assist when filling out the format. See [Location to Web Link Caster](#) in the advanced [Transforming Task Attribute Values with Casters](#) section for more details.

This caster can only be selected for a mapping that meets the following conditions:

1. One attribute in the mapping must be of type "externalLink".
2. One attribute in the mapping must be of type "url".
3. The mapping must be unidirectional from the "url" attribute to the "externalLink" attribute.

(To determine attribute type, click on the repository [schema](#) page, and click on the desired attribute.)

If these conditions are not met, the caster will not appear in the Caster selection list.

Note: If the link label is configured to contain the task summary, it is recommended to use a mapping strategy of "always" to ensure the label is always up to date. For more details on setting the strategy, see the [Advanced](#) section.

## Status Transition

If you need to synchronize status into a repository with a workflow, use the Status Transition caster. With this caster you can synchronize status by executing transitions to move the target task between statuses following the workflow defined in the remote repository.

This caster can only be selected for a mapping that meets the following conditions:

1. The target attribute is the Status attribute.
2. The mapping must be unidirectional.

If these conditions are not met, the caster will not appear in the Caster selection list.

To configure this caster both a value mapping and a transition graph must be specified. The value mapping is used to translate the status of the source task into a desired status on the target task. The transition graph is used to determine the sequence of transitions which must be executed to attain the desired status.

**Status Mapping**

Filter values...	ALM	JIRA
		Open
	Closed	Closed
	Fixed	Resolved
	New	Open
	Open	In Progress
	Rejected	Closed
	Reopen	Reopened

**JIRA Status Workflow**

From Status	Transition	To Status
<input type="checkbox"/> Open	Start Progress [connector.4]	In Progress
<input type="checkbox"/> Open	Resolve Issue [connector.5]	Resolved
<input checked="" type="checkbox"/> Open	Close Issue [connector.2]	Closed
<input type="checkbox"/> In Progress	Stop Progress [connector.301]	Open
<input type="checkbox"/> In Progress	Resolve Issue [connector.5]	Resolved
<input checked="" type="checkbox"/> In Progress	Close Issue [connector.2]	Closed
<input type="checkbox"/> Closed	Reopen Issue [connector.3]	Reopened
<input checked="" type="checkbox"/> Resolved	Close Issue [connector.201]	Closed
<input type="checkbox"/> Resolved	Reopen Issue [connector.3]	Reopened

Status on Creation:

The mapping from source repository status to target repository status is configured using the "Status Mapping" table on the left. For each source status a corresponding target status should be configured.

The status transitions are configured using the "Status Workflow" table on the right. For every possible transition between statuses in the target system, a row must be configured in this table. Each transition must be configured with a status from which the transition may be applied, as well as the status which will result from applying the transition. Some transitions may have more than one valid status from which the transition can be applied. A row in the table must be configured for each of these possible statuses. Similarly there may be multiple transitions originating from a status, or resulting in a status. Because of this the same status or transition may appear multiple times in the table.

To add rows to the table click the "Add" button and a new unconfigured row will be added. To remove rows from the table they must be selected using the checkboxes in the table, then the "Remove Selected..." button can be used to remove them.

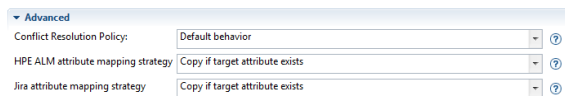
The configuration may be partially generated by clicking the "Suggest..." button. This button will add rows for all known status and transition combinations. Tasktop Sync is aware of which transitions may be executed from a given status, however it is not aware of what status the transition will result in. This information must be configured manually. Some statuses may not appear in the table after the "Suggest..." button is clicked. Details on which transitions are known to Tasktop Sync may be found in the [Schema View](#).

In some repositories, an initial status must be defined when creating a new task. This status represents the initial state of the task. To configure, select the starting status beside "Status on Creation".

For further details on the behaviour of the Status Transition caster see [Status Transition Caster](#).

If you need to synchronize workflows in both directions, create two attribute mappings with this caster, one in each direction.

## Advanced

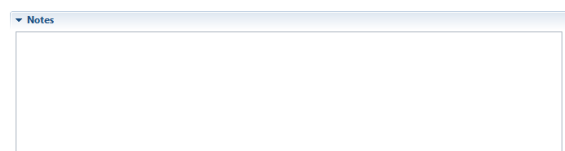


▼ Advanced	
Conflict Resolution Policy:	Default behavior ⓘ
HPE ALM attribute mapping strategy	Copy if target attribute exists ⓘ
Jira attribute mapping strategy	Copy if target attribute exists ⓘ

The `Conflict Resolution Policy` box specifies the policy for synchronization of this attribute in case of a conflict. If `Default behavior` is selected, the policy defined in the task mapping will take effect. Selecting any other option will override the mapping-wide default behavior. See the [resolution policy in the mapping settings](#) for details.

The `Attribute mapping strategy` box specifies when this attribute should be synchronized. The default behavior is `Copy if target attribute exists`, which only synchronizes the attribute if it exists in the other repository, so undefined attributes will not be created. `Initialize` will only create the attribute on the creation of a new proxy task, and will not synchronize in subsequent synchronizations. `Always` will always copy the attribute to the target regardless of whether a change has been detected on the source attribute.

## Notes



▼ Notes
<div></div>

This section allows for entry of any documentation or additional information about the attribute mapping. The value of this field is not used by Tasktop Sync directly.

## Transition Attribute Mappings Table

Transition Attribute Mappings may need to be configured if a [Status Transition](#) caster has been configured. These mappings write to attributes required during a transition, such as a **resolution**

attribute for a transition between the **In Progress** state and the **Done** state. As transition attributes are unique to a specific transition, there may be multiple mappings for a given transition attribute; one for each configured transition. For example, consider two configured transitions. The first is a transition between the **New** state and the **Done** state, the second is a transition between the **In Progress** state and the **Done** state, and both have the **resolution** transition attribute. Separate Transition Attribute Mappings are required for mapping the **resolution** attribute within the two transitions. Configuring the Transition Attribute Mappings, as described below, is more or less identical to configuring the [Attribute Mappings](#).

ALM	JIRA
Planned Closing Ver...	Fix Version/s [fixVersions]
<None>	Resolution [resolution]

The Transition Attribute Mappings table displays mappings between a task attribute and a transition attribute. The add button creates a new mapping for the selected transition within the dropdown. The dropdown is populated with all transitions within the current task mapping which are configured in a status mapping. All transition attribute mappings are unidirectional; the desired value will always be written into the transition attribute. The minus button will remove the selected transition attribute mapping.

## Transition Attribute Mapping

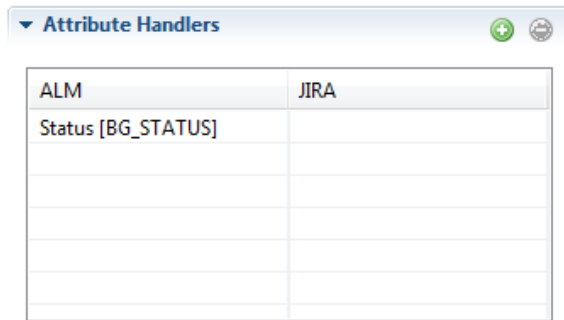
ALM	JIRA
Cannot Reproduce	Cannot Reproduce
Duplicate	Duplicate
Fixed	Fixed
Incomplete	Incomplete
Won't Fix	Won't Fix

JIRA	ALM
Cannot Reproduce	Cannot Reproduce
Done	Fixed
Duplicate	Duplicate
Fixed	Fixed
Incomplete	Incomplete
Won't Fix	Won't Fix

Map Using Labels ☐

Selecting a transition attribute mapping in the table will bring up the mapping editor, where the transition attribute mapping may be configured. The left and right columns display the attributes being mapped. One of the attributes is a transition attribute, and the other is a task attribute. The middle column displays the transition and the direction of the mapping. Casters may be configured for transition attribute mappings in the same way as the task attribute mappings. With the exception of the group casters, all casters may be used in transition attribute mappings. More information on casters can be found [here](#).

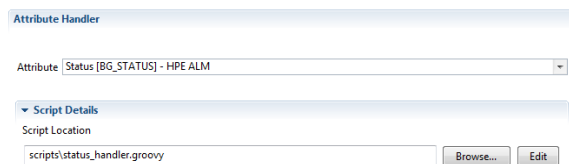
## Attribute Handlers Table



ALM	JIRA
Status [BG_STATUS]	

The Attribute Handlers table displays attributes that trigger scripted attribute handlers. The plus and minus buttons on the top right allow the addition and removal of attribute handlers. For details on how attribute handlers work, and on how to write attribute handlers, see [Scripted Attribute Handlers](#).

## Attribute Handler



Attribute Handler

Attribute | Status [BG\_STATUS] - HPE ALM

Script Details

Script Location

scripts/status\_handler.groovy

The `Attribute` identifies an attribute in one of the repositories that will be associated with an attribute handler. When this attribute value changes, it will trigger the script. If an attribute with the chosen identifier exists in both repositories, changes in either repository will trigger the script. To select, choose the attribute from the dropdown list.

The `Script Location` points to the script file in the workspace that will be executed when the attribute changes. To change the file location, click `Browse...` and locate the file in the workspace. To see the contents of the chosen file, click `Open`.

Note that the `Script Location` may contain `[ Embedded Script ]`, which indicates that the script is not in an external file, but instead inside the `synchronizer.xml` file.

For details on how to write attribute handlers, see [Scripted Attribute Handlers](#).

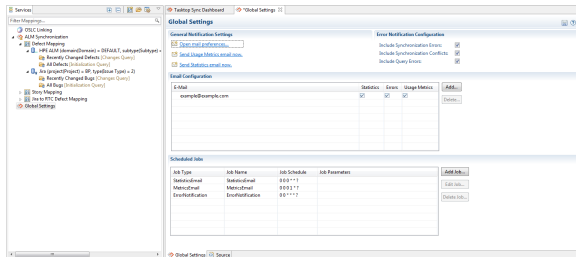
## Schema View

The `Schema View` can be accessed by selecting the links provided under the `Schema` section of the Task Mapping Editor. Tasktop Sync inspects both repositories to determine their schemas, including the attribute IDs, their labels, and their types. The Attribute Mapping Editor uses these schemas to populate the options for many attributes, including enumerated types used when producing value map casters and literal casters.



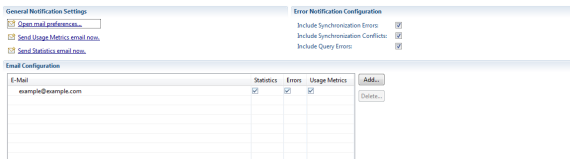
# Global Settings

To edit configuration that affects all task mappings, including Scheduled Jobs and Notifications, use the "Global Settings" editor.



To access the Global Settings Editor, double-click on `Global Settings` in the Services view.

## Notifications



There are three kinds of notification emails:

- **Statistics** emails are sent by the `StatisticsEmail` job, initially scheduled to send daily at midnight. The emails show detailed information about how much activity there has been in the last 24 hours. An explanation of the 24-hour average statistics can be found at [The Dashboard](#).
- **Error** emails are sent by the `ErrorNotification` job, initially scheduled to send hourly. Errors which occur during synchronization are collected and sent periodically according to the configured schedule of the [Error Notification Job](#). The contents of the error email can be configured using the checkboxes in the [Error Notification Configuration](#) section.
- **Usage Metrics** emails are sent by the `MetricsEmail` job, initially scheduled to send on the first of each month. See the [Usage Metrics](#) section for more information.

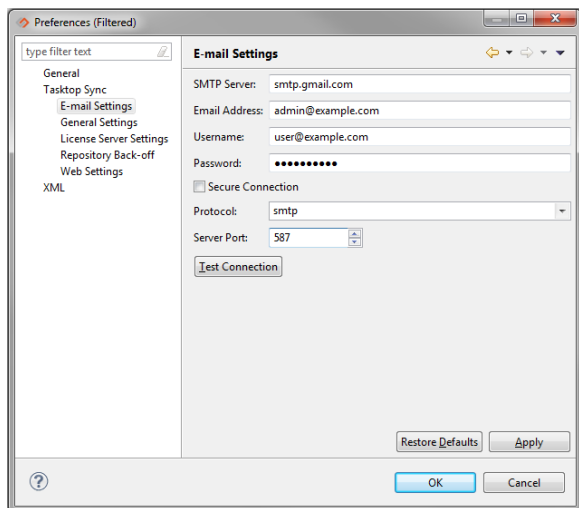
Details on configuring the jobs to run on different schedules can be found in the section [Configuring Scheduled Jobs](#).

If an error happens while connecting to a repository, an additional error email will be sent. These will only be sent once per repository in error. These errors should be resolved as quickly as possible since they will block Tasktop Sync from operating correctly; Tasktop Sync will not attempt synchronizations involving repositories in error.

## General Notification Settings

To configure the mail preferences, click on `Open mail preferences...`





Enter the details of your email to the dialog that pops up. Tasktop Sync notifications will be sent from the address configured here.

If an instance name has been set, it will appear in the subject of the notification email. More information on instance names can be found [here](#).

Clicking `Send Usage Metrics email now.` or `Send Statistics email now.` will send the respective email now to any users configured to receive them.

## Error Notification Configuration

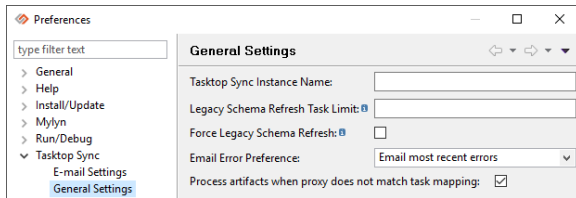
There are three different errors that may be contained within the Error email:

1. **Synchronization Errors:** Errors that occurred during synchronization, which may include conditions such as: a proxy task has been deleted, a repository is offline, or a task submission has failed.
2. **Synchronization Conflicts:** Conflicts that occurred during synchronization which resulted in the changed attribute not being synchronized.
3. **Query Errors:** Errors that resulted from queries being run by Tasktop Sync.

Tasktop Sync can be configured to email all errors or only the most recent errors:

- **Email all errors (default):** The default option is "Email all errors". If this is selected, Tasktop Sync will email all errors since the last Error Notification Job was run. If there are over 100 errors, Tasktop Sync will send multiple emails, with each email displaying at most 100 errors.
- **Email most recent errors:** If this option is selected, Tasktop Sync will only send one email containing the most recent 100 errors since the last Error Notification Job was run. If there were more than 100 errors in that period, the email will contain a URL link to the error page of the Tasktop Sync Web UI which displays all errors.

The Email Error preference can be selected on the General Settings tab:



## Email Configuration

The Email Configuration section allows the configuration of email addresses which will receive emails from Tasktop Sync.

Once a valid email address has been added to the list, clicking on the check boxes under "Statistics", "Errors", and "Usage Metrics" will determine which type of reports the recipient receives. Notification emails will not be sent without a valid email address provided in the mail preferences.

## Configuring Scheduled Jobs

The Schedule Jobs section allows the configuration of Tasktop Sync's scheduled jobs. For details on Tasktop Sync's usage of scheduled jobs see the section [jobs](#).

All of Tasktop Sync's scheduled jobs are displayed in the scheduled jobs table: the job's name, type and schedule are all displayed. If a job is in error then a red x will be placed in the table next to the job.

Scheduled Jobs			
Job Type	Job Name	Job Schedule	
StatisticsEmail	Daily Statistics	0 0 0 * * ?	
MetricsEmail	Monthly Metr...	0 0 0 1 * ?	
RepositoryConfi...	Refresh Local ...	0 0 0 * * ?	

Add Job...  
Edit Job...  
Delete Job...

Jobs may be added by clicking the "Add Job..." button. This will open a dialog to configure the new job. In this dialog the job's type, name, schedule and parameters may be configured. For the job to be added the job's configuration must be valid. Tasktop Sync will provide validation feedback as the job is configured.

### Create Scheduled Job

Set the name, schedule, and parameters for the Scheduled Job

Refreshes the repository configuration of the specified repository

Job Type: RepositoryConfigurationRefresh

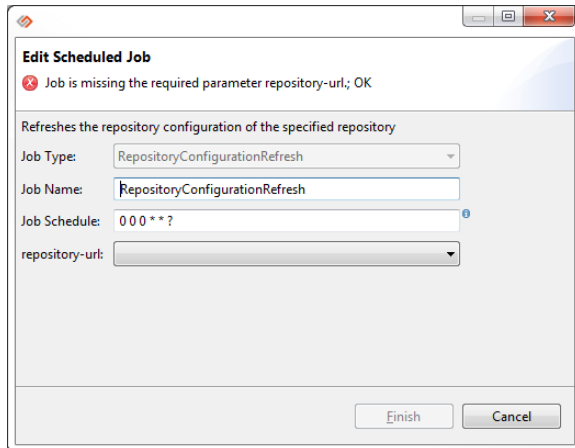
Job Name: RepositoryConfigurationRefresh

Job Schedule: 0 0 0 \* \* ?

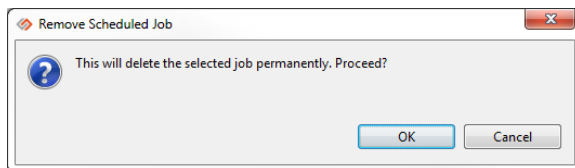
repository-url: local

Finish Cancel

Jobs may be edited by clicking the "Edit Job..." button with a job selected in the table, or double clicking a job in the table. This will open a dialog to configure the existing job. In this dialog the job's name, schedule and parameters may be configured. For the job to be saved the job's configuration must be valid. Tasktop Sync will provide validation feedback as the job is configured. A job's type may not be changed in the edit dialog.



Jobs may be deleted by clicking the "Delete Job..." button, this will open a dialog to confirm the deletion. Deleting a job will remove it entirely from the configuration.



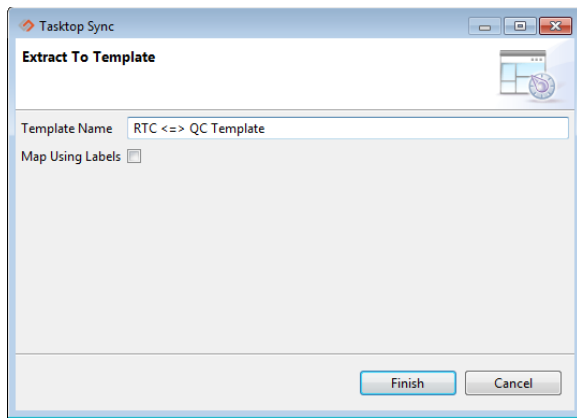
## Configuration Templates

After creating a task mapping, you may decide that you want to apply this mapping to multiple other project/workspace pairs. Configuration templates can help simplify this process.

### Creating a New Template from an Existing Mapping

To create a new template from an existing mapping:

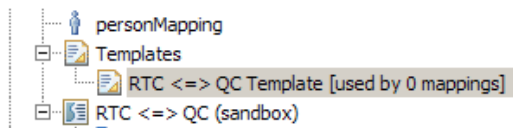
1. Find the task mapping from which you want to derive a template.
2. Right-click on the mapping in the Services View.
3. Select "Extract to Template".
4. Enter a name for the template in the dialog box that appears.
5. To create the template using attribute labels as identifiers, check "Map Using Labels". This is useful if attribute ids are not consistent between projects in a repository.



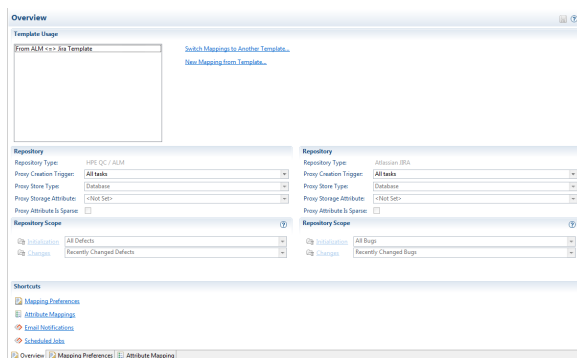
Note that the "Map Using Labels" checkbox may be disabled if it cannot be supported for the selected task mapping. To support labels, the task mapping must meet the following criteria:

- Tasktop Sync has repository schema for both repository scopes in the mapping
- There are no attribute mappings using group casters
- There are no attribute mappings using the Status Transition caster

Upon completion, the Templates bucket in the Services View will be populated with the new template.



The newly created template is shown in the Mapping Editor. Note that several fields are not editable.



## Create a New Task Mapping from a Template

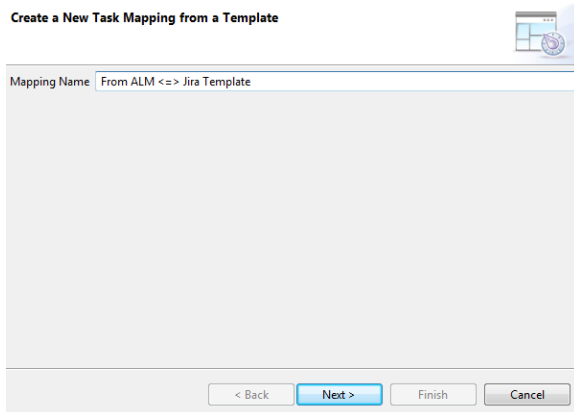
To create a new task mapping based on a template:

1. Right-click on the template in the Services View and select "New Mapping from Template...", or
2. Click on "New Mapping from Template..." in the template editor.

In the dialog box that pops up:

1. Designate a name for the new task mapping.
2. Select the repositories where the projects or workspaces you are synchronizing reside.
3. Define the queries and scope for the task mapping.

#### Create a New Task Mapping from a Template

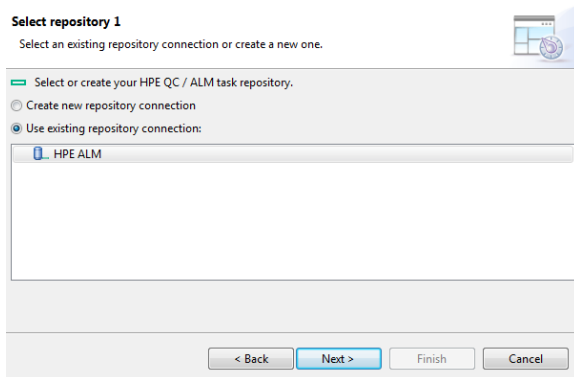


Mapping Name:

< Back   Next >   Finish   Cancel

#### Select repository 1

Select an existing repository connection or create a new one.



Select or create your HPE QC / ALM task repository.

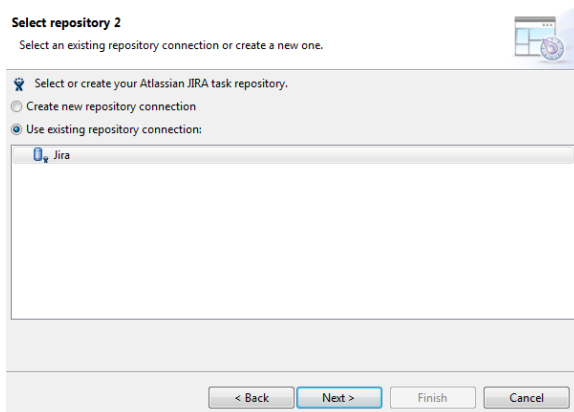
☐ Create new repository connection

☒ Use existing repository connection:

< Back   Next >   Finish   Cancel

#### Select repository 2

Select an existing repository connection or create a new one.



Select or create your Atlassian JIRA task repository.

☐ Create new repository connection

☒ Use existing repository connection:

< Back   Next >   Finish   Cancel

**Configure Mapping Source and Scope for HPE ALM**

HPE ALM

Initialization Query: All Defects 5 Tasks [New...](#)

Changes Query: Recently Changed Defects 5 Tasks [New...](#)

Domain:

Domain Project:

Type:

Subtype:

[Refresh](#)

[< Back](#) [Next >](#) [Finish](#) [Cancel](#)

**Configure Mapping Source and Scope for Jira**

Jira

Initialization Query: All Bugs 7 Tasks [New...](#)

Changes Query: Recently Changed Bugs 7 Tasks [New...](#)

Project:

Issue Type:

[Refresh](#)

[< Back](#) [Next >](#) [Finish](#) [Cancel](#)

The new task mapping will now be visible in the Services View and the editor for the mapping will be opened.

- From ALM <=> Jira Template
  - HPE ALM (domain(Domain) = DEFAULT, subtype(Subtype) = none, project(Doma
    - Recently Changed Defects [Changes Query]
    - All Defects [Initialization Query]
  - Jira (project(Project) = BP, type(Issue Type) = 2)
    - Recently Changed Bugs [Changes Query]
    - All Bugs [Initialization Query]

**Overview**

Template: [Use another Template...](#)

Based on template: [Default Mapping Template...](#)

[Disconnect from Template...](#)

**Repository**

URL: <http://dev-hpalm001.usm.testtop.com:8080/ps/alm>

Username/Password: testtop

Proxy Creation Trigger: All tasks

Proxy Store Type: Database

Proxy Storage Attribute: Check SaaS

Proxy Attribute Is Sparse: ☐

**Repository Scope**

Initialization: All Defects [New...](#)

Changes: Recently Changed Defects [New...](#)

**Artifact Scope**

Field	Value
subtype	none
domain	DEFAULT
project	PVT

[Edit...](#)

**Repository**

URL: <http://dev-jira001.usm.testtop.com:8080>

Username/Password: testtop

Proxy Creation Trigger: All tasks

Proxy Store Type: Database

Proxy Storage Attribute: Check SaaS

Proxy Attribute Is Sparse: ☐

**Repository Scope**

Initialization: All Bugs [New...](#)

Changes: Recently Changed Bugs [New...](#)

**Artifact Scope**

Field	Value
project	BP
type	2

[Edit...](#)

**Sidebars**

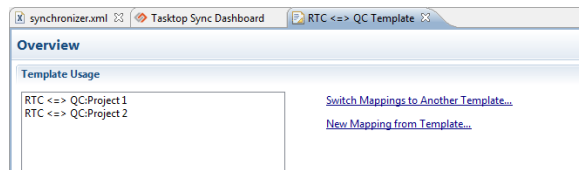
- [Mapping Preferences](#)
- [Attribute Mappings](#)
- [Email Notifications](#)
- [Scheduled Jobs](#)

[Overview](#) [Mapping Preferences](#) [Attribute Mapping](#) [HPE ALM schema](#) [Jira schema](#)

Notice that its mappings icon varies slightly to denote that it was configured based on a template. You can repeat this process multiple times, using your template to configure an unlimited number of task mappings that require the same configuration.

## Monitor Template Usage

In the Services View under Templates, you can see the templates that have been created and the number of task mappings that are using each template. Clicking on a template will open the template editor; there you can see mappings which use the template, as shown in the screenshot below.

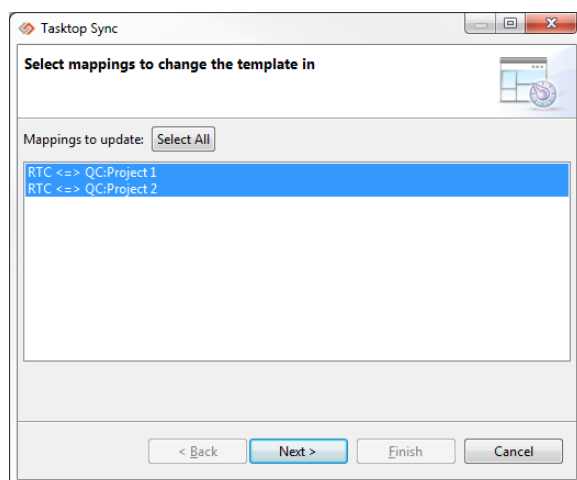
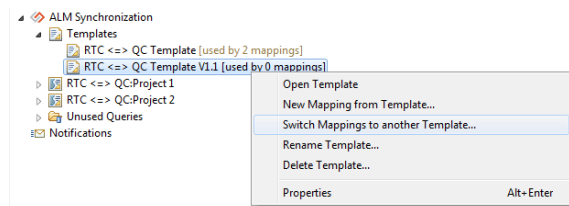


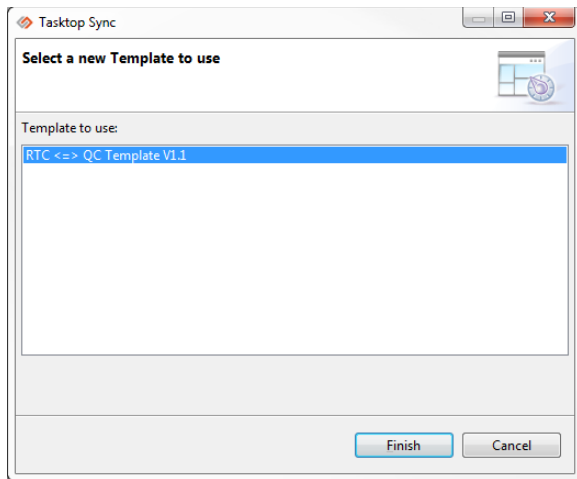
Alternatively, to see the template on which a task mapping is based, open the mapping in the editor, and click on the "Based on template" link.

## Making Changes to Task Mappings that Are Based on the Same Template

After creating a template and applying it to task mappings, Tasktop Administrators often end up needing to change or expand that template and apply it to all of the affected task mappings. To accomplish this:

1. Adjust the original task mapping that was used to create the template. This could include adding new attribute mappings, or configuring comment and attachment synchronization.
2. Repeat the process of creating a new template.
3. Right-click on the old template and select "Switch Mappings to another Template..."
4. In the dialog box, specify one or more task mappings to change and the new template to apply.





After creating the new template, there is another way to change the template upon which a single task mapping is based:

1. Select the task mapping which uses the old template.
2. Right-click and select "Use another Template".
3. This will start a dialog similar to above, but without the need to select the mapping.

## Disconnecting a Task Mapping from its Template

If a task mapping that was created using a template ever requires a unique synchronization configuration, Tasktop Sync administrators have the ability to disconnect it from a template:

1. Right-click on the mapping.
2. Select "Disconnect from Template".

(This action is also available in the Mapping Editor of the Task Mapping.)

The attribute mapping for this task mapping will remain as it was defined under the template, but will now be editable so that administrators can change and update it freely.

## Making Changes to a Task Mapping Based on a Template (Advanced)

Tasktop Sync administrators can add attribute mappings to a task mapping based on a template by editing the `synchronizer.xml` file. An attribute mapping defined in the task mapping can override an attribute mapping from the template, disable an attribute mapping from the template, or extend the template.

An attribute mapping defined in a task mapping will override an attribute mapping from the template if it involves at least one the same attribute as the attribute mapping from the template and it is a writable attribute (its strategy is not set to `ignore`). Note that this allows overriding the source attribute in a one-way attribute mapping from the template.

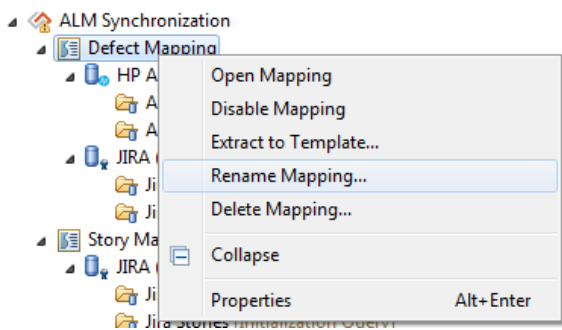


An attribute mapping in a task mapping will disable an attribute mapping from the template if it involves the same attributes as the attribute mapping from the template and neither attribute in the attribute mapping in the task mapping is writable (their strategies are set to `ignore`).

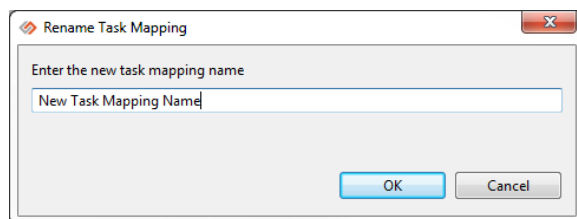
An attribute mapping defined in a task mapping will extend the template if it does not involve attributes present in the attribute mappings in the template, or it involves exactly one non-writable attribute from an attribute mapping in the template.

## Renaming a Task Mapping

You can rename a Task Mapping or a Template from Sync Studio. Before renaming the mapping, make sure you have saved all changes in Sync Studio. To rename the mapping, right click on the Task Mapping or Template in the Services view and select "Rename Mapping...".



From the "Rename Task Mapping" dialog, enter the desired name and click "OK".

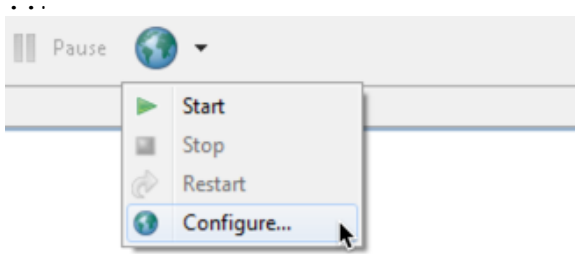


## Web Container Configuration

The web container in Tasktop Sync serves both the monitoring web UI and the OSLC Linking service. Starting and stopping the web container will start and stop both of those services (though the OSLC Linking service can be configured to be disabled, if desired).

1. Start Tasktop Sync.

2. Click on the pull-down menu to the right of the globe icon in the toolbar and select `Configure`.



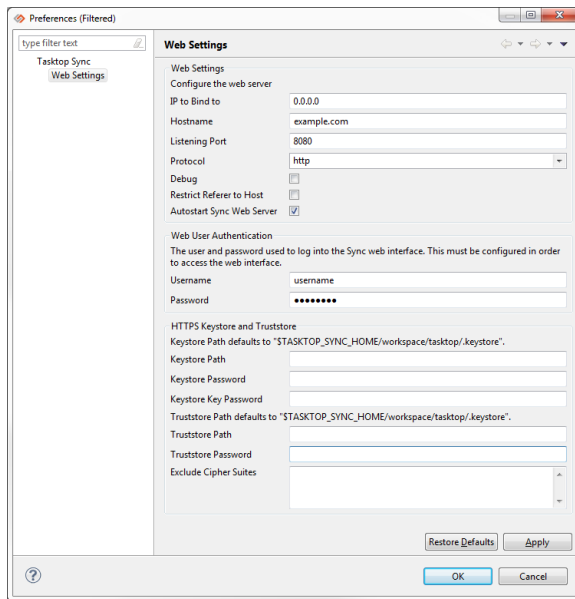
3. Several of the settings combine to form a URL that web browsers and other repositories will use to find the computer where Tasktop Sync is running.
  - For `IP to Bind to` (Required), you need to choose which IP address (of several that your server might have) that the web container should bind to. If you set `bindIP` to `0.0.0.0`, it will bind to all configured IP addresses on the computer.
  - For `Hostname` (Required), you must pick a hostname that is resolvable by DNS. If configuring OSLC Linking, the hostname set here must not change once task relationships begin to be persisted. Users will need to be able to reach it when they link tasks. Do not use "localhost" as your hostname.
  - For `Listening Port` (Required), you can choose any port which does not conflict with another port and which will not be blocked by security software.
  - For `Protocol` (Required), use the URL protocol (probably `http`) that users will connect to your computer with.
  - For `Debug`, set the checkbox to enable low-level debugging output. Note that setting this option will redirect all Tasktop Sync's error output (stderr) to the Jetty debug log file in the logs directory. It will also output full exception stack traces on Rest service fault responses.
  - For `Restrict Referer to Host`, set the checkbox to reject HTTP requests with a referer header different from the host header. This is to help protect against Cross Site Request Forgeries. Note that this setting can cause problems if you have a reverse proxy running in front of the Tasktop Sync web container.
  - For `Autostart Sync Web`, set the checkbox to have the Tasktop Sync Web Container start when the Sync application opens. Note that Sync will not autostart unless the configuration is valid.
4. Fill in the "Web User Authentication":
  - `Username` (Required)
  - `Password` (Required)

If you will be using TLS/HTTPS (SSL), there are a few more properties that you need to set.

Note: You should read <http://www.eclipse.org/jetty/documentation/current/configuring-ssl.html> for information on generating a keystore from your existing certificates or on using keytool to generate self-signed certificates.

- `Keystore Path` (Required) is the file where the keystore will be kept. The keystore contains certificates that are used to encrypt traffic between the client and server. It defaults to `$TASKTOP_SYNC_HOME/workspace/tasktop/.keystore` if unspecified.
- `Keystore Password` (Required) is the password used to access the keystore. Note that as this is a Java .properties file, you do not need to quote the password.

- **Keystore Key Password (Required)** is the password for individual records. This may be the same as **Keystore Password**.
- **Truststore Path (Required)** is the file where the truststore containing trusted certificates will be kept. This information will be used to authenticate trusted remote users. It also defaults to `$TASKTOP_SYNC_HOME/workspace/tasktop/.keystore`. It can be the same as the **Key Store Path** or a different file if desired.
- **Truststore Password (Required)** is the password used to access the truststore. In most cases, it can be the same as the **Keystore Key Password**.
- **Exclude Cipher Suites** can be used to exclude undesirable cipher suites from the web server. Enter in each suite separated by a comma. For a list of all active cipher suites, use control-space to bring up content assist.



## Task Linking Configuration

This section documents the steps required to configure Task Linking.

To initially configure your OSLC Adapter, complete the following steps:

1. Start Tasktop Sync.
2. Edit the OSLC properties file within Tasktop Sync. (You can open this file by right-clicking on **OSLC Linking** in the Services view and selecting "Properties".)
  - You should leave `enabled` set to `false` (or unspecified) until you are satisfied that your configuration is correct. Afterwards, setting `enabled` to `true` will cause Tasktop Linking to start as soon as Tasktop Sync is launched.
  - `disableAuthentication` should be `false` if you want authentication and `true` if you want to disable authentication. (See the authentication note below.)

- For `authenticationRepositories`, give a space-separated list of URLs of configured task repositories that can be used to authenticate the user's name/password. (See the authentication note below.) URLs may have a trailing '\*', in which case a wildcard match is applied.
- For `allowedRemoteHosts`, give a space-separated list of IP addresses that can access the OSLC Adapter without authentication. If unspecified, all IP addresses must authenticate.
- `OAuth Consumer Secret` must be a randomized 40 character hex string, used to allow OSLC consumers to create an OAuth connection via Task Linking.

Next, start testing your configuration. Start, stop, and restart the web container using the drop-down menu under the web globe icon in the Tasktop Sync toolbar.

Once you are happy with your configuration, you can set the `enabled` property to `true` so that Task Linking will start automatically.

### Authentication Note:

Repositories accessed via SDK connectors (connectors whose names do not have "Pre-Sync Version x" appended to the end) cannot be used to authenticate OSLC users

In the unauthenticated mode (`disableAuthentication=true`), Task Linking will be open to any traffic and is unauthenticated. It is recommended that you not use this mode in production or on any network that is open to the public. However, it can be useful for testing and initial configuration in a secure network environment.

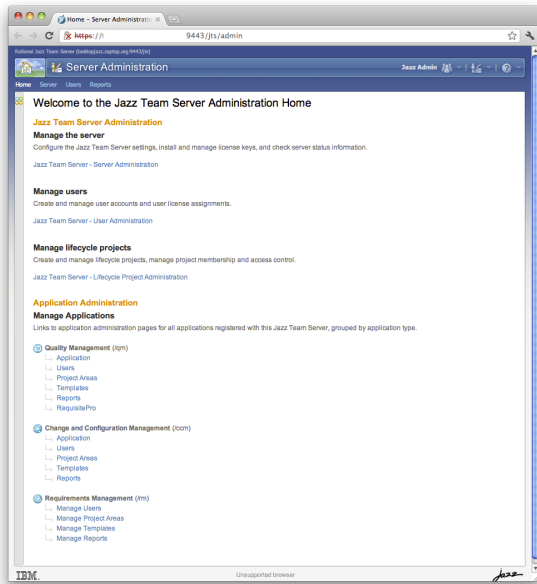
In the authenticated mode (`disableAuthentication=false`), Task Linking will present a login form to users and require that they authenticate using a username and password for one of the configured task repositories on the Tasktop Sync server. It is recommended that you run the service in this mode. The `oslcAdapter.authenticationRepositories` configuration field specifies which task repositories are used for authentication. If unspecified, Tasktop Sync will attempt to authenticate against all task repositories that are configured in the Tasktop Sync server until one validates.

## IBM Rational Team Concert Change Configuration Management (Change and Configuration Management) Configuration

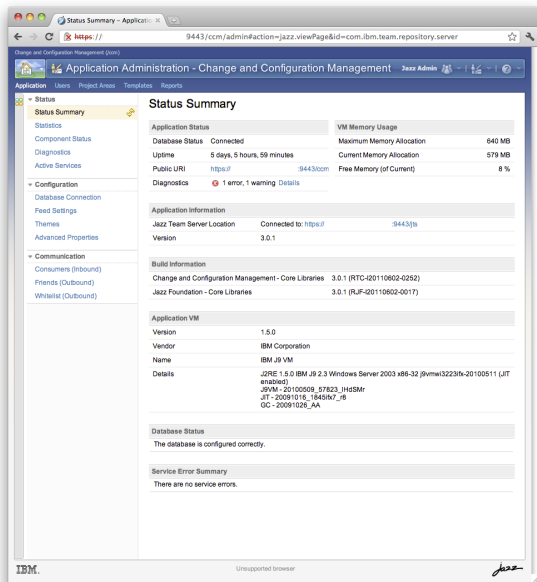
### Configure the Task Linking Server Relationship

To add Task Linking as a service that provides links to related tasks, complete the following steps:

Navigate to the Jazz Team Server Administration Home:



Click on Application under Change and Configuration Management to get to the Change and Configuration Management page:



Click on **Friends (Outbound)** section in the left sidebar. When the **Friends** page appears, click **Add** and complete the form using values from your `oslc.properties` file. These values must match the values in the `oslc.properties` file exactly.

The screenshot shows the 'Add Friend' form in the Jazz Administration console. The form is titled 'Add Friend' and has a 'Property' column and a 'Value' column. The fields are as follows:

Property	Value
Name	Enter a name to identify this entry in the friends list.
Root Services URI	The URI for root services on the server you want to add as a friend. Format: https://hostname:port-number/context-path/services
OAuth Secret	Enter a code phrase to be associated with the new OAuth consumer key from the friend server.
Re-type Secret	Re-type your code phrase to help prevent typos.
Trusted	<input checked="" type="checkbox"/> Trusted consumers will be able to share authorization with other trusted consumers and users will not be prompted for approval to access data. It is recommended that external web sites or products are considered as untrusted.

At the bottom of the form, there is a 'Create Friend' button and navigation buttons: '< Back', 'Next >', 'Finish', and 'Cancel'.

- Name: We recommend Tasktop OSLC Adapter.
- Root Services URI: protocol:// hostname:port/oslc/root services
- Consumer Secret: consumerSecret

The filled-in form will look something like this:

The screenshot shows the 'Add Friend' form with the following values filled in:

Property	Value
Name	Tasktop OSLC Adapter
Root Services URI	http://8080/oslc/root services
OAuth Secret	Enter a code phrase to be associated with the new OAuth consumer key from the friend server.
Re-type Secret	Re-type your code phrase to help prevent typos.
Trusted	<input checked="" type="checkbox"/> Trusted consumers will be able to share authorization with other trusted consumers and users will not be prompted for approval to access data. It is recommended that external web sites or products are considered as untrusted.

At the bottom of the form, there is a 'Create Friend' button and navigation buttons: '< Back', 'Next >', 'Finish', and 'Cancel'.

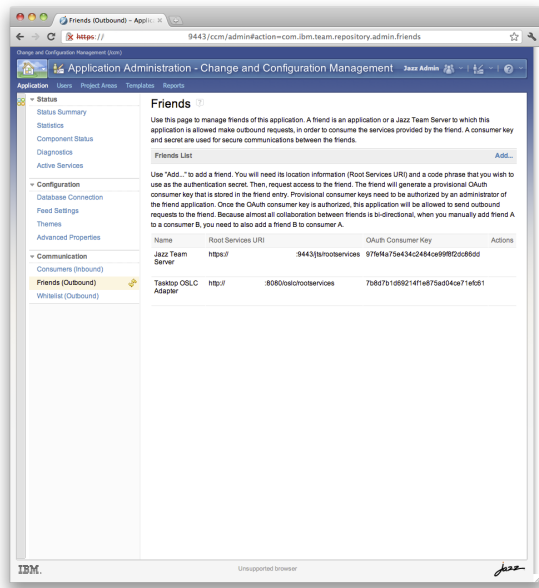
Click on the Create Friend button, and you will get confirmation that the service has been registered as a Friend:

The screenshot shows the 'Add Friend' dialog in the Jazzy Admin console. The dialog has a title bar with 'Friends (Outbound) - Application' and a URL bar showing 'https://9443/ccm/admin/inaction=com.ibm.team.repository.admin.friends'. The main content area has a green message bar at the top stating 'The Friend has been added and a provisional key has been generated. Please click Next to continue.' Below this, the 'Add Friend' form is displayed. It includes fields for 'Name' (with a placeholder 'Enter a name to identify this entry in the friends list'), 'Root Services URI' (with a placeholder 'http://8080/oss/rootservices'), 'OAuth Secret' (with a placeholder 'Enter a code phrase to be associated with the new OAuth consumer key from the friend server.'), and 'Re-type Secret' (with a placeholder 'Re-type your code phrase to help prevent typos.'). There is a checkbox labeled 'Trusted' which is checked, with a note: 'Trusted consumers will be able to share authorization with other trusted consumers and users will not be prompted for approval to access data. It is recommended that external web sites or products are considered as untrusted.' At the bottom of the form is a 'Create Friend' button. Below the form are navigation buttons: '< Back', 'Next >', 'Finish', and 'Cancel'.

Click on the Next button, and you will get a dialog which should confirm that the friend was created with a provisional 32-character hex key:

The screenshot shows the 'Authorize Provisional Key' dialog in the Jazzy Admin console. The dialog has a title bar with 'Friends (Outbound) - Application' and a URL bar showing 'https://9443/ccm/admin/inaction=com.ibm.team.repository.admin.friends'. The main content area has a title 'Authorize Provisional Key' and a message: 'The provisional key you requested needs to be authorized by an administrator on the other server.' Below this, the 'Provisional Key' is displayed: '7dbd7b1d89214f1e875a0d4ce71efdb1'. At the bottom of the dialog are navigation buttons: '< Back', 'Next >', 'Finish', and 'Cancel'.

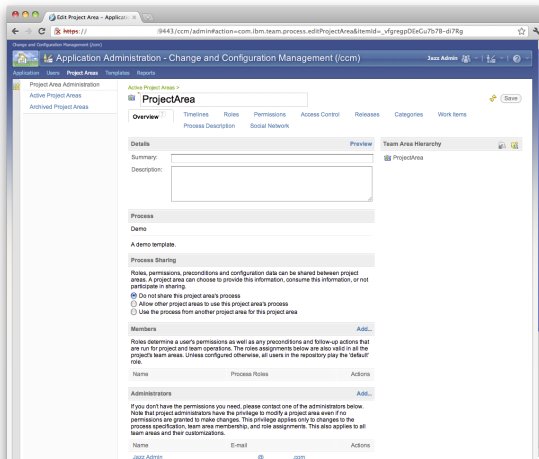
Click on the **Finish** button, and you will see the Tasktop OSLC Adapter service show up in the Friends List:



## Configure the IBM Rational Team Concert Change and Configuration Management Project Area

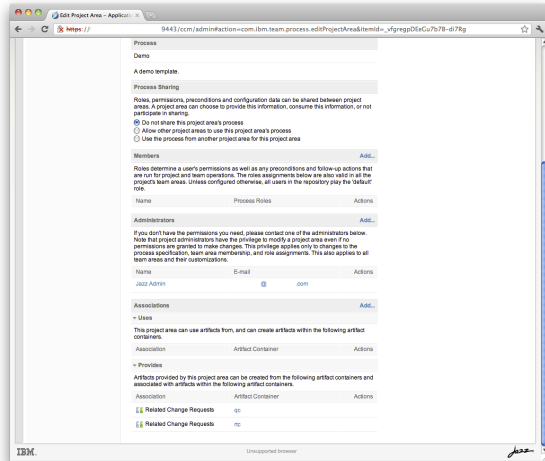
For each Project Area where you wish to enable the consumption of remote tasks using Task Linking, complete the following steps:

Navigate to the **Project Area Administration** Page on the top tab bar, then navigate to a project area:

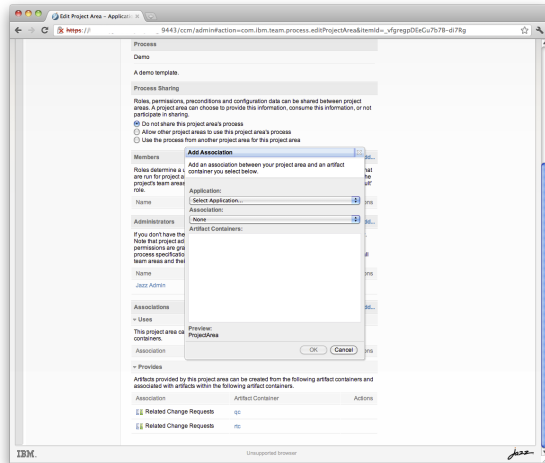




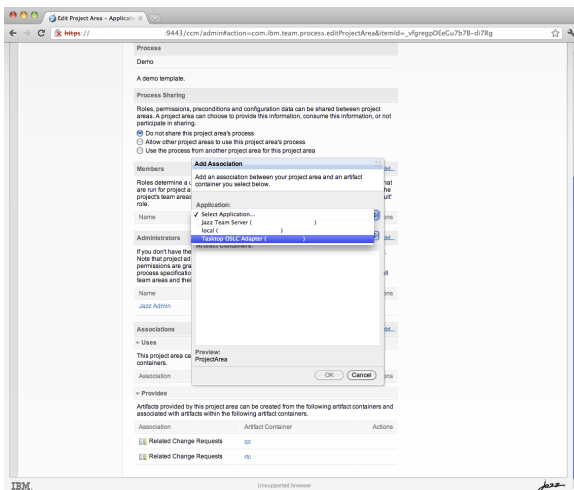
Navigate to the bottom section, Associations:



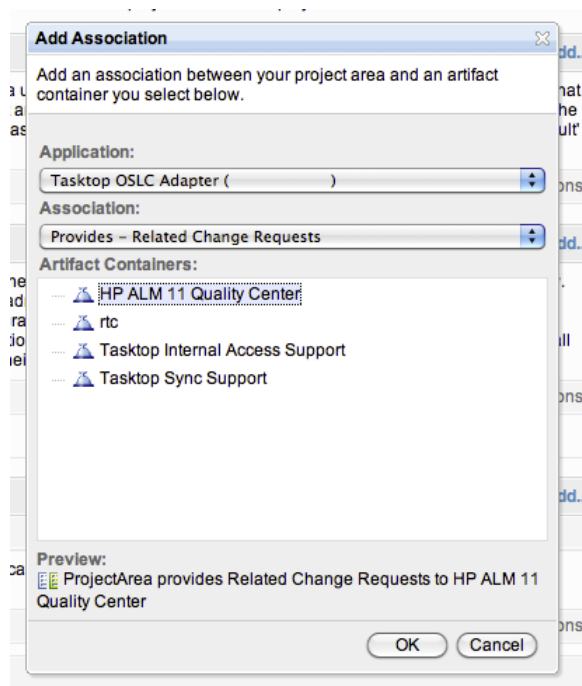
Click on Add . . . at the right of the Associations header bar:



Select an Application: "Tasktop OSLC Adapter" or whatever you named your server on the Friends page.



Select the desired Association (e.g. Provides - Related Change Requests) and the desired source Artifact Container (e.g. HPE ALM). The Artifact Container is where your remote task will be sourced from. The available Artifact Containers are the Task Repositories that are configured in Tasktop Sync.



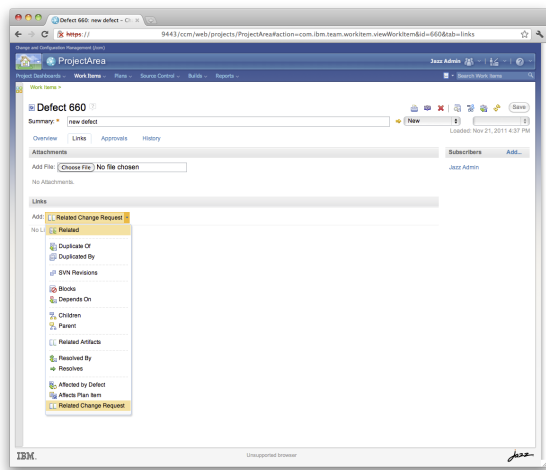
Click on OK, then click the Save button.

## Linking a IBM Rational Team Concert Change and Configuration Management Task to a Remote Artifact

To link a remote task to an IBM Rational Team Concert Change and Configuration Management task, complete the following steps:

Open a task (create one if you need to).

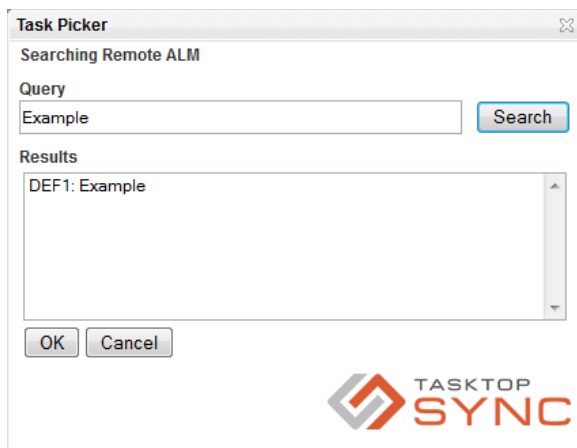
Navigate to the task's Links tab and select Related Change Request or another association name from the drop-down menu:



You might get a dialog which asks about secure content. Make sure that you allow mixed secure and non-secure content to be displayed.

Enter a search term and press Search.

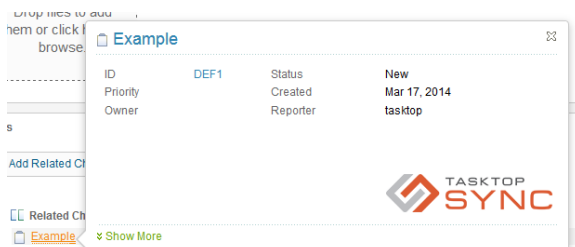
Select a search result you would like to link to.



Press OK.

Save the task.

You can now see a small preview of the task by hovering over the link:



You can see more detail by clicking the See more link:

Example

ID

DEF1

Status

New

Priority

Created

Owner

Reporter

Mar 17, 2014

tasktop

Attributes

URL Attachments

Domain

DEFAULT

Domain Project

Example

Type

Defect

Subtype

None

Actual Fix Time

226

Closed in Version

Closing Date

Detected in Release

Detected in Cycle

Detected in Version

Estimated Fix Time

Planned Closing Version

Project

Reproducible

Y

Severity

2-Medium

Sync Proxy


Target Release

Target Cycle

Linked Entities

Description

test



Show Less

You can click on the hyperlinked task ID in the ID field to open the target task in a new browser.

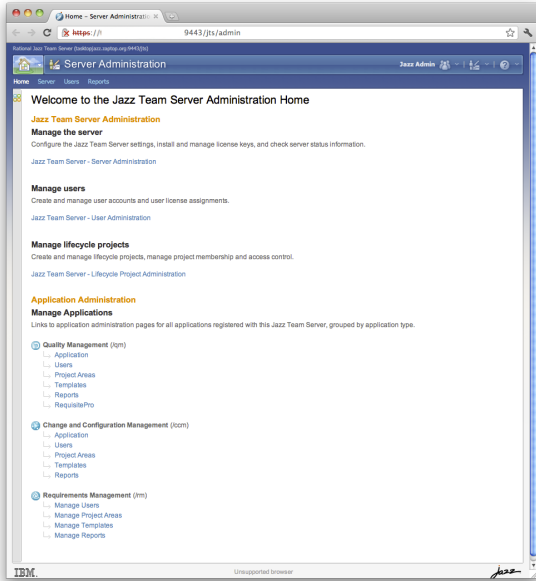
## IBM Rational Team Concert Requirements Management (Rational DOORS Next Gen) Configuration

💡 Please note: OSLC rich hover capabilities are not supported in DNG projects that have implemented configuration management.

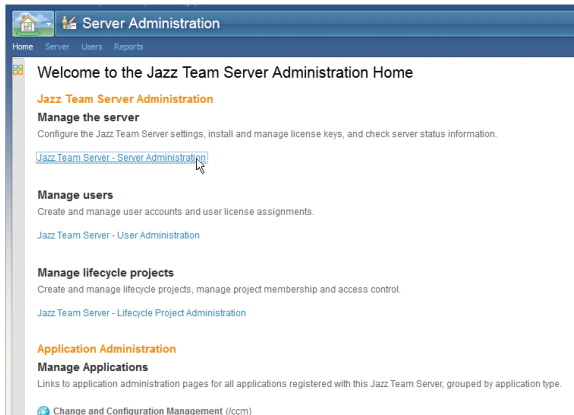
### Configure the Task Linking Server Relationship

To add Task Linking as a service that provides links to related tasks, complete the following steps:

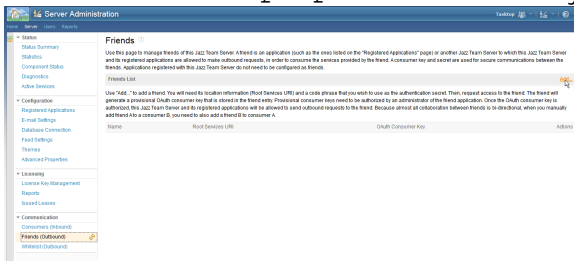
Navigate to the Jazz Team Server Administration Home:



Click on Jazz Team Server - Server Administration under Manage the Servers, and you will get to the Server Administration page:

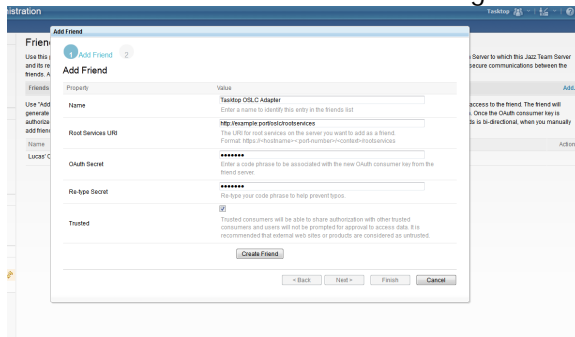


Click on Friends (Outbound) section in the left sidebar. When the Friends page appears, click and complete the form using values from your `oslc.properties` file. These values much match the values in the `oslc.properties` file exactly.



- Name: We recommend Tasktop OSLC Adapter.
- Root Services URI: `protocol://hostname:port/oslc/rootservices`
- Consumer Secret: `consumerSecret`

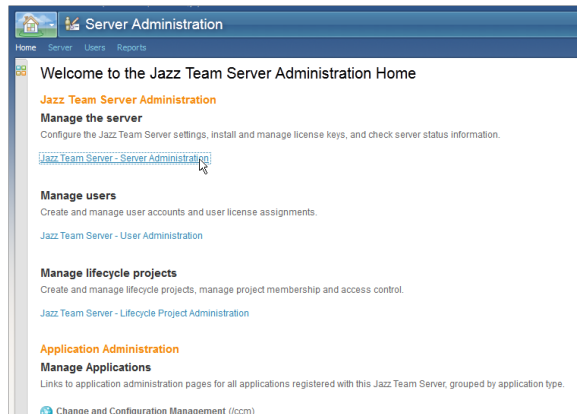
The filled-in form will look something like this:



Click on the **Finish** button, and you will get confirmation that the service has been registered as a **Friend**.

Click on the **Create Friend** button, and you will get a dialog which should confirm that the friend was created with a provisional 32-character hex key:

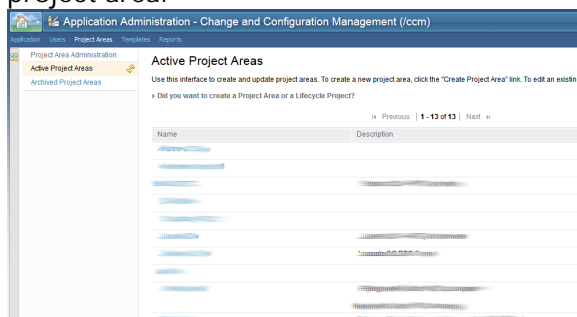
Click on **Finish**, and you will see the Tasktop Sync service show up in the Friends List:



## Configure the IBM Rational Team Concert Rational DOORS Next Gen Project Area

For each Project Area where you wish to enable the consumption of remote tasks using Task Linking, complete the following steps:

Navigate to the **Project Area Administration Page** on the top tab bar, then navigate to a project area:



Navigate to the bottom section, Associations:

Associations

Uses

This project area can use artifacts from, and can create artifacts within the following artifact containers.

Association	Artifact Container	Actions			
<div>Provides</div> <p>Artifacts provided by this project area can be created from the following artifact containers and associated with artifacts within the following artifact containers.</p> <table> <thead> <tr> <th>Association</th> <th>Artifact Container</th> <th>Actions</th> </tr> </thead> </table>			Association	Artifact Container	Actions
Association	Artifact Container	Actions			

Click on Add... at the right of the Associations header bar:

**Add Association**

Add an association between your project area and an artifact container you select below.

Application:  
Select Application...

Association:  
None

Artifact Containers:

Preview:  
Andrew QC Sync

OK Cancel

Select an Application: "Tasktop OSLC Adapter" or whatever you named your server on the `Friends` page.

**Add Association**

Add an association between your project area and an artifact container you select below.

**Application:**

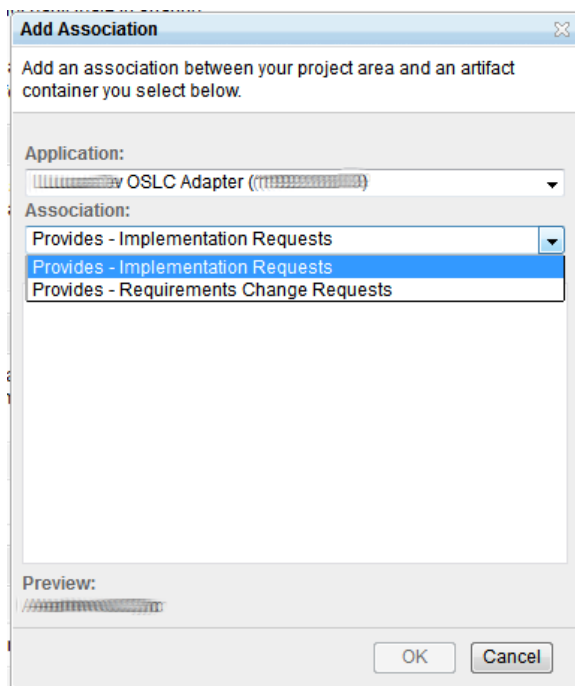
**Association:**

**Artifact Containers:**

**Preview:**  
 Andrew QC Sync

OK Cancel

Select the desired Association (e.g. Provides - Related Change Requests), and the desired source Artifact Container (e.g. HPE ALM). The Artifact Container is where your remote task will be sourced from. The available Artifact Containers are the Task Repositories that are configured in Tasktop Sync.



Click on OK, then click the Save button.

## Maintenance and Inspection

This section describes the tools provided by Tasktop Sync to ensure smooth operation and to allow inspection of synchronization and behaviour, as well as maintenance work that should be performed periodically.

### Upgrading an Existing Tasktop Sync Installation

Upgrading an existing installation of Tasktop Sync is simple and requires very minimal manual intervention. The upgraded version of Tasktop Sync will have access to all your existing repositories, queries and configurations.

To upgrade an existing installation, follows these steps:

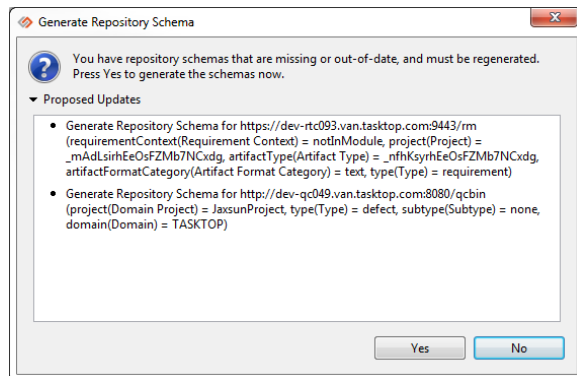
1. Backup installation customizations -; Installing Tasktop Sync will overwrite the following customizations you may have made to a previous install:
  - `TasktopSync.ini` -; Ensure any changes made to this file are backed up so they can be applied again after installing a new version of Tasktop Sync.



- Windows service log on credentials -; If there is an existing installation of Tasktop Sync as a Windows service, and the service is set up to run under a specific account, ensure that the account name and password are available to be entered again after upgrading to a new version.
2. Uninstall the existing Tasktop Sync version. Ensure that Tasktop Sync is not running before you attempt an uninstall.
    - a. In Windows: Uninstall using the Uninstall option in the start menu
    - b. In Linux: Run the uninstall script at <your Sync install location>/TasktopSync/scripts/uninstall.sh
  3. Install the new version of Tasktop Sync as described in the [Installation](#) section.
  4. Reapply installation customizations:
    - TasktopSync.ini -; Edit this file to apply any custom settings.
    - Windows service log on credentials – If you are running Tasktop Sync as a Windows service ensure that the log on details used from a previous installation are entered again.
  5. Complete the Tasktop Sync Post Installation Wizard (if required) as described in [Post Installation](#).
  6. Allow Tasktop Sync to generate new Repository Schemas (if required) as described in [Generate Repository Schema](#)

## Generate Repository Schema

New features in Tasktop Sync may require more detailed repository schemas. After upgrading, you may be prompted to regenerate these schemas. The prompt will list the schemas that must be generated, and ask you if you wish to generate them now.



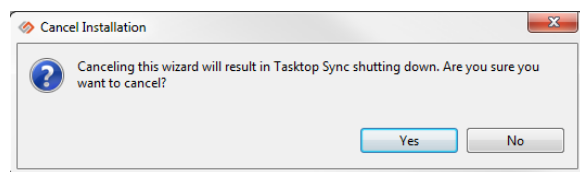
It is highly recommended that you generate the missing or out-of-date repository schemas before starting Tasktop Sync. Some features of Tasktop Sync will not function properly without valid schemas.

## Post Installation

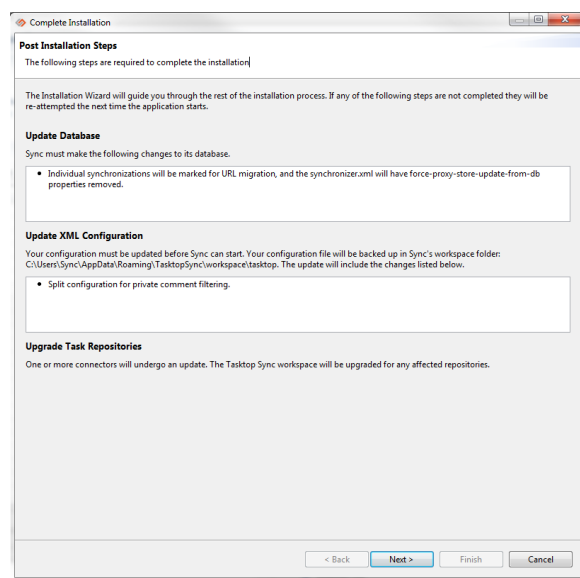
Sync may need to carry out the following required steps on installation upgrade:

- Perform database transformations
- Perform an automated configuration file update as described in [Configuration Updates](#).
- Upgrade task repositories

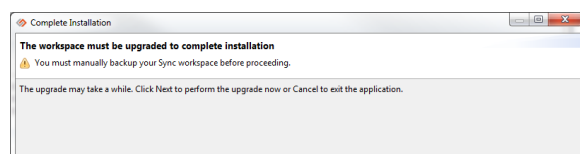
If any of these steps need to be completed, Sync will launch the Post Installation Wizard on startup. The wizard must be completed before Sync can be run. Clicking Cancel at any point in the wizard will bring up a prompt asking if Sync should be shut down. If "Yes" is chosen, the wizard will reappear on the subsequent startup.



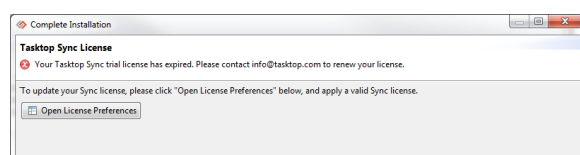
The first page of the wizard will list all required steps that will need to be performed.



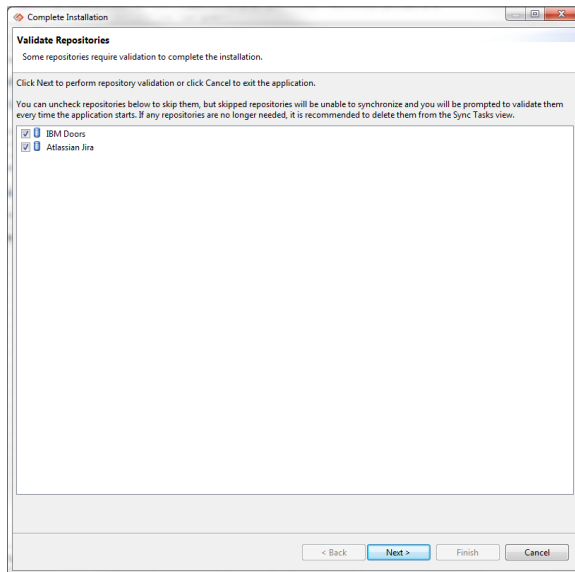
If an installation step requires a workspace backup, such as with the database transform and repository upgrade, a page will appear asking you to manually [back up](#) your workspace before proceeding.



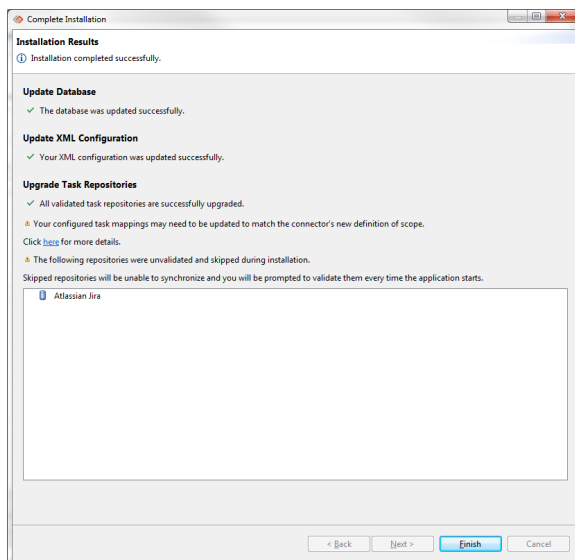
To proceed with the installation, Tasktop Sync requires a valid license. If there isn't one present, a page will appear to help you import a valid license. Click the "Open License Preferences" button, and the Tasktop Sync license preferences dialog will appear. Click the "Import from File..." button and select the license file you extracted from your installation package. This should populate the License dialog box with the details of your particular license. Finish the import by clicking the "OK" button.



Upgrading a task repository will require that the repository is validated. If any repositories needing upgrade require validation, a page will appear allowing you to select the repositories you wish to validate. If skipped, repositories will be unable to synchronize and you will be prompted to validate them every time the application starts.



After the installation finishes, the final page will notify you of the status of all completed post installation steps.



## Configuration Updates

A new version of Tasktop Sync will likely have more features and configuration options than your current version. Sometimes these features will require changes to your current configuration file. To make the configuration update process as smooth as possible, Tasktop Sync will attempt to migrate your configuration file for you automatically when you run the new version of Tasktop Sync.

Some changes may require that you manually edit your configuration after the update. This is the case if the new configuration contains a required attribute whose value must be specified by the user. The majority of configuration updates require no user intervention and will maintain the behavior of the previous Tasktop Sync configuration. Before updating the configuration, Tasktop Sync will back up your existing configuration for future reference.

## Upgrading from Pre-Sync XX Connectors (Advanced)

### Overview

Tasktop has recently changed some of the architecture of connectors to ensure that they are more consistent and can provide improved functionality. As there can be differences in the operation of the connector, the connectors are not automatically upgraded to the latest version and it is a manual process for the customer.

This document provides an overview of the steps that are required to upgrade to the latest connectors from the pre-sync XX connectors, however, may not contain all of the exact steps that are required for your environment. We highly recommend you run this in a test environment on representative mappings to ensure that the full set of steps can be created for your production environment. It is recommended to work closely with a Tasktop Solutions Architect to plan and test the procedure outlined in this document to ensure that the upgrade goes as smoothly as possible.

### Video Demo

## Upgrade Procedure

### Pre-requisites

Before upgrading the connectors, you should ensure that you have upgraded to the latest version of Tasktop Sync that is available. Please contact [support@tasktop.com](mailto:support@tasktop.com) or your solutions consultant to get the latest release. Additionally, you will need to ensure the following:

- A full backup of the workspace has been taken
- All tasks that have been synced at one point and will ever be synced in the future are part of a query (i.e., there will never be a situation in the future where a task that had been synced by the old connector was not upgraded because during upgrade time that task was not part of any query)

### Upgrade

The following steps must be performed in order to prepare the Tasktop Sync workspace for a connector upgrade:

- Synchronizer is stopped
- Web service is stopped

- OSLC linking service is stopped
- Configuration is valid
- All Task Editors and Task Mapping editors related to the repository to upgrade are saved and closed

Start the upgrade wizard from the Task Repositories view in Tasktop Sync. This can be run by selecting the repository that you want to upgrade and selecting “Upgrade To Latest Connector Version..”.

Follow the wizard to completion. This wizard will:

- Create the new repository
- Potentially update the mappings with new scoping information
- Update the internal database with the new proxy URL (as the format has changed) and ensure that the comment hashes are correct. This will ensure that no duplicates are created as a part of synchronization.

## Post-Upgrade Wizard

These tasks must be performed manually after the upgrade operation completes:

- Recreate all queries for the upgraded connector in Tasktop Sync
- Run queries and “Mark all as synchronized” from the Advanced menu in the Sync Tasks view for all queries after they complete execution so that they are not synced the next time that the Sync Job starts
- “Mark all for Synchronization” in the changes queries for synchronization from the Sync Tasks view to ensure that any changes that occurred while the upgrade was performed
- Run a local schema refresh from the Task repositories view
- Fix all task mappings using the upgraded connector
  - For each task mapping, click Edit beside the Artifact Scope section (Overview tab of mapping editor) and re-enter the desired scope (this will fix the scope in sync.xml behind the scenes)
  - Inspect all Attribute Mappings and ensure that they are correct as some field ids may have changed
  - Ensure that the correct casters are used (e.g. correct Text Markup types)
- Update any scripts Change any “connector kind” strings used in scripts
  - Other updates may be needed depending on the connectors – to be found through testing
  - Confirm that the mappings look correct otherwise via visual inspection
- Start Tasktop Sync and ensure that all data flows as expected

## Connector Specific Notes

### Atlassian JIRA

- All custom fields have different attribute ids and therefore the mappings need to be recreated for these fields
- All custom fields have different available values and therefore the mappings need to be re-created for these fields

- Common fields like description now have different attribute ids are therefore the mappings need to be recreated for these fields
- Rich text markup for comments and rich text fields should be set to JIRAConfluence

## HP ALM

- RQ\_FATHER\_ID has changed from an ID to a Task Link and will required re-mapping
- URL Attachments are now handled as Web Links and need to be mapped separately
- Rich text markup for comments and rich text fields should be set to HP ALM HTML

## IBM Rational Team Concert

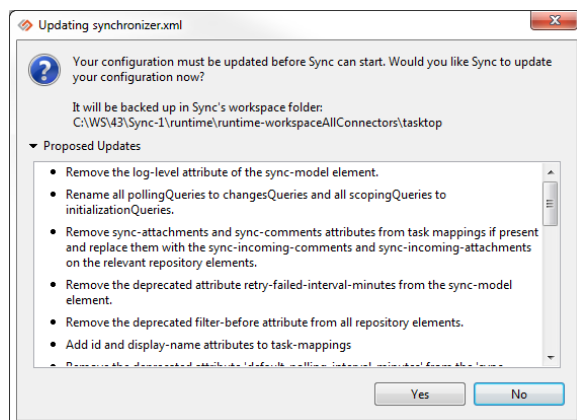
- The scope will need to be updated on each repository
- Task Relations can no longer be used for the proxy store. A custom field must be created and mappings updated to use this new proxy attribute.
- The internalState attribute can no longer be directly set and a status handler groovy script or status caster must be used to execute the correct transitions
- Rich text markup for comments and rich text fields should be set to IBM RTC HTML

## Microsoft Team Foundation Server

- Prior to upgrading, both the Team Explorer Everywhere plugin and the TFS SDK must be present. After upgrading, it is recommended to re-install Tasktop Sync without the Team Explorer Everywhere plugin to ensure that there are no conflicts
- Task Relations can no longer be used for the proxy store. A custom field must be created and mappings updated to use this new proxy attribute.
- Rich text markup for comments and rich text fields should be set to Microsoft TFS HTML

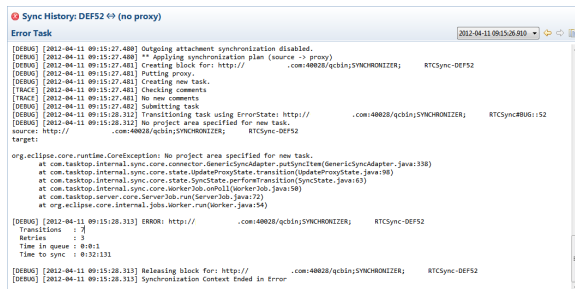
## Restoring a configuration without restarting Sync

If you revert your configuration to an older version, you will be prompted to update your configuration when starting the synchronizer, and the synchronizer will refuse to start unless you allow the configuration updater to run. The update prompt will outline the purpose of the configuration update and alert you to all the configuration changes that will be applied during the update. More details about configuration updates can be found [here](#).

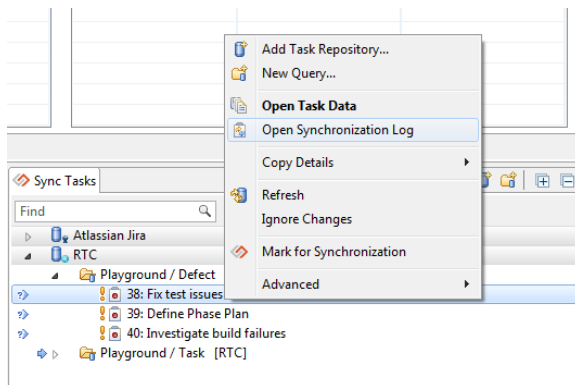


# Synchronization Log Viewer

The Sync Log displays logged console messages per synchronization activity for every task. It provides information that is easily accessible for debugging and may provide information in determining why a synchronization was unsuccessful.



There are two ways of accessing the Sync Log: either by right-clicking a task under a query in the Sync Tasks view and selecting **Open Sync Log**, or by double-clicking a failed synchronization in the Error Queue of the Sync Dashboard.



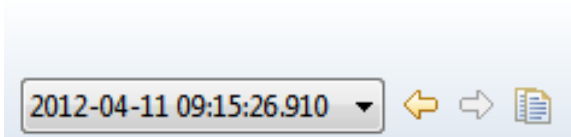
Error		
All Errors		
Source	Target	Message
DEF52: Error Task		No project area specified f...

The title is displayed in the top left of the Sync Log; this area displays the ID of each task being synchronized as well as its summary. Note that in this example, a red "x" icon is present in the upper left corner. The presence of this icon denotes that something went wrong during this synchronization, and manual intervention is required in order to successfully proceed. When viewing the log of a successful synchronization, this icon is absent.

## ❌ Sync History: DEF52 ⇔ (no proxy)

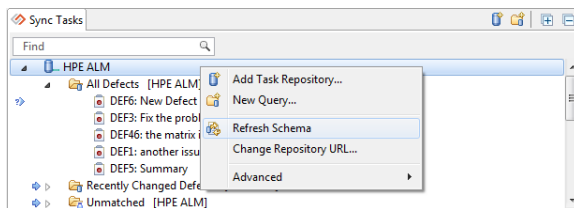
### Error Task

On the top right corner is the toolbar, which provides buttons to quickly navigate logs for previous synchronizations that this task has undergone. The timestamps in the dropdown are displayed in the current time zone and therefore can be different from the timestamps in the log file.



## Repository Configuration Changes

If the configuration for your repositories has changed, the repository schema must be refreshed for Tasktop Sync to operate correctly. To refresh the repository schema, open the Sync Tasks view, right-click on the repository in question, and select "Refresh Schema". When prompted, click "Yes" to get the latest repository data.



Alternatively, use the **Refresh Schema** button in the [Schema Viewer](#).

Refresh progress can be checked in the Progress view of Tasktop Sync. After refreshing the schema, the synchronization process must be restarted for changes to take effect. Some tasks may have been refreshed along with the repository schema, and if the synchronizer is running, any changes found may be synchronized.

## Legacy Schema Refresh

Most repositories can provide schema details from a single task, but some require that Tasktop Sync scan many tasks. This is referred to as "Legacy Schema Refresh". Refreshing the schema in this way may take some time. Additionally, refreshing the schema may run queries from the task repository.

## Periodically Refreshing the Repository Configuration

There is also an option periodically refresh the repository configuration. This is done by defining a scheduled job, which will perform a prespecified action based on the provided schedule.

There are two scheduled jobs which are associated with the Repository Configuration changes:



1. RepositoryConfigurationRefresh Job: This job will refresh the repository configuration of the specified repository
2. RefreshSchema Job: This job will refresh every repository schema associated with the specified repository

Details on how to to configure scheduled jobs can be found in the section [Configuring Scheduled Jobs](#).

## Repository Users

If a user is added to a repository, then that user must be mapped in the person mapping to ensure that they can be synchronized correctly to the other repository. Simply update the person mapping with a mapping containing the user's identity in each repository. Tasktop Sync must be restarted before the change to the person mappings will take effect. Read the [person mapping](#) section for more information on configuring person mappings.

## Service Outages

If either the Tasktop Sync server or one of the synchronized repositories goes down for a period of time that is longer than the changes query polling schedule, it is strongly recommended that the configured initialization query is run to discover all the changes made during the server downtime. To run the initialization query, locate it under the "Sync Tasks" view, right-click and choose "Refresh".

## Backup and Restore

All of the state maintained by Tasktop Sync, aside from password information, is stored in the Tasktop Sync workspace. Therefore, to perform backups of your Tasktop Sync data, the only requirement is that you back up this workspace directory. This section discusses one possible way of backing up the Tasktop Sync workspace, as well as how to restore this workspace on another machine if the primary Tasktop Sync machine becomes unavailable. See the final section on [Advanced Backup](#) for other possibilities.

### Archiving a Folder

The method for backing up and restoring described in this section requires that you archive a folder, and then restore that archive. This can be done by using a tool to create a zip archive file of the folder, and then using the same tool run unzip the archive file.

*Note: Windows zip compression has been found to skip files, and fail on large filenames, and is therefore not recommended.*

### Creating the Backup

This section describes how to use a zip archive to create a backup of the Tasktop Sync workspace.

See the [workspace directory](#) section for instructions on locating your workspace directory.

To create a zip archive backup, follow these steps:

- Open Windows explorer and locate your Tasktop Sync workspace as described in [workspace directory](#).
- Archive the folder into a file called `workspace.zip`.
- Copy or move the `workspace.zip` file created in the previous step to your backup media.

It is recommended that this process be automated and run on a schedule using your OS's task-scheduling application. The more up-to-date a backup is with respect to the current state of synchronization, the easier it is to restore that backup.

## Restoring a Backup

Restoring a backup of Tasktop Sync's workspace is a fairly simple procedure. Please see the following two sections depending on whether you are restoring a backup on the same machine where the backup was created or onto a secondary machine.

Note that there are a few things to consider when restoring a Tasktop Sync workspace backup. In particular, the most recently available backup may be quite old compared to the overall synchronization state of your repositories. For example, if you only create a backup once a week, and the machine running Tasktop Sync becomes unavailable six days after the last backup, then many synchronizations will have completed since the last backup. If this backup is being restored, then it is possible that Tasktop Sync will report spurious conflicts during synchronization while it catches up on processing all of the tasks that changed during that six-day period. The best way of dealing with this situation is to increase the frequency of backups to ensure that the most recent backup matches the most recent state of synchronization as closely as possible.

### Restoring a Backup on the Same Machine

To restore a backup saved in the file `workspace.zip` on the same machine where it was created, follow the steps below:

- Ensure Tasktop Sync is not running.
- Open Windows explorer and locate your Tasktop Sync workspace as described in the [workspace directory](#) section.
- If an existing workspace directory exists, rename it to `workspace.bk`.
- Unarchive the `workspace.zip` file. This should produce a `workspace` directory. Ensure that you don't get an extra layer of hierarchy when unarchiving: this step should not create a `workspace` directory inside of another `workspace` directory.
- Restart Tasktop Sync.
- Click the "Run Sync Queries" button to ensure that all modified tasks are found after restoring the backup.

### Restoring a Backup on a Secondary Machine

To restore a backup saved in the file `workspace.zip` on a different machine than the one where it was created, follow the steps below:

- Ensure Tasktop Sync has been installed.
- Ensure Tasktop Sync is not running.
- Open Windows explorer and locate your Tasktop Sync workspace as described in the [workspace directory](#) section.
- If an existing workspace folder exists, rename it to `workspace.bk`.
- Unarchive the `workspace.zip` file. This should produce a `workspace` directory. Ensure that you don't get an extra layer of hierarchy when unarchiving: this step should not create a `workspace` directory inside of another `workspace` directory.
- Start Tasktop Sync.
- For each repository in the Sync Tasks view:
  - Right-click the repository and select "Edit Repository" under the "Advanced" sub-menu.
  - Ensure the username and password has been entered, and click "Validate".
- Click the "Run Sync Queries" button to ensure that all modified tasks are found after restoring the backup.

Note that if you are also running OSLC, and restoring the backup on a secondary machine, you will have to repeat the OSLC configuration steps in your repositories to ensure that the IP address of the secondary machine is listed as a friend in your repositories. Please see the relevant sections in this Tasktop Sync documentation for configuring the secondary machine as a friend.

## Advanced Backup

The above sections described a simple way of backing up and restoring the Tasktop Sync workspace. However, Tasktop Sync is also compatible with other third-party backup solutions that your environment may have. Any third-party software that can archive the entire Tasktop Sync workspace will be effective at maintaining backups of Tasktop Sync's synchronization state.

For example, one effective way to back up Tasktop Sync's workspace so that a secondary instance of Tasktop Sync could be brought up quickly if the primary machine becomes unavailable is to use [rsync](#). `rsync` can be used to maintain a mirror copy of the main Tasktop Sync workspace on a second machine. If the main machine goes down, then the second machine will have an up-to-date copy of the workspace which can be used to continue synchronization.

## Logs and Logging Configuration

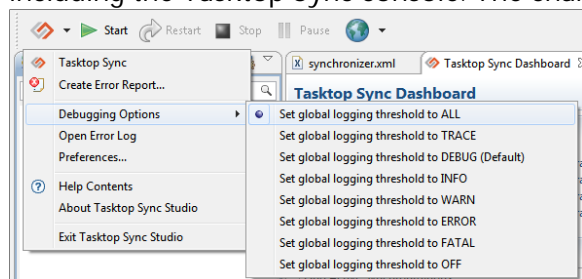
Tasktop Sync uses the Apache Log4j logging framework to maintain logs. By default, it keeps a single unfiltered log to make it easy for system administrators to diagnose problems with synchronization. The default location of the log file is `$TASKTOPSYNC_HOME$workspace\tasktop\log\tasktop-sync.log`. Log files are rotated and compressed on a daily basis.

Log files are never deleted by Tasktop Sync. It is up to the administrator to determine a retention policy for old log files. However, all log files that have been rotated and compressed may be moved or deleted by an administrator without affecting the functionality of Tasktop Sync.

Logging can be customized by creating the file `$TASKTOPSYNC_HOME%\workspace\tasktop\log4j.xml`, which follows the standard Log4j XML conventions. You can read the Log4j documentation for more information on how to configure logging with an XML file [here](#). If you wish to customize logging, a sample Log4j configuration file, `log4j-sample.xml`, is provided. To use this file, rename it to `log4j.xml` and make the required edits. Note that once a `log4j.xml` file is present, the logging of events to the `tasktop-sync.log` file described above is no longer enabled by default.

See [advanced configuration](#) for details regarding the value of `$TASKTOPSYNC_HOME%`.

While Tasktop Sync is running, you can temporarily change the amount of information being logged globally by changing the logging threshold. Log messages with a severity below the threshold will not be written to the log file or displayed in the console. To change the threshold, look for the Tasktop logo in the top left of your Tasktop Sync window. Select the down-arrow drop-down menu beside this icon and under "Debugging Options" select the desired logging threshold. This will override the threshold of the root logger, and so it will affect the log message output to all configured logging appenders, including the Tasktop Sync console. The changes will not be saved when you exit Tasktop Sync.



Please note that repository passwords are stripped from logs prior to writing the log file. The password string is removed in every instance it is encountered.

## Workspace, Configuration and Log File Paths

Tasktop Sync requires access to a directory to be able to store configuration information, save intermediate data for processing, and store cache data. This directory is determined automatically, but its location varies depending on what operating system you use and your initial installation procedure. Because of this variability, in the advanced sections of this manual we refer to this directory by the variable `$TASKTOPSYNC_HOME%`. The following is a list of common locations for this directory:

- On all supported versions of Windows: `$TASKTOPSYNC_HOME = c:\Users\<User>\AppData\Roaming\TasktopSync`

In each of these cases `<User>` refers to the user currently running Tasktop Sync.

The Tasktop Sync workspace directory is `$TASKTOPSYNC_HOME%\workspace`. This directory contains the Tasktop Sync working directory and other cache data used internally by Tasktop Sync.

The Tasktop Sync working directory is `$TASKTOPSYNC_HOME%\workspace\tasktop`. This directory contains all the Tasktop Sync configuration files, log files, and persisted data.

## Sync History Trimming Policy (Advanced)

Tasktop Sync keeps a record of all synchronizations that happened while it was running. Details about each synchronization are stored in a local database, and individual synchronization log files are stored in Sync's workspace. Over time, this history may grow to a significant size, negatively affecting Tasktop Sync's startup time. To minimize that effect you can apply one of the following trimming policies:

- `delete-up-to-last-success`: Deletes all sync logs for each task pair up to the last successful synchronization.
- `delete-all-but-last-x-syncs`: Deletes all but last X sync logs. By default X is 60. Can be changed by setting the param attribute.
- `delete-all-but-last-x-days`: Deletes all sync logs older than X days. By default X is 60. Can be changed by setting the param attribute.

By default, no trimming is performed.

Note that this applies to the files located in the `/tasktop/sync-log` folder, and not to logs in the `/tasktop/log` folder.

Below is an extract from a `synchronizer.xml` file which keeps sync history for the last week only.

```
<sync-model ...>
  <sync-history-trimming-policy policy="delete-all-but-last-x-days" param="7"/>
</sync-model>
```

A selected policy will be applied when the synchronizer starts and then twice a day while the synchronizer is running.

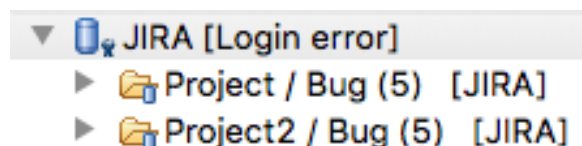
## Repositories in Error

If an error is encountered during synchronization while communicating with a repository, Tasktop Sync will mark the repository as being in error and will not attempt any further synchronization with the repository to avoid flooding the repository with invalid requests.

These errors should be resolved as soon as possible, as synchronizations will not proceed if one of the repositories on either side of the synchronization is in error.

## Repository Authentication Errors

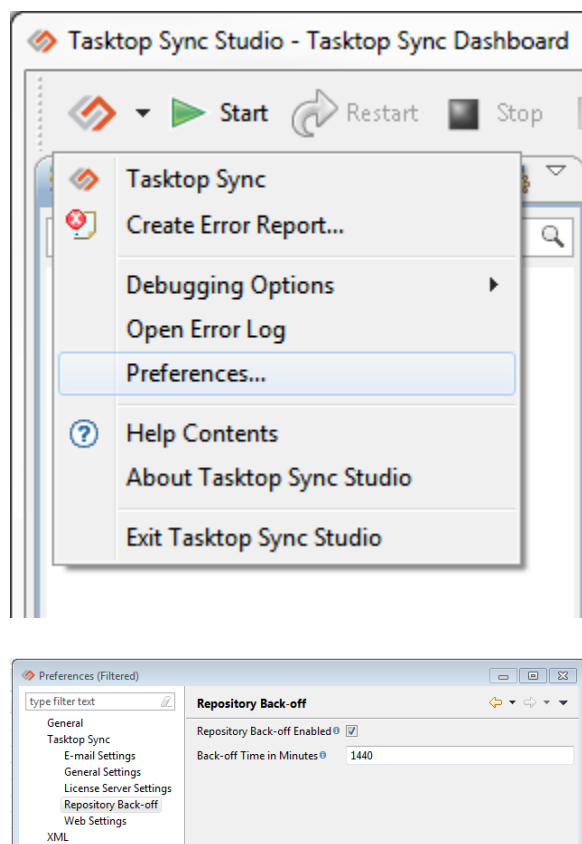
If an authentication error is detected with a repository, Sync will place the repository into "back-off" state, which is visible in the Sync Tasks view with a "[Login Error]" suffix:



When a repository is in back-off state, no communication will occur between Sync and the repository. This is to stop repositories from disabling the credentials used by Sync. The back-off will last for a period of time, and then communication will automatically resume. Back-off states are all cleared when the Synchronizer is stopped.

The length of time a repository is in back-off state is dependent on how many authentication errors have occurred recently. On the initial 3 authentication errors, a repository will enter back-off state for 5 minutes after each error. On the fourth authentication error, the repository will enter long-term back-off state, which lasts for 24 hours by default. If the authentication credentials are corrected, and communication resumes for 24 hours without any more authentication errors, then back-off counting is reset to 0.

Repository back-offs can be configured through the "Preferences" menu.



The following two configuration options are available:

1. **Repository Back-off Enabled:** If this checkbox is enabled, repositories can enter back-off state. If not enabled, the back-off feature is completely disabled, and repositories will not enter back-off state.
2. **Back-off Time in Minutes:** This sets the long-term back-off length, in minutes.

If you make any changes, and press the "Apply" or "OK" buttons, all repositories currently in back-off state are cleared, and communication will resume.

## Changing a Repository's URL (Advanced)

In the following section, we will refer to the actual server hosting the ALM service as the "server", and the object within Tasktop Sync as the "repository".

When the URL of a server is changed, Tasktop Sync needs to update its configuration and cache to use the new URL. Tasktop Sync provides a "Change Repository URL" wizard to assist you in this process. Please note that changing the URL of a repository used by the Tasktop Sync OSLC Linking service is not currently supported.

### Prerequisites

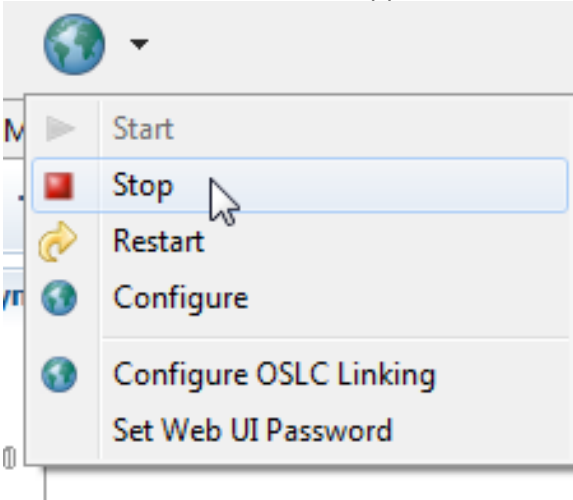
Before proceeding with the wizard, you should ensure that the workspace is backed up as described in [Backup and Restore](#).

The following criteria must be met before proceeding with changing the repository URL:

1. The synchronizer must be stopped. If this has not been done, please click



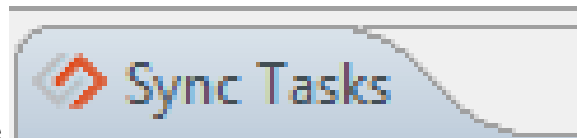
2. The web service must be stopped. If this has not been done, please click



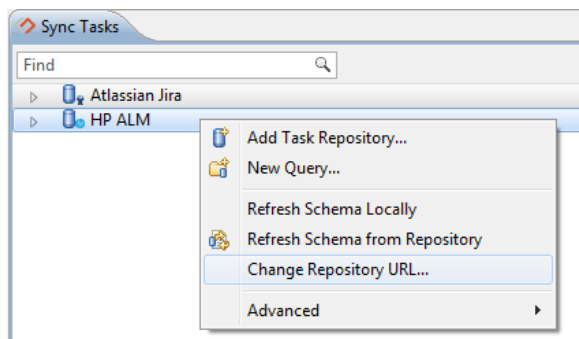
3. The configuration must be valid. If this is not so, please open the [Mapping Editor](#) (or alternatively open the XML) and make the necessary changes.
4. The repository must not be being used by the Tasktop Sync OSLC Linking service.

Additionally, you should ensure that all task mapping and XML editors are closed.

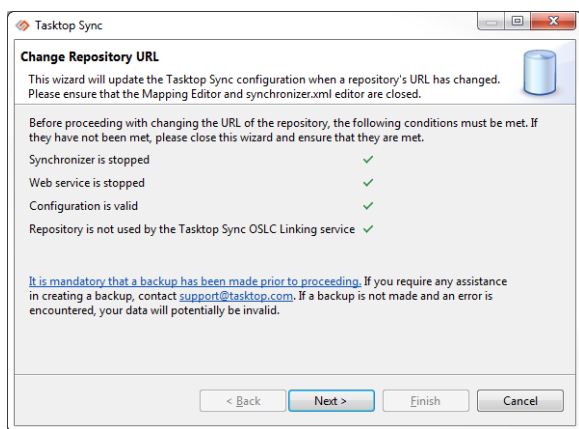
## Launching the Wizard



To launch the "Change Repository URL" wizard, open the view and right-click the repository whose server's URL has changed. Note that this option will only appear for repositories that support this feature.



A page will appear, checking for the aforementioned prerequisites. If any of these prerequisites are not met, you will not be able to proceed.



Next, a settings page for the repository will appear. Enter the new URL in the box labeled "Server:" and make any other necessary modifications to the repository settings. It is recommended to press "Validate Settings" if the server itself has already had its URL changed to check whether or not the settings are valid.



After entering the new URL, you will be prompted to confirm the new URL.

Clicking "Next" on this page will begin the migration process. This process may take several minutes.

Once migration is done, a results check list will appear indicating whether the migration succeeded. If a failure occurred, check the Error Log view and then [restore](#) the workspace from the backup created earlier. If the backup is not restored Sync will be in an invalid state after a failed repository URL change.

## Migrating a Project to Another Server (Advanced)

In the following section, we will refer to the actual server hosting the ALM service as the "server", and the object within Tasktop Sync as the "repository".

When a project is migrated to another server, Tasktop Sync needs to update its configuration and cache to use the new URL. Tasktop Sync provides a "Migrate Repository to Another Server" wizard to assist you in this process. Please note that automatically updating the Tasktop Sync OSLC Linking service is not currently supported.

## Prerequisites

This process is not supported by all ALM systems in Tasktop Sync. Currently the following systems are supported.

- HPE ALM

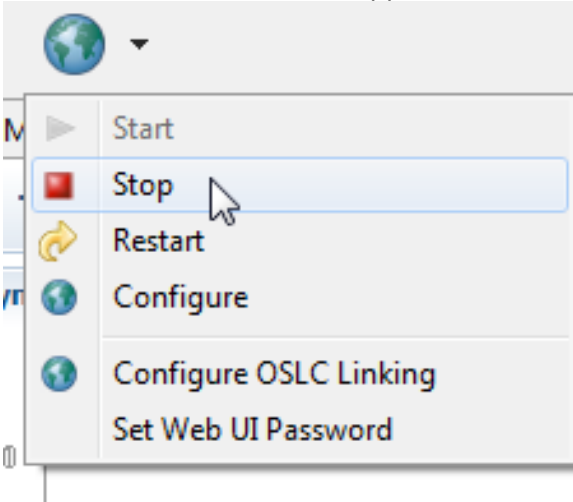
Before proceeding with the wizard, you should ensure that the workspace is backed up as described in [Backup and Restore](#).

The following criteria must be met before proceeding with changing the repository URL:

1. The synchronizer must be stopped. If this has not been done, please click



2. The web service must be stopped. If this has not been done, please click

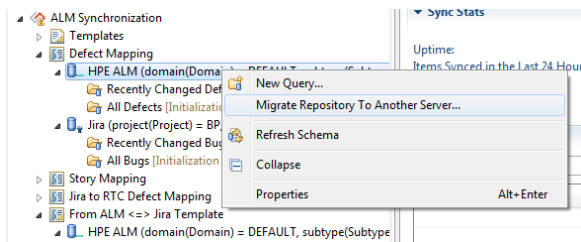


3. The configuration must be valid. If this is not so, please open the [Mapping Editor](#) (or alternatively open the XML) and make the necessary changes.
4. The repository must not be being used by the Tasktop Sync OSLC Linking service.
5. The mapping involving the repository must be a full scope mapping.

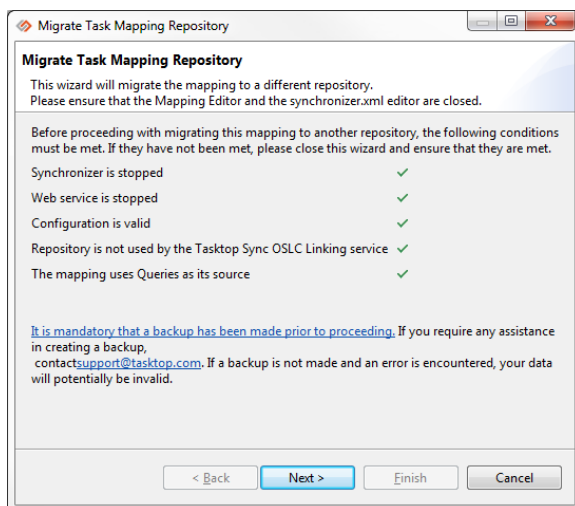
Additionally, you should ensure that all task mapping and XML editors are closed.

## Launching the Wizard

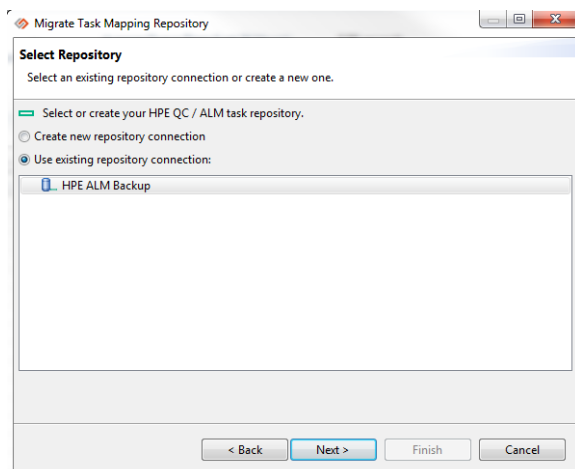
To launch the "Migrate Repository to Another Server" wizard, find the task mapping in the Services view which is synchronizing the migrated project and right click on the appropriate repository. Then select "Migrate Repository to Another Server...".



A page will appear, checking for the aforementioned prerequisites. If any of these prerequisites are not met, you will not be able to proceed. If all prerequisites are met, click "Next".



Next, the new server containing the project must be chosen. If a repository for this server has already been created it can be selected on the "Select Repository" page by selecting "Use existing repository connection". If a repository has not yet been created for this server one may be created by selecting "Create new repository connection".



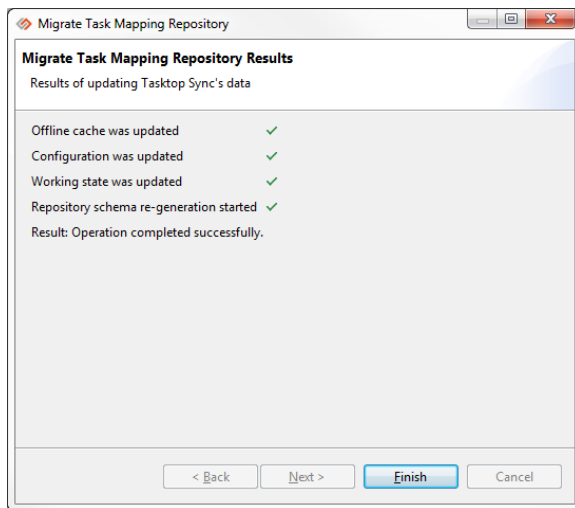
After choosing the repository to use in the mapping, new initialization and changes queries must be configured. When queries are configured Tasktop Sync will perform a schema generation and ensure that the configured queries contain tasks which match the mapping's scope. Note that the scope cannot be changed; to perform a migration the scope before and after the migration must be identical.

The screenshot shows a dialog box titled "Migrate Task Mapping Repository" with a subtitle "Configure Mapping Source and Scope for HP ALM 12.01". It contains two rows of configuration for "HP ALM 12.01":  
1. "Initialization Query" set to "HP ALM Defects Migrated" with "2 Tasks" and a "New..." button.  
2. "Changes Query" set to "HP ALM Defects Migrated" with "2 Tasks" and a "New..." button.  
Below these are dropdown menus for "Domain" (TASKTOP), "Domain Project" (Migration\_Project), "Type" (Defect), and "Subtype" (None). A "Refresh" button is at the bottom left. Navigation buttons at the bottom are "< Back", "Next >", "Finish", and "Cancel".

Once the schema has been generated and the configured queries have been validated, clicking "Next" will present a page detailing the migration process. Clicking "Next" on this page will begin the migration process. This process may take several minutes.

The screenshot shows a dialog box titled "Migrate Task Mapping Repository" with a subtitle "Mapping Repository Migration". It contains the text: "Confirm Mapping Repository Migration. The mapping will be updated to refer to the new repository. During this process Tasktop Sync's configuration will be altered such that future synchronizations will apply to the newly selected repository. This presumes that the tasks which were previously synchronized by this mapping have been moved to the new repository." Below this are two text fields: "The old repository URL is: http://dev-qc049.van.tasktop.com:8080/qcbin" and "The new repository URL will be: http://dev-qc239.van.tasktop.com:8080/qcbin". A note states: "This process could take several minutes. Press 'Next' to proceed." Navigation buttons at the bottom are "< Back", "Next >", "Finish", and "Cancel".

Once migration is complete, a results check list will appear indicating whether the migration succeeded. If a failure occurred, check the Error Log view and then [restore](#) the workspace from the backup created earlier. If the backup is not restored Sync will be in an invalid state after a failed migration.



## After Completing the Wizard

If a project which has been migrated is involved in more than one task mapping, for example if there are separate task mappings for synchronizing multiple item types within a project, then the wizard must be run separately for each of the relevant mappings.

# Manual Task Synchronization Configuration (Advanced)

This section documents the steps required to configure Tasktop Sync manually. Note that this section assumes that you are familiar with the Tasktop Sync UI, terminology, and concepts described in the previous sections. If you are configuring Tasktop Sync for the first time, or creating a task mapping which does not require complicated logic, then we recommend you configure your mappings as described in the [Task Synchronization Configuration](#) section.

Manually configuring Tasktop Sync follows these steps:

1. **Create task repositories.** In this step you create connections to each of the repositories you want Tasktop Sync to be aware of.
2. **Create changes and initialization queries for each repository.** In this step you create the queries which define which tasks in your repository are visible to Tasktop Sync for synchronization.
3. **Setup the synchronizer configuration file.** In this step you specify several runtime characteristics of Tasktop Sync and explicitly specify which repositories are synced and how they are synced. There are several subparts to this step:
  - a. **Create person mappings between repositories (optional).** In this step you configure the user identification mappings between repositories. This step is required only when your repositories use different user IDs for the same user.

- b. **Define task mappings between repositories.** In this step you specify how a task in one repository is synced to its proxy task in another repository. This step also has several subparts.
    - Define one or more **attribute mappings** for each task mapping.
4. **Configure logging** (optional, maintenance section).
5. **Define sync history trimming policy** (optional, maintenance section).

Before you begin configuration, we recommend that you first skim this section in its entirety to get an idea of how Tasktop Sync is configured. Then read the sections of **Certified Connectors** that apply to your setup, since parts of the configuration will differ depending on which vendors you use.

## Repository Connections

The first step in configuring Tasktop Sync is to set up connections to the repositories that Tasktop Sync will connect to. This section describes how to manually create repository connections for use. Note that repository connections will have already been created if you created a task mapping using the Quick Start Wizard and task mapping editor tools provided in Tasktop Sync.

Right-click in the Sync Tasks view and select "Add Task Repository...", or click the "Add Task Repository..." button in the top right corner of the view, to initiate repository creation. Follow the on-screen instructions to create the repository. Note the value you use in the `Server:` field of the repository as you will need to use this value to specify which repositories should be synchronized.

Once you have filled out that form, we strongly recommend that you validate your credentials before finishing; press the `Validate Settings` button to make sure that you can connect to the repository.

## Custom Authentication

Repositories can be configured to use custom authentication e.g. single sign-on systems. To configure such an authentication you can use the "Additional Settings" section within the repository dialog.

The first thing to do is to select the Authentication of your choice e.g. "CA Single Sign-On (Login Form)". "Standard Authentication" is the default and requires no additional configuration.

After selecting the authentication you need to provide input for the required fields. In the case of "CA Single Sign-On (Login Form)" this is a URL.

**Properties for Task Repository**

Please specify parameters for connection.

Server:

Label:  ☐ Disconnected

User ID:

Password:  ☒ Save Password

**Additional Settings**

Authentication:

Location of the Login Form:

The User ID and Password fields from the top of this dialog will be used to authenticate.

☒ Validate on Finish

Some authentications need to fetch additional settings dynamically. For this reason the "Connect..." button needs to be pressed after entering values for the required fields. The next step is to fill out the additional settings that appear after connecting by following the on screen instructions.

Once you have filled out that form, we strongly recommend that you validate your credentials before finishing; press the `Validate Settings` button to make sure that you can connect to the repository.

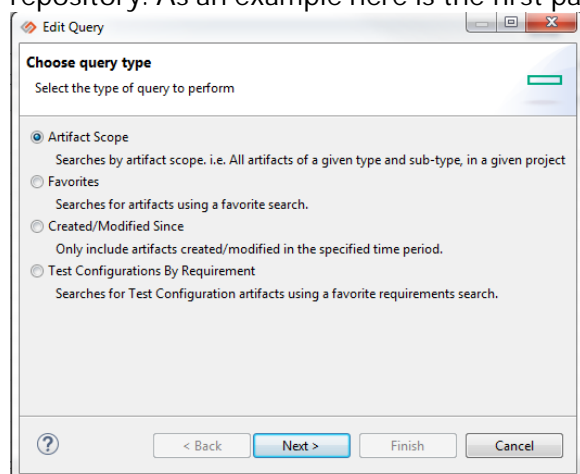
## Create Queries

Note that queries may have already been created if you created a task mapping using the Quick Start Wizard and task mapping editor tools provided in Tasktop Sync.

You need to specify two queries for each configured repository: the initialization query and the changes query. These queries define the tasks which come to Tasktop Sync as input. For example, if you only want defects to synchronize but don't care about test cases, then you would set up your queries to return only defects.

To create queries, go to the Sync Tasks view and right-click on the repository that you would like to create a query for and select "New Query...", or click the "New Query..." button in the top right corner of the view.

At this point, you will get a wizard which walks you through creating a query for the specified repository. As an example here is the first page of the HPE QC / ALM new query wizard:

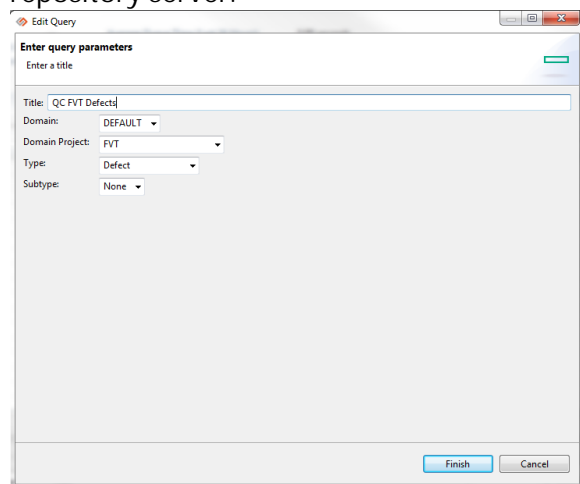


The screenshot shows the 'Edit Query' window with the title 'Choose query type'. Below the title is the instruction 'Select the type of query to perform'. There are four radio buttons with corresponding descriptions:

- ☒ **Artifact Scope**  
Searches by artifact scope. i.e. All artifacts of a given type and sub-type, in a given project
- ☐ **Favorites**  
Searches for artifacts using a favorite search.
- ☐ **Created/Modified Since**  
Only include artifacts created/modified in the specified time period.
- ☐ **Test Configurations By Requirement**  
Searches for Test Configuration artifacts using a favorite requirements search.

At the bottom, there is a help icon (?), a '< Back' button, a 'Next >' button, a 'Finish' button, and a 'Cancel' button.

It is possible to use a query that you have saved on the server. We advise doing so when possible. In such cases, the first page of the query-building wizard will say something like "Create query using a favorite filter". Select that radio button. You will then be able to choose from named queries from the repository server:



The screenshot shows the 'Edit Query' window with the title 'Enter query parameters'. Below the title is the instruction 'Enter a title'. There is a text field for the title containing 'QC FVT Defect[]'. Below this are four dropdown menus:

- Domain: DEFAULT
- Domain Project: FVT
- Type: Defect
- Subtype: None

At the bottom right, there are 'Finish' and 'Cancel' buttons.

Otherwise, select the radio button that says something like "Create query using a form" from the first page of the wizard and fill out the form to create a query for that repository.

Note: Depending on your task repository's capabilities, there may be a restriction on the tasks you wish to sync which cannot be specified with a query. This is a normal situation, and Tasktop Sync provides finer-grained tools for selecting which tasks should be synced (discussed in the [attribute mapping](#) section). However, you should create queries which capture as few extraneous tasks as possible, in order to reduce communication bandwidth usage between Tasktop Sync and the repository server.

## Initialization Query

The initialization query is designed to be used to determine all of the tasks you are interested in synchronizing in a repository so that you can easily synchronize all existing tasks of interest in a new



installation of Tasktop Sync, or can reinitialize synchronization between two repositories if you change your synchronization mappings. As such, the initialization query is only run manually, never automatically.

Under ideal conditions, you will run the initialization query once, when you are doing your initial synchronization, and if necessary again if you change your synchronization configuration. For your initialization query, we advise using a query that is guaranteed to get every last task that might possibly be of interest, even if the query is computationally expensive.

## Changes Query

While the initialization query is run manually and only a few times, the changes query is run frequently to determine which tasks have changed recently and require synchronization. The changes query is run automatically at a configurable time interval or a cron schedule in the [synchronizer.xml](#).

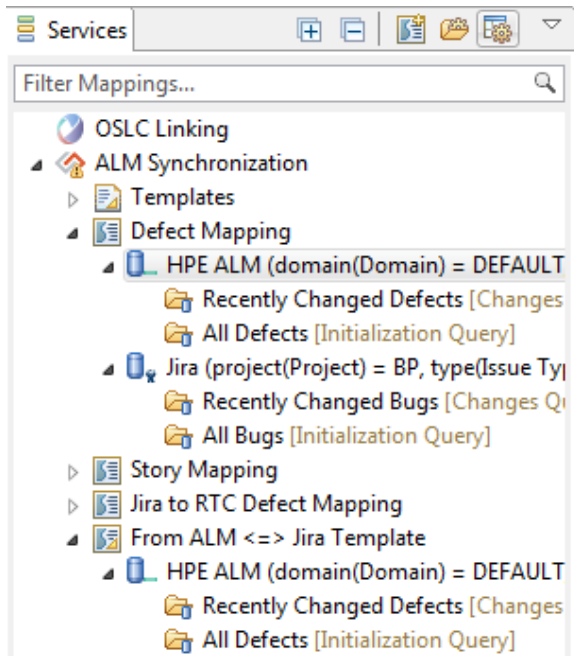
Because it is run so frequently, it should be designed more carefully than the initialization query to run quickly and to return as small a number of tasks as possible while still returning all the tasks of interest. While you should only get recently-changed tasks, "recent" is a relative term. The farther back in time your query extends, the more tasks will be found. This means that there is less danger of omitting relevant tasks (for example, because a server went down for longer than the last query execution) but that it will take longer to run. We recommend that your changes query get tasks which have changed in the past day.

## synchronizer.xml

The file `synchronizer.xml` is the main configuration file, specifying the overall runtime characteristics of the synchronizer as well as all task mappings and user mappings. It is stored in the Tasktop Sync working directory.

The primary way to configure the `synchronizer.xml` file is via the Task Mapping Editor. Before proceeding with this section, please read the [Task Synchronization Configuration](#) section, as it discusses the specifics of the form and function of the `synchronizer.xml` file itself.

You can access the `synchronizer.xml` configuration file for editing by clicking on the gear icon in the right of the title bar of the Services view.



Note that we recommended that you do not edit the `synchronizer.xml` file, or files referenced by `synchronizer.xml`, in external editors. Tasktop Sync assumes the UTF-8 character encoding of the configuration files. Editing these files outside of a UTF-8 environment may cause Tasktop Sync to work incorrectly. Extra care must also be taken when pasting content into the configuration files.

The `synchronizer.xml` file has four main sections, where you specify

- the run characteristics,
- person mapping definitions,
- task mappings, and
- where to send statistics and/or error messages.

The task mappings section can be further broken down into two sections:

- the repository specification, and
- the task attribute mapping specification.

Below is an example of a `synchronizer.xml` file which synchronizes three task attributes (summary, description, and owner) between an HPE QC / ALM repository and an IBM Rational Team Concert repository. The individual elements of the file are discussed afterwards.

```
<sync-model
  default-hide-query-progress="false"
  worker-threads="5"
  default-query-interval-minutes="0"
  pause-on-start="false"
  xmlns="http://tasktop.com/xml/ns/sync/model"
  version="4.0.0">

  <person-mapping name="PersonMap" mapping-path="person-map.xml"/>

  <task-mapping id="1"
    display-name="Defects"
    conflict-notification-policy="Log"
    person-mapping="PersonMap">
```

```

<!-- HPE QC / ALM Repository -->
<repository proxy-creation-trigger="all"
    auto-comment-user="developer1"
    id="com.tasktop.sync.QC"
    url="http://qc.example.com:8080/qcbin;SYNCHRONIZER;ProjectArea">
    <scope>
        <field id="itemType" value="BUG"/>
    </scope>
    <source>
        <query name="initialization" value="Tasktop Sync - HPE QC / ALM - All Defects"/>
        <query name="changes" value="Tasktop Sync - HPE QC / ALM - Recently Modified Defects"/>
    </source>
    <proxy-storage store="attribute" attribute-id="BG_USER_01"/>
</repository>

<!-- IBM Rational Team Concert Repository -->
<repository proxy-creation-trigger="all"
    auto-comment-user="tester1"
    id="com.tasktop.sync.RTC"
    url="https://rtc.example.com:9443/ccm">
    <scope>
        <field id="itemType" value="task"/>
        <field id="project" value="RTCProjectName"/>
    </scope>
    <source>
        <query name="initialization" value="Tasktop Sync - IBM Rational Team Concert - All Defects"/>
        <query name="changes" value="Tasktop Sync - IBM Rational Team Concert - Recently Modified Defects"
/>
    </source>
    <proxy-storage store="task-relations"/>
</repository>

<!-- Specification of comment mappings between repositories ..?
<comment-mapping>
    <repository sync-incoming-comments="true" comment-content-type="text/html" />
    <repository sync-incoming-comments="true" comment-content-type="text/plain;markup=RTCRichText" />
</comment-mapping>

<!-- Specification of task attribute mappings between repositories -->
<attribute-mapping key="BG_DESCRIPTION"/>
<attribute-mapping key="BG_SUMMARY"/>
<attribute-mapping>
    <attribute-key key="BG_RESPONSIBLE" caster="person"/>
    <attribute-key key="BG_RESPONSIBLE" caster="person"/>
</attribute-mapping>
</task-mapping>

<job name="StatisticsEmail" type="StatisticsEmail" schedule="0 0 0 * * ?"/>
<job name="MetricsEmail" type="MetricsEmail" schedule="0 0 0 1 * ?"/>

<statistics-notification>
    <email address="statsUser1@example.com"/>
    <email address="statsUser2@example.com"/>
</statistics-notification>

<error-notification include-synchronization-errors="true" include-synchronization-conflicts="true" include-
query-errors="true">
    <email address="errorUser@example.com"/>
</error-notification>
</sync-model>

```

## Run Characteristics

The run characteristics are specified in the root node of `synchronizer.xml`: the `sync-model` element. It contains several attributes which you may configure as desired:

Attribute	Value Type	Description	Default
-----------	------------	-------------	---------

pause-on-start	true or false	Specifies whether Tasktop Sync should be paused when it starts.	false
default-query-interval-minutes	non-negative decimal	Specifies how many minutes to wait between changes queries. If you wish to synchronize continuously, set this attribute to "0". For longer delays, set it higher. Either this, or the default-query-schedule should be specified.	1.0
default-query-schedule	a cron expression	Specifies when to run the changes queries. Either this, or the default-query-interval-minutes should be specified.	0/30 * * * ?
worker-threads	positive integer	Specifies how many threads to allocate to processing synchronizations.	5
default-hide-query-progress	true or false	Specifies whether or not the Progress view is used when the queries are run.	false
update-repository-configurations-on-start	true or false	Specifies whether or not to update the repository configuration of every repository referenced by a task mapping when Tasktop Sync starts up.	true
repair-proxy-associations	true or false	If true, Tasktop Sync will make a best effort to automatically repair missing proxy associations. Tasktop Sync will only be able to repair proxy associations on tasks if the synchronization information for those tasks exists in Tasktop Sync's cache. Note that this functionality is required for normal functioning of Tasktop Sync.	true
attachment-limit	integer	Specifies the maximum number of attachments Tasktop Sync will post to a single task. When a task with more than this number of attachments is encountered then synchronization of attachments will be aborted and a warning will be written to the log. Synchronization of other attributes will continue normally. A negative number indicates an unlimited number of attachments.	100
maximum-submissions-per-sync	integer	Specifies the maximum number of times Tasktop Sync will submit changes per synchronization when using <a href="#">scripted attribute handlers</a> that are configured to submit multiple times. If a task synchronization submits changes more often than this number, the synchronization will enter the error state.	15

## Person Mapping Definitions (Optional)

It is common for a person to have different identities on each repository they use. For example, a person may be identified by a username in one repository and by an email address in another. Person mappings can be used to map between a person's identities in different repositories when determining, for example, the owner or reporter. (Note that you will need to define a person caster for any attributes which should use the person mapping. See [Using Person Mappings](#) below.)

The `person-mapping` element declares a person mapping for use in the rest of the configuration file. It specifies the path to the file containing the definition of the person mapping. In our above example we have the following definition:

```
<person-mapping name="PersonMap" mapping-path="person-map.xml" />
```

This creates a person mapping named "PersonMap" which is specified in the file "person-map.xml" in the working directory. Once a person mapping has been defined in this way, it is available for use in all following `task-mapping` clauses. See [Person Mapping Files](#) for their structure and syntax. You will need to write a person mapping file for each person mapping you define.

When you declare a person mapping in the `synchronizer.xml` file, the Tasktop Sync Services view will display a node for this mapping. Double-clicking the node will create a sample person mapping file and open it for editing.

The `default-person-id` in the `repository` element specifies the default identity which will be used for that repository, when Tasktop Sync encounters a person identity not included in the mapping.

## Task Mapping Specification

Each task mapping specifies a mapping between two repository items which the synchronizer should keep in sync. Task mappings are specified in the `task-mapping` element.

There are three parts to a `task-mapping`. The first is the specification of behaviour that applies to the entire task mapping, the second defines the repositories together with properties that should participate in task synchronization, and the third part defines the mapping between individual task attributes in each of the repositories.

There are four options that are set as attributes on the `task-mapping` element that govern some global behaviour of the mapping:

Attribute	Value Type	Description
id	A unique value	Specifies a unique system ID for this mapping. Once tasks have been synced using this mapping, this value should not be changed.
display-name	Any text	Specifies a descriptive name for the mapping.

conflict-resolution-policy	no-sync, repository-1-dominates, or repository-2-dominates	Specifies which <a href="#">conflict resolution policy</a> to use. "1" and "2" refer to the order the repositories are specified in the <code>repository</code> element, below. The default is <code>no-sync</code> .
conflict-notification-policy	Comment or Log	Specifies how conflicts are reported by Tasktop Sync. When <code>Comment</code> is specified, a comment is appended to the task to indicate the conflict in addition to logging it in the log files. <code>Log</code> is the default, and specifies that the conflict should only be noted in the log files.
person-mapping	string	Name of a person-mapping element, defined above. Identifies the person mapping that should be used.

Tasktop Sync makes sure that an incoming task matches at most one task mapping. If the task matches more than one mapping, synchronization does not proceed and is moved to the error queue.

## Repository Specification

Repositories in each task mapping are defined in the `repository` element. There must be exactly two such elements in each `task-mapping` to identify participating repositories. Each repository specification has a set of required configuration settings. The configuration specified as attributes of the `repository` element are:

Attribute	Value Type	Description
id	String	A unique identifier for this particular repository configuration.
proxy-creation-trigger	all or none	Specifies when a new incoming task should have a synchronized proxy task created.
write-policy	allow-writes or proxy-association-only	Specifies a policy for writing data to the repository. Default is <code>allow-writes</code> . See below for more information.
auto-comment-user	String	The user that should be used by Tasktop Sync when it creates incoming notification comments for tasks on the repository (i.e. for reporting a conflict).

url	URL	The URL used to connect to this repository (e.g. an IBM Rational Team Concert URL might be  <a href="https://rtc.example.com:9443/ccm">https://rtc.example.com:9443/ccm</a>  )
query-interval-minutes	non-negative decimal	Specifies how many minutes to wait between consulting the changes query for this repository only. This value overrides default-query-interval-minutes or default-query-schedule in the sync-model element. You cannot specify this and a query-schedule.
query-schedule	cron expression	Specifies when the changes query will run, for this repository only. This value overrides default-query-interval-minutes or default-query-schedule in the sync-model element. You cannot specify this and a query-interval-minutes.
author-validation-policy	no-validation, default-user-on-failure	Specifies whether or not to validate the authors of comments and attachments, and what to do if validation fails. When default-user-on-failure is used the auto-comment-user will be set as the author of a comment or attachment if the mapped author is found to not exist in the target repository. When no-validation is used no checks will be made to ensure the author exists in the target repository.

The `write-policy` specifies whether or not Tasktop Sync is expected to write data to the repository and can be used to enforce a one-way sync of data. The default policy `allow-writes` performs no checking of write operations, whereas the policy `proxy-association-only` ensures that all other settings related to write operations are set to eliminate writes, except for setting the proxy association on each task. Specifically, it checks the following:

- All attributes mapped to this repository have their strategy set to `ignore`.
- Comment syncing is turned off by setting `sync-incoming-comments` to `false`.
- Attachment syncing is turned off by setting `sync-incoming-attachments` in the `attachment-mapping` to `false` if an `attachment-mapping` exists.
- Time Worked syncing is turned off by setting `sync-incoming-entries` in `time-worked` to `false` if a `time-worked` element exists.
- The `proxy-creation-trigger` is set to `none`.

For `proxy-creation-trigger`, it is possible to customize when proxy tasks are created. If `proxy-creation-trigger` is "all" (the default), then all new tasks from a repository will have proxies created for them in the corresponding target repository. If `proxy-creation-trigger` is "none" then proxy tasks are never created. This can be useful in a scenario where you want to enforce task creation in one repository and disallow it in the other.

**Important!** The repository specification is position-dependent. The order that the repositories are specified in the file determines which is `repository-1` and which is `repository-2` in `repository`

-1-dominates and repository-2-dominates; this order is also the order in which they are referenced in attribute mappings, below.

## Scope Specification

Each `repository` can limit which tasks it can access through a `scope` element. The `scope` element contains a list of `field` elements with IDs and values. Each vendor's repository defines a unique list of fields for their scope. It is recommended that you use [Quick Start](#) to create an initial mapping, which provides help with defining the scope fields and their values.

It is possible that the specification of a system's scope may change when Tasktop Sync is updated. When this occurs Tasktop Sync's configuration must be updated to align with the new scope definition. Until the configuration is updated Tasktop Sync will not run. Details on how to handle this scenario are found on the Tasktop Sync [FAQ](#).

## Source Specification

The `repository` element is connected to the actual repository through the `source` element. The `source` element can either have a set of two `query` elements (see [Create Queries](#)), or a `task` element (see [Limited Scope](#)). Each is described below.

Element	Value Type	Restrictions	Description
<code>&lt;query name="changes" value="" /&gt;</code>	String	Must be paired with <code>initialization</code>	The name of the query to be used for the <a href="#">changes query</a> .
<code>&lt;query name="initialization" value="" /&gt;</code>	String	Must be paired with <code>changes</code>	The name of the query to be used for the manual <a href="#">initialization query</a> .
<code>&lt;task url="" /&gt;</code>	URL	Must be the only <code>source</code> element	The URL identifying the single task to map to, described in <a href="#">Limited Scope</a> .

## Proxy Association Storage

Tasktop Sync needs to keep track of the association between a task and its proxy task. The association is persisted in a *proxy store*. There are different types of proxy stores available and each `repository` element must specify the store type through a `proxy-storage` element.

### Attribute Proxy Store

The *attribute* store sets an attribute on each task to the URL of its proxy. The attribute can be specified through the `attribute-id` or `attribute-label` property on the `proxy-storage` element. In the example below, we specifying that the attribute `BG_USER_04` should be used to store the proxy association URL.



```
<repository ...>
...
<proxy-storage store="attribute" attribute-id="BG_USER_04"/>
</repository>
```

Some connectors store some attributes as *sparse* (see [Sparse Attributes](#)). If this is the case for a configured proxy attribute, the repository must declare the proxy attribute as such. This is done by adding the `sparse="true"` property to the `proxy-storage` element on a repository. In the example below, we specify that the `sparse` attribute `task.optional.attribute` should be used to store the proxy association URL. Refer to the connector-specific documentation to determine if the `sparse` property is required.

```
<repository ...>
...
<proxy-storage store="attribute" attribute-id="task.optional.attribute" sparse="true"/>
</repository>
```

## Database Proxy Store

The *database* store stores proxy association information in Tasktop Sync's internal database.

```
<repository ...>
...
<proxy-storage store="database"/>
</repository>
```

**Important!** When a repository is configured to use the database store, Tasktop Sync's internal database becomes the only source of proxy association information and needs to be backed up regularly. Failure to do so could cause duplicate proxy tasks being created.

## Task relations Proxy Store

Some connectors have a built-in way of storing links to other tasks, and for those connectors it's possible to use the *task relations* store.

```
<repository ...>
...
<proxy-storage store="task-relations"/>
</repository>
```

However, it's recommended to use the other stores because they allow Tasktop Sync to store additional information together with the proxy association that cannot be stored using the built-in linking mechanism. This additional information allows Tasktop Sync to detect when a task has been copied and avoids the possibility of two tasks having links to the same proxy task. Such a situation will result in a synchronization error.

## Changing Configured Proxy Store

Note that if two repositories contain a number of tasks that have already been synced, then changing configured proxy store will have the following consequences:

- **Switching from a `task-relations` to an `attribute` proxy store:** If Tasktop Sync inspects the storage attribute and finds it empty, it will fall back to the default method of storing associations.

In this case, existing tasks that have been synced using the default method will still maintain their proxy association. Subsequent updates to all tasks will result in the proxy associations being stored on the new attribute.

- **Switching from an attribute to a task-relations proxy store:** Existing tasks that have been synced will lose their proxy association and will appear to have not been synced. This will result in new proxy tasks being created and linked using the built-in method.
- **Changing the attribute-id property of an attribute store:** Similar to switching from the attribute to task-relations proxy store, tasks will lose their links and Tasktop Sync will create new proxy tasks and link them using the new attribute.
- **Switching from a database to another proxy store:** If Tasktop Sync inspects the newly configured proxy store and finds it empty, it will recover the proxy association from the database providing the `repair-proxy-associations` property on the `sync-model` element is set to `true`. Subsequent updates to all tasks will result in the proxy associations being stored on the newly configured stores.
- **Switching to a database proxy store from another store:** Extra care is necessary when performing this configuration change. Prior to the configuration change, Tasktop Sync admin has to make sure that the internal Sync cache stores a backup copies of proxy associations. This can be ensured by re-synchronizing all the tasks in scope of a task mapping.

## Property Specification

Additionally, each `repository` can contain a number of `property` elements. Only the generally applicable properties are described here. Each vendor-specific repository will likely have an additional set of required and optional properties. These properties are described in sections of [Certified Connectors](#).

## Attachment Mapping Specification

Every `task-mapping` element may have an optional `attachment-mapping` element which specifies the attachment synchronization behaviour. Every `attachment-mapping` element must contain two `repository` elements each with a required boolean attribute `sync-incoming-attachments` which when set to `true` will cause attachments to be synced into the corresponding repository.

```
<attachment-mapping>
  <repository sync-incoming-attachments="true"/>
  <repository sync-incoming-attachments="false"/>
</attachment-mapping>
```

## Attachment file size filter configuration

Optionally each `repository` element may define an `attachment-filesize-filter` element which will enable filtering of oversized attachments. The `attachment-filesize-filter` elements has two required attributes `filter-strategy` and `size-limit`. `filter-strategy` may be one of either `error` or `ignore`, `error` will cause any synchronization with an oversized attachment to enter the error queue stopping all synchronization of those tasks, `ignore` will skip synchronizing the

oversized attachment allowing the rest of the synchronization to proceed. `size-limit` is the maximum attachment size in bytes, any attachment which is larger than this limit will trigger the attachment filter.

```
<attachment-mapping>
  <repository sync-incoming-attachments="true">
    <attachment-filesize-filter filter-strategy="error" size-limit="362659"/>
  </repository>
</attachment-mapping>
```

## Failed attachments filter configuration

Optionally each `repository` element may define a `failed-attachment-filter` element which will handle invalid attachments. The `failed-attachment-filter` elements has one required attribute: `filter-strategy`. `filter-strategy` may be one of either `error` or `ignore`, `error` will cause any synchronization with an invalid attachment to enter the error queue stopping all synchronization of those tasks, `ignore` will skip synchronizing the invalid attachment allowing the rest of the synchronization to proceed.

```
<attachment-mapping>
  <repository sync-incoming-attachments="true">
    <failed-attachment-filter filter-strategy="error"/>
  </repository>
</attachment-mapping>
```

## Comment Mapping

Every `task-mapping` element may have an optional `comment-mapping` element which specifies the comment synchronization behaviour. Every `comment-mapping` element must contain two `repository` elements. Each `repository` element may have the following attributes:

Attribute	Value Type	Description
sync-incoming-comments	true or false	Specifies whether comments should be synced to this repository. Default is false.
comment-content-type	Mime type string	The content-type of the comments attributes in the repository (e.g. "text/html").
add-author-header-to-comments	true or false	Specifies whether a line specifying the author of the comment in the source repository will be added to the beginning of all comments synced to this repository.

```
<comment-mapping>
  <repository sync-incoming-comments="true" comment-content-type="text/html" add-author--header-to-comments="true" />
```

```

    <repository sync-incoming-comments="true" comment-content-type="text/html" add-author--header-to-
comments"true" />
  </comment-mapping>

```

## Comment Visibility Configuration

In addition, each `repository` element may define a `comment-visibility-filter` element, and a `comment-visibility-assignment` element. The `comment-visibility-filter` defines what comments may be synchronized out of the repository, while the `comment-visibility-assignment` defines the visibility of comments being synchronized into the repository. The comment visibility filter, and comment visibility assignment, are only applicable for repositories which support the notion of public and private comments. The `comment-visibility-filter` element has a single required attribute `comment-visibility`, which may be set to `all`, `private`, or `public`. When it is set to `all`, no filtering is done and both public and private comments will be synchronized out of and into this repository. When it is set to `private`, only private comments will be synchronized out of this repository. When it is set to `public`, only public comments will be synchronized out of this repository. The `comment-visibility-assignment` element has a single required attribute `comment-visibility`, which may be set to `retain`, `private`, or `public`. When it is set to `retain`, all comments synchronized into this repository will be created with the same visibility as in their originating system. When it is set to `private`, all comments created in this repository will be created with a visibility of `private`. When it is set to `public`, all comments created in this repository will be created with a visibility of `public`.

```

<comment-mapping>
  <repository sync-incoming-comments="true" comment-content-type="text/html" add-author--header-to-
comments"true" >
    <comment-visibility-filter comment-visibility="private" />
    <comment-visibility-filter comment-visibility="public" />
  </repository>
  <repository sync-incoming-comments="true" comment-content-type="text/html" add-author--header-to-
comments"true" />
</comment-mapping>

```

## Oversized Comment Handling

Each `repository` may define an `oversized-comment-handling` element. This element determines how Tasktop Sync will handle comments which are too large to be synchronized into the target system. The element has a single attribute `filter-strategy` which is required and may have one of two values, `error` and `ignore`. Configuring the element with `error` will cause Tasktop Sync to throw an error during comment synchronization when it detects that a comment is too large to be submitted to the target system. Configuring the element with `ignore` will cause Tasktop Sync to not synchronize any comments which are too large to be submitted to the target system. If no `oversized-comment-handling` is specified Tasktop Sync will not check the comment size and comments may be submitted to the target system which are too large which may result in errors or other unexpected behaviour. The size threshold which triggers the `oversized-comment-handling` is determined by the connector, and is stored in the Tasktop Sync's repository schema.

## Attribute mapping

This section describes how to configure attribute mapping from the `synchronizer.xml` within Tasktop Sync. The more user-friendly way to edit these attribute mappings is through the [Attribute Mappings Editor](#). Note that some repositories are flexible enough to support task attribute mappings natively; for these repositories, you can use any combination of Tasktop Sync attribute mapping and repository-native attribute mappings. Please see the vendor-specific configuration section for your repositories to see if they support native mappings.

For each pair of task attributes to synchronize, one `attribute-mapping` element is specified as a child of the `task-mapping` element. All of the information required to synchronize values of this attribute from one repository to another and vice versa are included in this element definition. Note that the order in which you specify options in an `attribute-mapping` corresponds to the order of the `repository` clauses of the task mapping.

This section is heavily example-based; most users should be able to modify examples in this section slightly to suit their needs.

## Mapping Task Attributes with the Same Name and Type

In some (fortunately) simple cases, your task repositories may have attributes with the same name and type that can be mapped directly. To specify such a mapping, use the simplest attribute mapping:

```
<attribute-mapping key="Description"/> <!-- maps the "Description" attribute between your repositories -->
```

## Mapping Task Attributes with different names

In many cases, the attributes you want to synchronize have the same type but different names. In this case you can use the `attribute-mapping` element as follows:

```
<attribute-mapping>
  <attribute key="BG_SUMMARY"/> <!-- The name of the attribute in your first repository (in our case, HPE
QC / ALM) -->
  <attribute key="Description"/> <!-- The name of the attribute in your second repository (in our case,
IBM Rational Team Concert) -->
</attribute-mapping>
```

Note that this expanded way of specifying the task attribute mapping is position-dependent: the first `attribute` line corresponds to task attributes in the first repository and the second line refers to task attributes in the second repository. Thus, in this example, the "BG\_SUMMARY" task attribute in the qc.example.com repository is synchronized with the "Description" task attribute in rtc.example.com.

## Mapping Task Attributes by ID

There are many task attributes which have the same meaning in different vendors' repositories, but which have slightly different names, e.g. "Description" and "Summary". As a convenience, Tasktop has mapped many of the task attributes to a common ID which can be used in place of the name. For the example above, if qc.example.com's repository's "Description" task attribute and rtc.example.com's "Summary" task attribute both have the ID of "task.summary.com", then it can be written like this:

```
<attribute-mapping id="task.common.summary"/> <!-- one line for both -->
```

## Mixing Names and IDs

It is completely possible to mix task attribute names with task attribute IDs. For example, if your company uses qc.example.com's "Note" task attribute in the same way that rtc.example.com uses the "Description" field, you could do something like this:

```
<attribute-mapping>
  <attribute key="Note"/>          <!-- first repository -->
  <attribute id="task.common.description"/> <!-- second repository -->
</attribute-mapping>
```

## Attribute Mapping Strategy

By default, task attribute mappings cause task attribute values to be copied to the destination repository. However, there are several other mapping strategies that can be employed.

```
<attribute-mapping strategy="copy">
```

This is the default strategy, and doesn't need to be specified explicitly. In this case, the source attribute is copied to the target attribute. If the target attribute does not exist, the attribute is not synchronized.

```
<attribute-mapping strategy="initialize">
```

This strategy copies the source value to the target attribute only if the target task is new. This is useful for the purpose of initializing task attributes when the task is first synchronized. The attribute mapping is skipped for subsequent synchronizations.

```
<attribute-mapping strategy="ignore">
```

This strategy ignores this attribute. Applying this strategy at the `attribute-mapping` level is equivalent to not specifying an attribute mapping at all. However, it can be useful to explicitly document that not synchronizing a task attribute was deliberate.

Each of these strategies can also be applied to one end of the mapping by specifying the strategy in the `attribute` sub-element of `attribute-mapping`. For example, to map the `task.common.description` attribute in one direction only, from the first repository to the second repository, you could use the following mapping:

```
<attribute-mapping>
  <attribute id="task.common.description" strategy="ignore"/>
  <attribute id="task.common.description"/>
</attribute-mapping>
```

Additionally there is an `always` copy strategy that can be applied at the attribute level. This strategy always copies the source value to the target attribute whether or not a change has been detected on the attribute. This is useful if you have tasks that move between different queries or synchronization scopes during the task's lifecycle.

This strategy can only be applied one-way. To map the `task.common.summary` attribute from the first repository to the second repository, you would use the following mapping:

```
<attribute-mapping>
  <attribute id="task.common.summary" strategy="ignore"/>
  <attribute id="task.common.summary" strategy="always"/>
</attribute-mapping>
```

## Attribute Mapping Conflict Resolution

Tasktop Sync gives you the flexibility to specify a conflict resolution policy on an attribute-by-attribute basis. This overrides any setting of `conflict-resolution-policy` in the task mapping.

For example, with the following attribute mapping, the `Note` attribute in the first repository would always "win" in a conflict with the `Description` attribute of the second repository, regardless of the task mapping's global conflict resolution policy.

```
<attribute-mapping conflict-resolution-policy="repository-1-dominates">
  <attribute key="Note"/>      <!-- first repository -->
  <attribute key="Description"/> <!-- second repository -->
</attribute-mapping>
```

## Transforming Task Attribute Values with Casters

Frequently, repositories will have slightly different formats for the same information. Casters are used to transform task attribute values from one format to another. Tasktop Sync comes with several built-in casters for common transformations.

### Date-Time Caster

For repositories that represent date values differently, the `date-time` caster is used. For example, one repository might store date values in seconds since the 1 January 1970 epoch, while another might store dates as strings like "5:32 pm, Wed, 5 October 1963". In order to cast the values from one format to another, there are several built-in casters which can be specified.

The following can be used to transform task attribute values from epoch date to string date/time format, and vice-versa:

```
<attribute-mapping>
  <attribute key="Date" caster="date-time"> <!-- first repository's task attribute -->
    <caster-configuration>
      <source-format>epoch</source-format>
      <target-format>yyyy-MM-dd'T'HH:mm:ss.SSSZ</target-format>
    </caster-configuration>
  </attribute>
  <attribute key="DateTime" caster="date-time"> <!-- second repository's task attribute-->
    <caster-configuration>
      <source-format>yyyy-MM-dd'T'HH:mm:ss.SSSZ</source-format>
      <target-format>epoch</target-format>
    </caster-configuration>
  </attribute>
</attribute-mapping>
```

Note that it is not common to need a Date-Time caster; the Tasktop connectors handle almost all date-time conversions.

## Value Mapping Caster

For task attributes that can have a value from a finite set of choices, you can use the `value-map` caster. This is useful for task attributes that are presented using a drop-down in the repository's native application's GUI.

For example, one repository might have priority values of `P1`, `P2`, `P3`, `P4`, and `P5`, while the other repository has values of `Urgent`, `High`, `Medium`, `Low`, and `WontFix`. The following example causes these values to be mapped correctly:

```
<attribute-mapping>
  <attribute key="Priority" caster="value-map"> <!-- first repository -->
    <caster-configuration>
      <map map-by-label="true" ignore-case="false">
        <entry key="Urgent" value="P1"/>
        <entry key="High" value="P2"/>
        <entry key="Medium" value="P3"/>
        <entry key="Low" value="P4"/>
        <entry key="WontFix" value="P5"/>
      </map>
    </caster-configuration>
  </attribute>
  <attribute key="Priority" caster="value-map"> <!-- second repository -->
    <caster-configuration>
      <map map-by-label="true" ignore-case="false">
        <entry key="P1" value="Urgent"/>
        <entry key="P2" value="High"/>
        <entry key="P3" value="Medium"/>
        <entry key="P4" value="Low"/>
        <entry key="P5" value="WontFix"/>
      </map>
    </caster-configuration>
  </attribute>
</attribute-mapping>
```

Note that the `key` specifies the incoming value, and that it is mapped to `value` before being saved in the target attribute. In the above example, this means that the values `P1`, `P2`, `P3`, `P4`, `P5` are associated with the `Priority` attribute in the first repository, and the values `Urgent`, `High`, `Medium`, `Low`, `WontFix` are associated with the `Priority` attribute in the second repository.

Also note the attributes that are present in the `map` element. The `map-by-label="true"` attribute indicates that the option labels are used, as opposed to the option identifiers. The `ignore-case="false"` attribute indicates that the case of the option will not be ignored when mapping. For more details on the `map-by-label` and `ignore-case` settings, see [Value Mapping](#).

## External Value Mapping Caster

For task attributes that can have a value from a finite set of choices, you can also use the `external-value-map` caster. This caster works the same as the [Value Mapping Caster](#), only it allows you to define the mappings in external properties files.

The following example is for the same mapping as in [Value Mapping Caster](#) example:



```

    <attribute-mapping>
      <attribute key="Priority" caster="external-value-map"> <!-- first repository -->
        <caster-configuration>
          <map map-by-label="true" ignore-case="false" file="priority1.properties"
>
          </caster-configuration>
        </attribute>
      <attribute key="Priority" caster="external-value-map"> <!-- second repository -->
        <caster-configuration>
          <map map-by-label="true" ignore-case="false" file="priority2.properties"
/>
          </caster-configuration>
        </attribute>
      </attribute-mapping>

```

The priority1.properties file:

```

Urgent=P1
High=P2
Medium=P3
Low=P4
WontFix=P5

```

The priority2.properties file:

```

P1=Urgent
P2=High
P3=Medium
P4=Low
P5=WontFix

```

Note that the properties files are located as a reference of the `synchronizer.xml` file. So in this example, they reside in the same directory.

Also note the attributes that are present in the `map` element. The `map-by-label="true"` attribute indicates that the option labels are used, as opposed to the option identifiers. The `ignore-case="false"` attribute indicates that the case of the option will not be ignored when mapping. For more details on the `map-by-label` and `ignore-case` settings, see [Value Mapping](#).

## Text

Many repositories support rich text, especially in the description attribute. Different repositories store the representation of rich text in different formats, so to provide high-fidelity rendering of that text after synchronization, Tasktop Sync provides text casters, e.g.:

```

<attribute-mapping>
  <attribute id="task.common.description" caster="confluence-to-html"/>
  <attribute id="task.common.description" caster="html-to-confluence"/>
</attribute-mapping>

```

Examples of text casters:

- `confluence-to-html`
- `html-to-confluence`
- `textile-to-confluence`
- `html-copy`

If one of the repositories uses plain text, then we recommend using the textile caster. Textile is a wiki markup language that is easily readable as plain text. If both repositories use HTML to represent rich text, then we recommend using the `html-copy` caster. The `html-copy` caster handles differences in HTML generated by repositories from different vendors.

## Literal Values

Occasionally it is useful to have a task attribute set to an arbitrary value. You can do this by specifying a literal value. For example, if tasks from one repository should always have the task attribute `task_type` set to "Defect", and the other repository should have the `TaskType` attribute set to "Bug", the task attribute mapping could be specified as follows:

```
<attribute-mapping>
  <attribute key="task_type" caster="Defect"/> <!-- first repository -->
  <attribute key="TaskType" caster="Bug"/>      <!-- second repository -->
</attribute-mapping>
```

If only the second repository should have a literal value, you can use an initialization strategy as follows:

```
<attribute-mapping>
  <attribute id="task_type"/>
  <attribute id="TaskType" strategy="initialize" initial-value="Bug"/>
</attribute-mapping>
```

as before, this will cause the attribute to be initialized with the literal "Bug"

## Task Link To Task Link Caster

When synchronizing tasks with relationships, you can maintain the relationships in both repositories using the `task-link-to-task-link` caster. For example, if a defect is created with references to a test case, and you are synchronizing both the test cases and defects, you can keep their relationships in both repositories. Each task will expose attributes representing these relationships. Create a mapping between each relationship attribute and use the `task-link-to-task-link` caster to maintain the relationships in the two repositories.

If it is known that the relationship references tasks that will not be synchronized by Tasktop Sync, or that there are restrictions on accepted relationship types, then the caster can be configured to filter references to undesired tasks by setting `filter-links` to `true`. When enabled, the caster will ignore any references to tasks that will not be synchronized by the Tasktop Sync instance, or references to tasks that are not of the desired type.

```
<attribute-mapping>
  <attribute caster="task-link-to-task-link" id="linked-entities">
    <caster-configuration>
      <filter-links>true</filter-links>
    </caster-configuration>
  </attribute>
  <attribute caster="task-link-to-task-link" id="Defects">
    <caster-configuration>
      <filter-links>true</filter-links>
    </caster-configuration>
  </attribute>
</attribute-mapping>
```

This caster will only work between two attributes of type "taskDependency".

## Task Link To Task Link Group Caster

When synchronizing tasks with multiple relationship attributes into a task with a single relationship attribute, you can still maintain the relationships in both repositories using the `task-link-to-task-link-group` caster, along with the `task-link-to-task-link` caster. For example, if a defect is created with both "clones" and "blocks" relationships, and you are synchronizing to a target repository defect that supports only "linked-entities" relationships, you can keep combine all the relationships from "clones" and "blocks" into the "linked-entities" attribute.

This caster can only exist as a compliment to an existing mapping using a `task-link-to-task-link` caster. You first create the `task-link-to-task-link` mapping between first of the many attributes to your single target attribute (see [Task Link To Task Link Caster](#) ). You then define a unidirectional mapping from your subsequent of the many attributes to the same single target attribute, including the configuration of the `group-default-attribute-id` containing the first of the many attributes. For more details, see [Task Link to Task Link Group](#).

```
<attribute-mapping>
  <attribute caster="task-link-to-task-link" id="clones"/>
  <attribute caster="task-link-to-task-link" id="linked-entities"/>
</attribute-mapping>
<attribute-mapping>
  <attribute strategy="ignore" caster="task-link-to-task-link-group" id="blocks"/>
  <attribute caster="task-link-to-task-link-group" id="linked-entities">
    <caster-configuration>
      <group-default-attribute-id>clones</group-default-attribute-id>
    </caster-configuration>
  </attribute>
</attribute-mapping>
```

This caster will only work as a unidirectional mapping between two attributes of type "taskDependency". The `caster-configuration` must exist on the target attribute of the unidirectional mapping, and its `group-default-attribute-id` must contain a valid source attribute id of an existing mapping that uses the `task-link-to-task-link` caster.

## Task Link To Web Link Caster

When synchronizing a task with internal references to other tasks, you may not want to synchronize the referenced tasks, but instead simply synchronize references to those tasks as web URLs. For example, if a defect is created with references to one or more test cases, you may want to synchronize the defect but not the test cases. Instead, you can synchronize the referenced test cases as web URLs that when clicked will open the test case in the source repository.

```
<attribute-mapping>
  <attribute strategy="ignore" caster="task-link-to-web-link" id="Test"/>
  <attribute caster="task-link-to-web-link" id="web-links">
    <caster-configuration>
      <target-format>%task.common.summary (%ID) - %task.common.status (%DATE)</target-format>
    </caster-configuration>
  </attribute>
</attribute-mapping>
```

This cast is a unidirectional cast from a task link to a web link. Therefore, the source attribute must have `strategy="ignore"` and must be of type "taskDependency". The target attribute must be of type "externalLink".

The `target-format` is used to format the label of the web link. The format can contain any clear text and the following wild cards:

Wild Card	Description
<code>%URL</code>	Will be replaced by a web URL representing the referenced task.
<code>%ID</code>	Will be replaced by the task ID of the referenced task.
<code>%KEY</code>	Will be replaced by the task key of the referenced task.
<code>%DATE</code>	Will be replaced by the system date of the Tasktop Sync server at synchronization time.
<code>%TIME</code>	Will be replaced by the system time of the Tasktop Sync server in 24-hour format at synchronization time.

Additionally any attributes on the linked task can be referenced by common key or attribute id. Please refer to the [schema](#) for a list of available attribute IDs.

For example, using the format `%KEY: %summary (%task.common.status) as of %DATE` will produce:

```
Task-1234: Deliver Presents (In Progress) as of 2015-12-25
```

When the linked task has a summary of "Deliver Presents" and a status of "In Progress".

It is important to note that when task attributes are present in the target link label format, Tasktop Sync will retrieve the linked tasks during synchronization to get the information needed to create the link label. It is recommended that the [synchronizations strategy](#) be set to always for mappings which use attribute IDs in the link label to keep the link labels as up to date as possible as the linked tasks change.

If no format is specified, the link label will not be filled out.

## Task Link To Web Link Group Caster

When synchronizing tasks with multiple relationship attributes into a target task's single URL attribute, use the `task-link-to-web-link-group` caster, along with the `task-link-to-web-link` caster. For example, if a defect is created with both "clones" and "blocks" relationships, and you are synchronizing to a target repository defect that supports only "url-attachments" relationships, you can keep combine all the relationships from "clones" and "blocks" into the "url-attachments" attribute.

This caster can only exist as a compliment to an existing mapping using a `task-link-to-web-link` caster. You first create the `task-link-to-web-link` mapping between first of the many attributes to your single target attribute (see [Task Link To Web Link Caster](#)). You then define a unidirectional mapping from your subsequent of the many attributes to the same single target attribute, including the configuration of the `group-default-attribute-id` containing the first of the many attributes. For more details, see [Task Link to Web Link Group](#).

```

<attribute-mapping>
  <attribute strategy="ignore" caster="task-link-to-web-link" id="clones"/>
  <attribute caster="task-link-to-web-link" id="url-attachments"/>
</attribute-mapping>
<attribute-mapping>
  <attribute strategy="ignore" caster="task-link-to-web-link-group" id="blocks"/>
  <attribute caster="task-link-to-web-link-group" id="url-attachments">
    <caster-configuration>
      <group-default-attribute-id>clones</group-default-attribute-id>
    </caster-configuration>
  </attribute>
</attribute-mapping>

```

This cast is a unidirectional cast from a task link to a web link. The `caster-configuration` must exist on the target attribute of the unidirectional mapping, and its `group-default-attribute-id` must contain a valid source attribute id of an existing mapping that uses the `task-link-to-web-link` caster.

## Task Link To String Caster

If you want to use the Task Link to Web Link caster, but your target repository does not support web URL attachments, you can format the references into text attributes using the `task-link-to-string` caster.

```

<attribute-mapping>
  <attribute strategy="ignore" caster="task-link-to-string" id="Parents"/>
  <attribute caster="task-link-to-string" id="LongText">
    <caster-configuration>
      <target-format>%URL%,</target-format>
    </caster-configuration>
  </attribute>
</attribute-mapping>

```

This cast is a unidirectional cast from a task link. Therefore, the source attribute must have `strategy="ignore"`, and must be of type `"taskDependency"`. The target attribute must support text context, and it is highly recommended that the attribute support strings of at least 1024 characters.

The `target-format` is used to format each task URL. Multiple task URLs will each be formatted using `target-format` and then sequentially concatenated. The format can contain any clear text and the following two wild cards:

Wild Card	Description
%URL	Will be replaced by a web URL representing the referenced task.
%,	Places a comma in all formats except the last one. Useful for separating a list of task URLs.
%EOL	Inserts a line break if the target attribute supports them. Useful for formatting a list of task URLs.

For example, using the format `Test: %URL%,` on the two task URLs `"http://www.repository.com/1"` and `"http://www.repository.com/2"`, the resulting text attribute will be:

Test: <http://www.repository.com/1>, Test: <http://http://www.repository.com/2>

If no format is specified, a default of %URL%, is used, which will list the web URLs separated by commas.

## Task Link To OSLC Link Caster

When synchronizing a task with internal references to other tasks, you may not want to synchronize the referenced tasks, but instead simply synchronize references to those tasks as OSLC Links. For example, if a defect is created with references to one or more test cases, you may want to synchronize the defect but not the test cases. Instead, you can synchronize the referenced test cases as OSLC Links which when presented in the target system provide information about the linked tasks using Tasktop Sync's OSLC Adapter.

```
<attribute-mapping>
  <attribute strategy="ignore" caster="task-link-to-oslc-link" id="Test"/>
  <attribute caster="task-link-to-oslc-link" id="web-links"/>
</attribute-mapping>
```

This cast is a unidirectional cast from a task link to an OSLC link. Therefore, the source attribute must have `strategy="ignore"` and must be of type "taskDependency". The target attribute must be of type "externalLink". For these links to provide information about the linked tasks Tasktop Sync's OSLC Adapter must be configured and running, the target system must support OSLC linking and be configured to accept OSLC links from Tasktop Sync, and the link target must be exposed via Tasktop Sync's OSLC Adapter. For further information on OSLC linking see the section on [Task Linking Configuration](#).

## Value to Task Link Caster

To synchronize a data enumeration with an internal task reference, use the `value-to-task-link` caster.

```
<attribute-mapping>
  <attribute id="values" caster="value-to-task-link">
    <caster-configuration>
      <query>Categories</query>
      <map>
        <entry key="13312" value="10700"/>
      </map>
    </caster-configuration>
  </attribute>
  <attribute id="tasks" caster="value-to-task-link">
    <caster-configuration>
      <query>Categories</query>
      <map>
        <entry key="10700" value="13312"/>
      </map>
    </caster-configuration>
  </attribute>
</attribute-mapping>
```

Note that this caster will only work between an attribute of type "Single Select" or "Multi Select", and an attribute of type "Task Dependency".

## Web Link To Web Link Caster

To synchronize attached web URLs between tasks, use the `web-link-to-web-link` caster.

```
<attribute-mapping>
  <attribute caster="web-link-to-web-link" id="related-links"/>
  <attribute caster="web-link-to-web-link" id="web-links"/>
</attribute-mapping>
```

The caster will only work between two attributes of type "externalLink".

## Web Link To String Caster

If you want to use the Web Link to Web Link caster, but your target repository does not support web URL attachments, you can format the references into text attributes using the `web-link-to-string` caster.

```
<attribute-mapping>
  <attribute strategy="ignore" caster="web-link-to-string" id="Parents"/>
  <attribute caster="web-link-to-string" id="LongText">
    <caster-configuration>
      <target-format>%URL%,</target-format>
    </caster-configuration>
  </attribute>
</attribute-mapping>
```

This cast is a unidirectional cast from a web link. Therefore, the source attribute must have `strategy="ignore"`, and must be of type "externalLink". The target attribute must support text context, and it is highly recommended that the attribute support strings of at least 1024 characters.

The `target-format` is used to format each web URL. Multiple web URLs will each be formatted using `target-format` and then sequentially concatenated. The format can contain any clear text and the following two wild cards:

Wild Card	Description
%URL	Will be replaced by then web URL from the source.
%LABEL	Will be replaced by the URL's label (or name), if it exists.
%,	Places a comma in all formats except the last one. Useful for separating a list of web URLs.
%EOL	Inserts a line break if the target attribute supports them. Useful for formatting a list of web URLs.

For example, using the format `Test: %URL%,` on the two web URLs "`http://www.repository.com/1`" and "`http://www.repository.com/2`", the resulting text attribute will be:

Test: <http://www.repository.com/1>, Test: <http://www.repository.com/2>

If no format is specified, a default of `%URL%,` is used, which will list the web URLs separated by commas.

## Location To Web Link Caster

If you need to have the URL of the source task as a web link on the target task, use the `location-to-web-link` caster. This caster will create a web link, which points to the source task, on the target task.

```
<attribute-mapping>
  <attribute strategy="copy" id="web-links" caster="location-to-web-link">
    <caster-configuration>
      <target-format>%KEY - %SUMMARY</target-format>
    </caster-configuration>
  </attribute>
  <attribute strategy="ignore" id="location" caster="location-to-web-link"/>
</attribute-mapping>
```

This cast is a unidirectional cast from a location. Therefore, the source attribute must have `strategy="ignore"`, and must be of type "url". The target attribute must be of type "externalLink".

The `target-format` is used to format the label of the web link. The format can contain any clear text and the following wild cards:

Wild Card	Description
%URL	Will be replaced by the source task's URL.
%SUMMARY	Will be replaced by the source task's summary.
%KEY	Will be replaced by the source task's key.

For example, using the format `%KEY - %SUMMARY (%URL)` on a task with URL "http://www.repository.com/1", summary "Task 1" and key "T-1" will result in a web link with the following label:

T-1 - Task 1 (<http://www.repository.com/1>)

If no format is specified, a default of `%KEY - %SUMMARY` is used.

Note: If the summary is used in the label, it is recommended that the strategy be set to "always" to ensure the label of web link stays up to date. For more details, see the [Attribute Mapping Strategy](#) section.

## Status Transition Caster

If you need to synchronize status into a repository with a workflow use the `status-transition` caster. With this caster you can synchronize status by executing transitions to move the target task between statuses following the workflow defined in the remote repository.

```
<attribute-mapping>
  <attribute caster="status-transition" id="task.common.status" strategy="ignore" />
  <attribute caster="status-transition" id="task.common.status">
    <caster-configuration>
      <map>
        <entry key="status.new" value="1-New" />
        <entry key="status.assigned" value="2-Active" />
        <entry key="status.resolved" value="3-Closed" />
        <entry key="status.reopened" value="4-Reopened" />
      </map>
    </caster-configuration>
  </attribute>
</attribute-mapping>
```



```

id="task.common.operation-accept"/>
id="task.common.operation-close"/>
operation-id="task.common.operation-resolve"/>
operation-id="task.common.operation-reopen"/>
operation-id="task.common.operation-accept"/>
operation-id="task.common.operation-resolve"/>
</graph>
<creation-value>1-New</creation-value>
</caster-configuration>
</attribute>
</attribute-mapping>
<edge from-option-id="1-New" to-option-id="2-Active" operation-
<edge from-option-id="1-New" to-option-id="3-Closed" operation-
<edge from-option-id="2-Active" to-option-id="3-Closed"
<edge from-option-id="3-Closed" to-option-id="4-Reopened"
<edge from-option-id="4-Reopened" to-option-id="2-Active"
<edge from-option-id="4-Reopened" to-option-id="3-Closed"

```

To configure this caster both a value mapping and a transition graph must be specified. The value mapping is used to translate the status of the source task into a desired status on the target task. The transition graph is used to determine the sequence of transitions which must be executed to attain the desired status.

For the synchronization to proceed a path must exist in the graph between the current and desired statuses. If no such path can be found then the synchronization will fail. For instance in the above example no path exists from 3-Closed to 1-New. This means that if the target task had the 3-Closed status, and the source task's status was changed to `status.new`, the synchronization would fail as no sequence of transitions will leave the target task with the 1-New status.

All edges defined in the graph are unidirectional. If transitions exist between statuses in both directions then one edge must be configured for each direction. This has been done in the above example configuration where the transition `task.common.operation-resolve` exists from 4-Reopened to 3-Closed while the transition `task.common.operation-reopen` exists from 3-Closed to 4-Reopened.

If more than one transition is required to attain the desired status Sync will execute them one at a time in order to reach the desired status. In the example above if the target task has the 1-New status, and the source task has the `status.reopened` status then the following transitions will be executed: `task.common.operation-close`, `task.common.operation-reopen`. This will change the status of the target task first from 1-New to 3-Closed, then from 3-Closed to 4-Reopened. The caster will always choose the path with the fewest number of transitions, this is why the target task never attained 2-Active status as this would require three transitions instead of two.

For repositories that require an initial status on new task creation, use `creation-value` to assign this status. The status should represent the starting status on the target repository's workflow.

This caster is a unidirectional caster only. Therefore, the source attribute must have `strategy="ignore"`. If you need to synchronize workflows in both directions create two attribute mappings with this caster, one in each direction. This is done to facilitate mappings where status transitions are required in one direction, but a different caster may be used in the reverse direction.

This caster is only supported when mapping the status field, and only transitions affecting status are currently supported.

## Caster Customization

Casters can also be written in an external scripting language such as Groovy or JavaScript; we recommend Groovy. Here is an example:

```
<caster name="toUpper" language="groovy"><![CDATA[  
  
    def cast(value) {  
        value.toUpperCase()  
    }  
  
]]></caster>
```

The above caster must be declared after any `<task-mapping>` declarations. You can refer to this caster in an attribute mapping like this:

```
<attribute-mapping>  
  <attribute id="Summary" caster="toUpper"/>  
  <attribute id="Summary"/>  
</attribute-mapping>
```

Casters can also be written in external files and included via the `script` attribute of the `caster` element:

```
<caster  
  name="toUpper"  
  language="groovy"  
  script="scripts/statusHandler.groovy"/>
```

The path to the Groovy script is resolved relative to the Tasktop Sync working directory. The above caster would look for a directory called `scripts` alongside the `synchronizer.xml` file, and it would look for a file called `statusHandler.groovy` in that `scripts` directory.

For more details on scripting in Tasktop Sync, see [Advanced Scripting](#)

## Using Person Mappings

For repository attributes representing people, you can use a person caster if a person mapping has been specified in the enclosing `task-mapping` element:

```
<attribute-mapping>  
  <attribute id="AssignedTo" caster="person"/>  
  <attribute id="AssignedTo" caster="person"/>  
</attribute-mapping>
```

The person caster uses the person mapping named in the enclosing `task-mapping` element to cast its values.

## Advanced Concepts

### One-Way Mappings

Tasktop Sync's `attribute-mapping` can be used to support several different configurations of one-way synchronization of objects. Below are several examples using attribute mapping strategies to realize a one-way mapping.

The first example is a generic one-way mapping of the status attribute from the first repository into the second repository:

```
<attribute-mapping>
  <attribute id="task.common.status" strategy="ignore" />
  <attribute id="task.common.status" />
</attribute-mapping>
```

This causes a status attribute change in the first repository to be synchronized to the status attribute in the second repository, but not vice versa.

To populate the status attribute of the task in the second repository only the first time it is created, you can use the initialization strategy as follows:

```
<attribute-mapping>
  <attribute id="task.common.status" />
  <attribute id="task.common.status" strategy="initialize" />
</attribute-mapping>
```

This will cause the attribute to be initialized with a copy from the first repository, with subsequent changes being ignored.

If you want to synchronize an attribute one way with a literal value, so that no source attribute is specified, then you can leave off the `id` and `key` properties on one of the attributes as long as the strategy for that attribute is set to `ignore`. For example:

```
<attribute-mapping>
  <attribute strategy="ignore" />
  <attribute id="BG_SEVERITY" caster="'2-Medium'" strategy="initialize" />
</attribute-mapping>
```

This will cause the `BG_SEVERITY` attribute to be initialized to the literal value "2-Medium".

## Synchronizing Subsets of Queried Incoming Tasks

Sometimes you might want to only synchronize a subset of the tasks found in your specified queries. For example, you might only be interested in synchronizing the Cobra project defects -; not anything about the Mongoose project, not anything about the Cobra test cases. You can restrict which subset of entries in the repositories to synchronize by using the `nary` element in conjunction with `condition` elements, both inside the `repository` element.

For example, to specify that only tasks which have a value of `Requirement` or `TestCase` in the `Type` task attribute will be synchronized in our example HPE QC / ALM repository we could rewrite that `repository` element as follows:

```
<repository proxy-creation-trigger="all"
  auto-comment-user="developer1"
  id="com.tasktop.sync.QC"
  url="http://qc.example.com:8080/qcbin;SYNCHRONIZER;ProjectArea">
  <property name="defaultCommentUser" value="developer1" />
```

```

<scope>
  <field id="itemType" value="BUG"/>
</scope>
<source>
  <query name="initialization" value="Tasktop Sync - HPE QC / ALM - All Defects"/>
  <query name="changes" value="Tasktop Sync - HPE QC / ALM - Recently Modified Defects"/>
</source>
<proxy-storage store="attribute" attribute-id="BG_USER_01"/>
<nary operator="or">
  <condition attribute-key="type" operator="=" value="Requirement"/>
  <condition attribute-key="type" operator="=" value="TestCase"/>
</nary>
</repository>

```

The `nary` element can use one of three operators: `or`, `and`, or `!`. The `or` operator evaluates to true when at least one of the conditions within it evaluates to true. The `and` operator evaluates to true when each and every condition within it evaluates to true. The `!` operator inverts the truth value of the contained condition (i.e. true becomes false and false becomes true). A `nary` element may also be nested within another `nary` as one of its conditions.

The `condition` element must use one of `attribute-id`, `attribute-key`, or `attribute-label` as the identifier for the attribute to filter by. The value for the operator to evaluate can be set using `value` or with `value-label` to find the corresponding value option identified through label.

The operators available for the `condition` element are summarized in the following table:

Operator	Type of Attribute	Description
=	Single-value	True when the value provided equals the value of the specified attribute.
!=	Single-value	True when the value provided does not equal the value of the specified attribute.
null	Single-value	True when the value of the specified attribute is empty, or the specified attribute doesn't exist.
equals_ignore_case	Single-value	True when the value provided equals the value of the specified attribute, ignoring case.
contains	Single-value	True when the value of the specified attribute contains the value provided. For example, "abc" contains "bc" but not "ac".
any-contains	Multi-value	True when at least one of the values in the specified multi-value attribute contains the value provided.
all-contains	Multi-value	True when each and every value in the the specified multi-value attribute contains the value provided.
matches	Single-value	True when the value of the specified attribute matches the given regular expression.

any-matches	Multi-value	True when at least one of the values in the specified multi-value attribute matches the given regular expression.
all-matches	Multi-value	True when each and every value in the specified multi-value attribute matches the given regular expression.

## Scripted Attribute Handlers

If you need even more flexibility than the normal attribute mapping, you can write a scripted attribute handler. This is particularly useful if there is not a one-to-one mapping between some of the task attributes in the two repositories. For example, if one repository has a single "mailing address" task attribute but the other has three separate task attributes for "street", "city", and "country", you could write an attribute handler to synchronize those task attributes.

The scripting languages that are available to you include JavaScript and Groovy; if you have other scripting languages available on your system, those might be available to you as well. Between JavaScript and Groovy, we recommend that you use Groovy.

```
<attribute-handler language="groovy" id="task.common.status">
  <script><![CDATA[

def map(context,sourceAttribute,targetTask) {
  if (targetTask.root.attributes['bug_status']?.value != sourceAttribute.value ||
      (sourceAttribute.value == 'RESOLVED' && targetTask.root.attributes['resolution']?.value !=
sourceAttribute.parent.attributes['resolution']?.value)) {
    if (targetTask.isNew()) {
      //
    } else {
      def operation = targetTask.root.attributes['task.common.operation']
      switch (sourceAttribute.value) {
        case 'ASSIGNED':
          operation.value = 'accept';
          break;
        case 'VERIFIED':
          operation.value = 'verify';
          break;
        case 'RESOLVED':
          operation.value = 'resolve';
          targetTask.root.attributes['resolution'].value =
            sourceAttribute.parent.attributes['resolution']?.value;
          break;
        case 'CLOSED':
          operation.value = 'close';
          targetTask.root.attributes['resolution'].value =
            sourceAttribute.parent.attributes['resolution']?.value;
          break;
        case 'REOPENED':
          operation.value = 'reopen';
          break;
        case 'NEW':
          operation.value = 'marknew';
          break;
        case 'DUPLICATE':
          operation.value = 'resolve';
          targetTask.root.attributes['resolution'].value =
            sourceAttribute.parent.attributes['resolution']?.value;
          break;
      }
      if (targetTask.root.attributes['task.common.operation-'+operation.value] == null) {
        targetTask.root.createAttribute('task.common.operation-'+operation.value)
      }
    }
  }
}

]
```

```
    ]]></script>
</attribute-handler>
```

Scripted attribute handlers can also be written in external files and included via the `script` attribute of the `attribute-handler` element:

```
<attribute-handler
  language="groovy"
  id="task.common.status"
  script="scripts/statusHandler.groovy"/>
```

The location of the Groovy script specified in the `scripts` attribute is resolved relative to the location of the `synchronizer.xml` file. In the above attribute handler, Tasktop Sync will look for a file called `statusHandler.groovy` in a directory called `scripts` which is located in the same directory as the `synchronizer.xml` file.

Note: Tasktop Sync will log warnings for required attributes which are not mapped. If such attributes are mapped using a scripted attribute handler, the attributes must be specified in the attribute handler's configuration in order for the warnings to be removed. Attributes may be identified by `id` or `key`.

```
<attribute-handler
  language="groovy"
  id="task.common.status"
  script="scripts/statusHandler.groovy">
  <target-attributes-handled>
    <attribute id="status"/>
    <attribute key="task.common.summary"/>
  </target-attributes-handled>
</attribute-handler>
```

For more details on scripting in Tasktop Sync, see [Advanced Scripting](#)

## Multiple Submissions per Synchronization

One use case for attribute handlers is to perform more complicated computations during a synchronization than just taking an input from the source attribute and storing an output in the proxy attribute. Many of these computations are related to workflow restrictions in the proxy repository which restrict the values an attribute can be changed to based on its current value. For example, in some repositories, new tasks must have the "New" status. However, if you are synchronizing tasks from an existing repository, many of the source tasks may already be closed; the target repository will then prevent proxy tasks from being created because it won't allow new tasks to be created in the "closed" state.

To address this issue, Tasktop Sync allows scripted attribute handlers to request that the scripts be applied again once all of the current changes have been submitted to the repositories. This allows a script to step through workflow restrictions and submit each step along the way, effectively allowing a closed task from one repository to create a new proxy task in the closed state in the target repository. Here is an example Groovy script which causes a target "counter" attribute to be incremented by one each time the script is run, stopping at 10. This script will cause 10 submissions to the target repository, one for each value from 1 to 10.

```

<attribute-handler language="groovy" id="counter">
  <script><![CDATA[
    def map(context,sourceAttribute,targetTask) {
      if (targetTask.root.attributes['counter'] == null) {
        targetTask.root.createAttribute('counter')
        targetTask.root.attributes['counter'].setValue('1')
        context.setSyncAgain(true)
      } else if (targetTask.root.attributes['counter'].value.toInteger() < 10) {
        targetTask.root.attributes['counter'].setValue((targetTask.root.attributes['counter'].value.
toInteger() + 1).toString())
        context.setSyncAgain(true)
      }
    }
  ]]></script>
</attribute-handler>

```

There is one important feature to notice about this script that is crucial in how it works. The script must call `context.setSyncAgain(true)` to request that another round of processing occur after the current one. This also causes the script mapping to be applied even if the source attribute hasn't changed since the last synchronization. Otherwise, each round of processing after the first would not apply the script mapping since the source attribute hasn't changed since the last round.

Note that there is a hard limit on the number of iterations per complete synchronization that Tasktop Sync will allow, to prevent erroneous scripts from submitting indefinitely to any task repository. This limits the number of rounds of synchronization to 15.

#### Ignoring Automated Changes

When performing a synchronization involving multiple submissions it is usually important that the synchronization is not interrupted by other changes. This is especially true when submitting to attributes whose change may alter other attributes. A common case for this is when synchronizing status attributes with a workflow, the workflow constraints may require that multiple submissions are performed, for example to move from 'Open' to 'Reopened' a task may need to proceed through the following states 'Open', 'In Progress', 'Closed', and 'Reopened'. During such a procession it is important that changes in the proxy repository do not cause any interruption. This is especially important when the attribute handler does not submit to the status attribute directly but instead triggers a transition which may alter multiple attributes on the task being submitted to.

```

<attribute-handler language="groovy" id="status">
  <script><![CDATA[
    def map(context,sourceAttribute,targetTask) {
      def transitions = computePath(sourceAttribute, targetTask)
      if (transitions.length == 1) {
        targetTask.root.attributes['status-transition'] = transitions[0]
      } else if (transitions.length > 1) {
        targetTask.root.attributes['status-transition'] = transitions[0]
        context.setSyncAgain(true)
        context.ignoreTargetAttribute('status')
      }
    }
  ]]></script>
</attribute-handler>

```

In the above example the script computes the transitions it must perform and then performs the first transition as well as requesting a subsequent synchronization to perform the rest of the transitions. However this transition will change the value of the proxy task's status attribute which will cause Sync to attempt to synchronize this change back to the source. To avoid this interruption of the multiple synchronization submissions the script instructs Sync to ignore changes to the target attribute 'status'

during the reverse synchronization. All other attributes will be considered during the reverse synchronization and any changes to the 'status' attribute will be considered during any subsequent synchronizations.

## Sparse Attributes

Occasionally connectors store an attribute as *sparse* meaning that such attribute is not present on a task when there is no existing value for the attribute. Tasktop Sync has a mechanism for properly detecting erasure of the value of a sparse task attribute. The mechanism is enabled through using the *sparse* property of an attribute element of an attribute-mapping.

```
<attribute-mapping strategy="copy">
  <attribute id="BG_USER_04" />
  <attribute id="task.sparse.attribute" sparse="true" />
</attribute-mapping>
```

In most cases, using the *sparse* property will not be required and should not be used. Please refer to the connector-specific documentation to determine if the *sparse* property is required.

## Operation Attribute Mappings

When using the [Status Transition Caster](#) certain transitions may require extra information to execute. This information can be provided by using operation attribute mappings. One common scenario where this may be necessary is when executing a status transition which resolves or closes a defect. These status transitions often require a resolution to be specified.

An operation attribute mapping provides this information to the transition by mapping attributes from source task attributes to target operation attributes.

```
<operation-attribute-mapping>
  <attribute id="task.common.resolution" />
  <operation-attribute id="task.common.resolution" operation-id="task.common.operation-resolve" />
</operation-attribute-mapping>
```

The above example maps the source task's resolution directly to the resolution attribute of the operation `task.common.operation-resolve`. The `operation-id` must be specified so that the mapping is applied to the correct operation. This means that if there are multiple status transitions which require a resolution attribute to be set a separate operation attribute mapping must be configured for each.

Operation attribute mappings are always one way mappings from the task attribute to the operation attribute. Under no circumstances will synchronizations take place from the operation attribute into the task attribute. As such the `attribute` element in the `operation-attribute-mapping` element cannot have any strategy set on it other than `ignore`, if no strategy is set "ignore" is used by default. Operation attribute mappings may use casters in the same way that task attribute mappings can, the only exclusion is that group casters are not supported.

## Jobs



Scheduled jobs can be configured to run using CRON expressions. The following job types can be scheduled:

Type	Description	Required
StatisticsEmail	Send statistics emails that show detailed information about how much activity there has been in the last 24 hours.	True
MetricsEmail	Sends emails with synchronization activity over the last month.	True
RepositoryConfigurationRefresh	Periodically refreshes the repository configuration of the specified repository.	False
RefreshSchema	Periodically refreshes every repository schema corresponding to the specified repository.	False
CheckForChanges	Periodically downloads all tasks within the specified query and checks for changes. This can be used to identify changes which are not reported by the repository.	False
ErrorNotification	Send error notification emails which show detailed information about errors which have occurred during synchronization.	False
DiskSpaceWarning	Periodically checks the free space on the disk containing the Tasktop Sync workspace to see if it has fallen below a configurable threshold. If it has fallen below the threshold, the job will send a warning email and optionally stop Tasktop Sync.	False

Each job has the following attributes:

Attribute	Description	Sample Value
name	A unique name for the job.	MyJob
type	One of the types defined in the table above.	MetricsEmail
schedule	A CRON expression defining when the job should run.	0 0 0 * * ?

Each job can also have custom parameters, defined as child elements with the following attributes:

Attribute	Description	Sample Value
name	A expected parameter name.	repositoryUrl
value	The value to use for the parameter.	<a href="http://repo.com/">http://repo.com/</a>

For example:

```
<job name="StatisticsEmail" type="StatisticsEmail" schedule="0 0 0 * * ?"/>
<job name="Sample" type="SampleType" schedule="0 0 0 1 * * ?">
  <job-parameter name="sampleParam" value="1"/>
</job>
```

## Refresh Repository Configuration Job

The RepositoryConfigurationRefresh job requires the following parameter:

Name	Value
repository-url	The url of the repository

For example:

```
<job name="RepositoryConfigurationRefresh" type="RepositoryConfigurationRefresh" schedule="0 0 0 * * ?">
  <job-parameter name="repository-url" value="http://repo.com/" />
</job>
```

## Refresh Schema Job

The RefreshSchema job requires the following two parameters:

Name	Value
repository-url	The url of the repository
refresh-type	local or remote

For example:

```
<job name="RefreshSchema" type="RefreshSchema" schedule="0 0 0 * * ?">
  <job-parameter name="repository-url" value="http://repo.com/" />
  <job-parameter name="refresh-type" value="local" />
</job>
```

## Check For Changes Job

The CheckForChanges job requires the following two parameters:

Name	Value
repository-url	The url of the repository
query-name	The name of the query

For example:

```
<job name="CheckForChanges" type="CheckForChanges" schedule="0 0 0 * * ?">
  <job-parameter name="repository-url" value="http://repo.com/" />
```

```
<job-parameter name="query-name" value="defects"/>
</job>
```

## Disk Space Warning Job

The DiskSpaceWarning job requires the following two parameters:

Name	Value
threshold-in-MB	The minimum amount of remaining disk space in MB before the warning email will be sent.
stop-sync-if-below-threshold	Whether this job should stop Tasktop Sync if the remaining disk space is below the threshold

For example:

```
<job name="DiskSpaceWarning" type="DiskSpaceWarning" schedule="0 0 0 * * ?">
  <job-parameter name="threshold-in-MB" value="2000"/>
  <job-parameter name="stop-sync-if-below-threshold" value="false"/>
</job>
```

## Error Notification Job

The Error Notification Job will send an email containing the configured error notification at the provided schedule. For more information on error notifications, see the [Notifications](#) section.

## Notifications

This section describes how to configure the email notifications from the `synchronizer.xml` within Tasktop Sync. The more user friendly way to configure this is detailed in the [Notifications](#) section.

There are three types of notifications which may be sent, each corresponding to an xml element:

Name	Contents
error-notification	Any errors which may have occurred when Tasktop Sync is running
statistics-notification	Statistics on the Tasktop Sync instance, such as uptime and average processing time
metrics-notification	Metrics on the number of synchronizations performed in the last three months

Each notification element may contain any number of email elements, which each have a single address attribute. For example, the following is the configuration of the statistics notification being sent to the email addresses "recipient1@example.com" and "recipient2@example.com".

```
<statistics-notification>
  <email address="recipient1@example.com" />
```

```
<email address="recipient2@example.com" />
</statistics-notification>
```

## Error Notification

The Error notification has additional configuration for the contents of the notification:

Name	Description
include-synchronization-errors	Errors that occurred during synchronization, which may include conditions such as: a proxy task has been deleted, a repository is offline, or a task submission has failed
include-synchronization-conflicts	Conflicts that occurred during synchronization which resulted in the changed attribute not being synchronized
include-query-errors	Errors that resulted from queries being run by Tasktop Sync

For example, the following is the configuration of the error notification configured to only receive synchronization errors and synchronization conflicts, being sent to the email addresses "recipient1@example.com" and "recipient2@example.com".

```
<error-notification include-synchronization-errors="true" include-synchronization-conflicts="true" include-
query-errors="false">
  <email address="recipient1@example.com" />
  <email address="recipient2@example.com" />
</error-notification>
```

## Person Mapping Files

You can map people in one repository to people in the other repository via person mapping files. These XML files have a similar structure to the `task-mapping` element in the `synchronizer.xml` file. Here is an example person mapping file:

```
<?xml version="1.0" encoding="UTF-8"?>
<person-mappings xmlns="http://tasktop.com/xml/ns/sync/person-mapping-model">

  <repository
    url="http://qc.example.com"
    default-person-id="Admin"
    mapping-ignore-case="true"/>
  <repository
    url="http://rtc.example.com"
    default-person-id="administrator"/>

  <!--
  This mapping is for optional attributes. If an optional attribute has no value set
  on it then its synced attribute will also have no value. If the desired behaviour
  is to map an empty value to the default user then remove this mapping.
  -->
  <person-mapping>
    <person id="" />
    <person id="" />
  </person-mapping>
```

```

<person-mapping>
  <person id="dev1"/>
  <person id="dev1@example.com"/>
</person-mapping>
<person-mapping>
  <person id="dev2"/>
  <person id="dev2@example.com"/>
</person-mapping>
</person-mappings>

```

In the above example, the first mapping maps the empty value "" in the first repository to another empty value "" in the second repository. As stated in the comments, this mapping is optional, and can be removed to map blank values to the default-person-id.

dev1 in the repository at <http://qc.example.com> is mapped to [dev1@example.com](mailto:dev1@example.com) in the repository at <http://rtc.example.com>. Similarly, dev2 in <http://qc.example.com> is mapped to [dev2@example.com](mailto:dev2@example.com) in <http://rtc.example.com>. The repository URLs in this file must correspond to the repository URLs used in each task-mapping element in the `synchronizer.xml` file that uses the mapping.

The default-person-id of <http://qc.example.com> is Admin, meaning that any items syncing from <http://rtc.example.com> to <http://qc.example.com> that contain a person identity not included in this mapping will be mapped to Admin on <http://qc.example.com>. For items going from <http://qc.example.com> to <http://rtc.example.com>, the default-person-id is administrator.

The default-person-id will not override any valid person identities that are included in the person mapping file. For example, on an item syncing from <http://rtc.example.com> to <http://qc.example.com>, using the person identity administrator when the item on <http://qc.example.com> already uses dev1 will have no effect on the item on <http://qc.example.com>.

In the case of a multi person attribute mapping the default-person-id will not be synchronized to the target repository. In addition, any person identities in the target repository which are not mapped will not be overwritten. For example, on an item syncing from <http://rtc.example.com> to <http://qc.example.com>, if the RTC multi person attribute contains the values [dev1@example.com](mailto:dev1@example.com) and administrator while the HPE QC / ALM multi person attribute contains the values dev2 and dev3. [dev1@example.com](mailto:dev1@example.com) will map to will be mapped to dev1, administrator will not be mapped, and dev3 will be retained resulting in the HPE QC / ALM multi person attribute having the values dev1 and dev3 set after synchronization has completed.

Setting the mapping-ignore-case attribute to true indicates that Sync should match person identifiers without regard to case. This is useful if the repository allows users to enter person identifiers with the wrong case, such as testuser1 when the real identifier is TestUser1. In example above, the first repository, <http://qc.example.com>, has a user with the identifier dev1, but allows the user to enter their identifier as Dev1 onto tasks. This would normally not match your defined mappings in the Person Mapping file. But since the first repository has mapping-ignore-case="true", it will allow Sync to match the task's value Dev1 with the Person Mapping configured person dev1. If this attribute is not found on the configuration, the default value is false.

NOTE: Be careful to only set `mapping-ignore-case` to `true` on repositories that match person identifiers without regard to case. If set incorrectly, you may encounter errors matching the incorrect person identifier. For instance, if there exists an old mapping to a deactivated or deleted person identifier of `admin`, this may be matched instead of your newly created person identifier `Admin`.

## Attribute Based Person Mappings

If the two systems being mapped have users with common attributes, such as their email addresses, then these attributes can be used to automate the person mapping. Here is an example person mapping configuration exercising this capability:

```
<?xml version="1.0" encoding="UTF-8"?>
<person-mappings xmlns="http://tasktop.com/xml/ns/sync/person-mapping-model">

  <repository
    url="http://jira.example.com"
    default-person-id="Admin" />
  <repository
    url="http://versionone.example.com"
    default-person-id="administrator" />

  <person-mapping>
    <person id="developer" />
    <person id="154J865A246" />
  </person-mapping>

  <field-person-mapping>
    <repository
      url="http://jira.example.com"
      mapping-field-id="person-email" />
    <repository
      url="http://versionone.example.com"
      mapping-field-id="Email" />
  </field-person-mapping>

</person-mappings>
```

In the above example a `field-person-mapping` has been added which tells Tasktop Sync to attempt to match persons based off the specified fields. Here we've chosen to map the email field from each repository. This will cause Tasktop Sync to use the person's email address to match the corresponding users in each system. If no such person can be found Sync will attempt to map the user by other means.

It may be the case that more than one user exists in the target system which matches the mapped field, for example there may be multiple users in a system with the same display name, or email address. If multiple results are found during person mapping the Synchronization will be placed in error. This can be resolved by adding direct person mappings for the duplicate users, the direct person mappings will be tried before the attribute based person mapping and if a result is found the attribute based person mapping will not be tried.

Multiple `field-person-mapping` elements can be configured, and the same repository can appear in multiple `field-person-mapping` elements with different `mapping-field-id` attributes configured. This allows a single source repository to be mapped to multiple target repositories using different person fields, for example:

```

<?xml version="1.0" encoding="UTF-8"?>
<person-mappings xmlns="http://tasktop.com/xml/ns/sync/person-mapping-model">

  <repository
    url="http://jira.example.com"
    default-person-id="Admin" />
  <repository
    url="http://versionone.example.com"
    default-person-id="administrator" />
  <repository
    url="http://tfs.example.com"
    default-person-id="DOMAIN/administrator" />

  <person-mapping>
    <person id="developer" />
    <person id="154J865A246" />
    <person id="DOMAIN/developer" />
  </person-mapping>

  <field-person-mapping>
    <repository
      url="http://jira.example.com"
      mapping-field-id="person-email" />
    <repository
      url="http://versionone.example.com"
      mapping-field-id="Email" />
  </field-person-mapping>

  <field-person-mapping>
    <repository
      url="http://versionone.example.com"
      mapping-field-id="Name" />
    <repository
      url="http://tfs.example.com"
      mapping-field-id="display-name" />
  </field-person-mapping>

</person-mappings>

```

In the above example users in VersionOne will be mapped with users in Jira using the person's email, and be mapped with users in Team Foundation Server using the display name. This configuration lacks a `field-person-mapping` between Jira and Team Foundation Server, as such no attribute based field mapping will occur between these two repositories.

## Scripted Person Mappings

If you need even more flexibility than a direct person to person mapping, you can write a scripted person mapper. This can be useful in situations where the difference in the mapped user follows an algorithm, such as `jsmith@email.com => jsmith`. You can also include much richer logic, such as a connection to a remote server to map the users.

Here is a basic example with two mapping scripts; one defined inside the person mapping configuration, the other defined as a reference to an external groovy file.

```

<person-mappings xmlns="http://tasktop.com/xml/ns/sync/person-mapping-model">

  <repository default-person-id="tasktop" url="https://rallydev.com"/>
  <repository default-person-id="tasktop@email.com" url="http://alm/qcbin"/>

  <!-- A direct person mapping -->
  <person-mapping>
    <person id="jsmith" />
    <person id="johnsmith@email.com" />
  </person-mapping>

```

```

<!-- Links a script to a pair of repositories -->
<scripted-person-mapping
  source-repository="https://rallydev.com"
  target-repository="http://alm/qcbin"
  person-script="convertToEmail"/>

<!-- Links a script to a pair of repositories -->
<scripted-person-mapping
  source-repository="http://alm/qcbin"
  target-repository="https://rallydev.com"
  person-script="removeEmail"/>

<!--An inline script -->
<person-script language="groovy" name="convertToEmail"><![CDATA[
  def map(value) {
    return value+"@email.com";
  }
]></scripted-person-mapper>

<!--An external script -->
<person-script language="groovy" name="removeEmail" script="scripts/removeEmail.groovy"/>

</person-mappings>

```

The `person-script` must contain one function, `map`, which receives a string parameter containing the source person id, and returns a string containing the target person id. If no target person id exists, then the function should return `null`, which will be mapped to the default person id.

The `person-script` is connected to a pair of repositories using the `scripted-person-mapping` node. That script will be executed when a synchronization is occurring from the `source-repository` URL to the `target-repository` URL.

The mapping script will only be executed if there is not a match in the direct `person-mapping` nodes, and no match is found by the attribute based person mapping (if configured). For example, using the script defined above, if the source person is "jsmith", this user exists within a `person-mapping` and therefore the "convertToEmail" `person-script` **will not** be run. If the source person is "bjohnson", there is no `person-mapping` that matches and therefore the `person-script` **will** be run (returning "bjohnson@email.com").

For more details on scripting in Tasktop Sync, see [Advanced Scripting](#)

## Person Mapping Priority Order

Sync exposes several mechanisms by which to map persons, these mappings are tried in sequence and the first mapping which yields a target result is used. If an error occurs during one mapping step the entire mapping is aborted and the Synchronization will be placed in error. The order in which the mappings will be applied is as follows:

- Direct person mapping
- Attribute based person mapping
- Scripted person mapping
- Default person mapping

If no result can be found after all mapping stages are tried the Synchronization will be placed in error.



# Appendix: Advanced Scripting

When the basic set of attribute mappings and casters will not satisfy your requirements, Tasktop Sync can be configured to run custom scripts. This gives the administrator a very powerful tool for solving complex problems, but does require a more thorough knowledge of both Tasktop Sync and software development. This Appendix is not meant as a developer guide, but instead offers useful suggestions to help an administrator comfortable with basic software development.

## Scripting Overview

Scripts in Tasktop Sync offer the administrator the ability to perform complex and custom data transformations during synchronizations. For example, if one system stores versions as a string, such as "Version 1-22", and the other as a decimal, such as "1.22", a script can be written to transform the values.

There are three types of scripts supported in Tasktop Sync, each performing a different type of transformation:

1. **Scripted Casters**: Convert the content of a source attribute into the desired content for the target attribute. For example, if the source attribute may contain line feeds, but the target attribute does not allow them, a caster script could be written to remove all line feed characters.
2. **Scripted Attribute Handlers**: A script is associated with a task attribute, and whenever that attribute value changes, the script is executed. These scripts can update one, or multiple attributes in the target task.
3. **Scripted Person Mappings**: A script to convert the person identifier in the source repository into the associated person identifier in the target repository.

Each script is associated with a Task Attribute or Person Mapping reference, and is triggered when the associated value changes. When triggered, the script is required to calculate the changes for the target task's attribute(s), which Sync will then apply to the target repository.

Scripts in Tasktop Sync can be written in one of two different script languages: `javascript` and `groovy`. `groovy` is the recommended language, and supports some features that `javascript` does not (as described in this appendix).

## The Script Context and Lifecycle

A Groovy script goes through a small lifecycle each time it is executed:

1. The script context is created.
2. The script is parsed using the specified language
3. The script is executed. This does not run the specified method, but instead defines the method(s) and declared variable(s) on the context.

4. If defined within the script, the method `setup()` is executed.
5. The primary method is executed. The method signature is unique to the script type; for example, the scripted casters method is.
6. If defined within the script, the method `teardown()` is executed.
7. The script context is destroyed.

A script's context contains all the methods and variables. During each execution, the context is created, the script runs through its lifecycle, and then the context is destroyed. Therefore, you cannot retain data between executions.

## Retaining a Groovy Script's Context between Executions

If you need to retain data stored on the context between executions of the script, this can be accomplished by specifying a subset of the groovy language, called `groovy/concurrent`.

```
<caster name="ConcurrentScript" language="groovy/concurrent" script="script.groovy"/>
```

This language offers the same Groovy language support as `groovy`, but its lifecycle is very different.

When the "Start" button is clicked in Tasktop Sync, the following is executed:

1. The script context is created.
2. The script is parsed using the specified language
3. The script is executed. This does not run the specified method, but instead defines the method(s) and declared variable(s) on the context.
4. If defined within the script, the method `setup()` is executed.

Any variables defined as "script binding" variables will be retained on the script

When it is time to execute the script during a Synchronization, the following is executed:

1. The primary method is executed. The method signature is unique to the script type; for scripted casters, the method is "cast(value)".

Once again, the "script binding" variables are retained after execution.

When the "Stop" button is clicked in Tasktop Sync, the following is executed:

1. If defined within the script, the method `teardown()` is executed.
2. The script context is destroyed.

A unique context is created for each point in the configuration that references a script. This means that if a single script is referenced twice in the configuration, then there will be two unique script contexts, and they will not share their "script binding" variables.

Please note that it is up to the script to handle concurrency issues. The same script context can be actively executed at the same moment by different threads, leading to unexpected results. If concurrent execution of your script is an issue, you will need to solve this using Groovy multithreading support. Please refer to Groovy language documentation for more details.

The following is an example of a `groovy/concurrent` script that reads a properties file once at startup. In this script, the properties file contains a property `TestValue`, which will be prepended to the attribute value during casting.

```
<caster name="SampleCaster" language="groovy/concurrent"><![CDATA[
    import java.io.FileReader;
    import java.util.Properties;
    import org.apache.log4j.Logger;

    properties = new Properties();
    logger = Logger.getLogger("my-groovy-script");

    def setup() {
        reader = new FileReader("/workspace/tasktop/my.properties");
        properties.load(reader);
        reader.close();
        logger.debug("Starting script with value "+properties.get("TestValue"));
    }

    def cast(value) {
        return properties.get("TestValue") + value;
    }

    def teardown() {
        logger.debug("Ending Script");
    }
}]></caster>
```

## Logging from Scripts

Scripts in Tasktop Sync can be difficult to debug, but Tasktop Sync provides logging facilities to help with this process. To add the logging facilities in a Groovy script, ensure you import the `log4j` logger class in your script:

```
import org.apache.log4j.Logger
```

Then, within your script you send messages to the logging facilities by retrieving a logger object and invoking function calls on it:

```
Logger logger = Logger.getLogger("my-groovy-script")
logger.error("My error message")
logger.debug("My debug message")
```

Tasktop Sync's logging facilities support `fatal`, `error`, `warn`, `info`, `debug`, and `trace` levels of logging messages.

## Expanding the Groovy Classpath

When writing groovy scripts, you may want to import classes from other java libraries. For instance, you may want to communicate with your own custom server in a scripted person mapping, and you have a `.jar` file that can help accomplish this. Access to these libraries can be accomplished by placing the `.jar` library files into a folder named "dropins" in the Tasktop Sync workspace.

Before proceeding, please make note of the following:

- Calling 3rd party Java libraries may have a significant performance impact on Tasktop Sync.
- 3rd party Java libraries can also cause synchronization failures, and even lead to Tasktop Sync crashing.
- When contacting Tasktop with questions or issues with Tasktop Sync, please specify if you have imported 3rd party libraries into your configuration.

You will need to relaunch Tasktop Sync for this to take effect. Once relaunched, the specified Java classes can then be imported into your scripts.

## Appendix: Known Limitations

### Linux Installer Limitations

Tasktop Sync does not properly support installation on Linux from or into a folder with spaces or special characters (e.g. `\ / { }`) in the name. The installer will work, but synchronization may not start, and uninstall and shortcuts will not work.

### Task Linking Limitations

- Tasktop Sync does not support automated back-linking.
- Repositories accessed via SDK connectors (connectors whose names do not have "Pre-Sync Version x" appended to the end) cannot be used to authenticate OSLC users (see [Task Linking Configuration](#) section for details on OSLC configuration).

### Tasktop Synchronization Limitations

Here are some known limitations of Tasktop Sync Task Synchronization. There are other vendor-specific limitations; check the [Connector Documentation](#) document for more information.

- Tasktop Sync reports an error "Could not find task in task list" when creating new proxy tasks, but the proxy task is still created and the synchronization completes successfully.
- The Confluence to HTML caster may lose newlines. Multi-line comments or descriptions in the confluence format will lose line breaks when transformed to HTML.
- The IBM Rational Team Concert SDK connector does not support attributes of type `internalTags`.

- Tasktop Sync does not map links in comments which refer to tasks originating in the other repository. Any comments from the foreign repository which refer to IDs in the foreign repository (e.g. "See bug 32") will remain unchanged, even if the ID in the current repository is completely different. In future releases of Tasktop Sync, the links will get properly mapped to the current IDs.
- Tasktop Sync does not synchronize deletions. Deleting a task from one repository will not delete it from the proxy repository.
- Tasktop Sync can only synchronize one task type (e.g. defect, requirement, work item, etc.) per task-mapping. If you want to synchronize more than one task type, then you need to have multiple task-mapping sections.
- There are a few limitations related to running as a Windows service. See the [Windows Service How-to](#) for details.

## Appendix: Troubleshooting

### Inspecting the Error Log

Tasktop Sync will log errors and warning to the Error Log. The Error Log can be selected by clicking on the "Tasktop Sync" drop-down menu in the Tasktop Sync toolbar and selecting "Open Error Log". Once opened the Error Log will appear in the lower pane of Tasktop Sync.

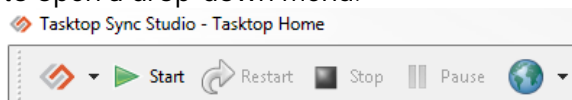
The Error Log shows real-time messages at a higher level than the console view. Messages shown in this view will be of higher importance than those in the console view and will typically indicate that there have been errors during synchronization. Each error will provide information such as a description of the error, when the error took place, and a stacktrace indicating what code was executing when the error was encountered.

### Generating an Error Report

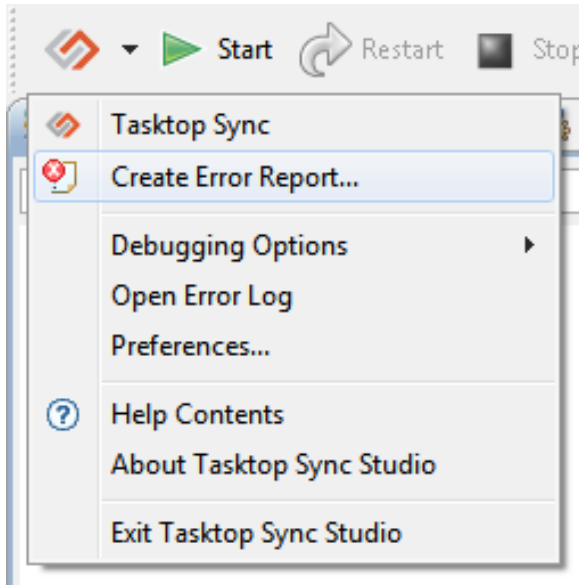
In the event of an error while using Tasktop Sync, Tasktop Sync can generate an error report in .zip format for sending to Tasktop support. To generate the report, follow these steps:

#### Tasktop Sync Studio

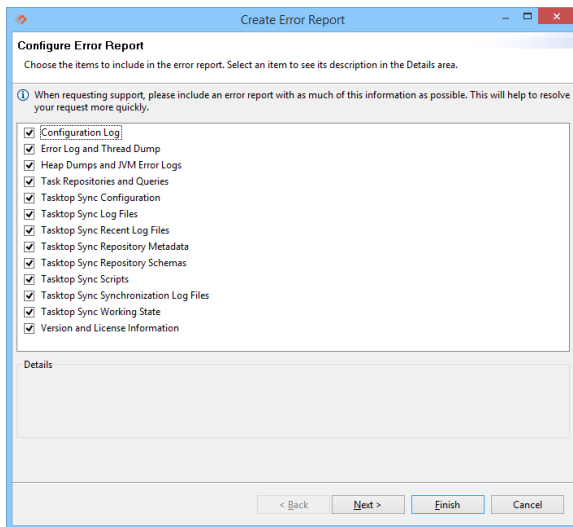
In the top left corner of the Tasktop Sync Studio UI, click on the arrow next to the Tasktop Sync logo to open a drop-down menu.



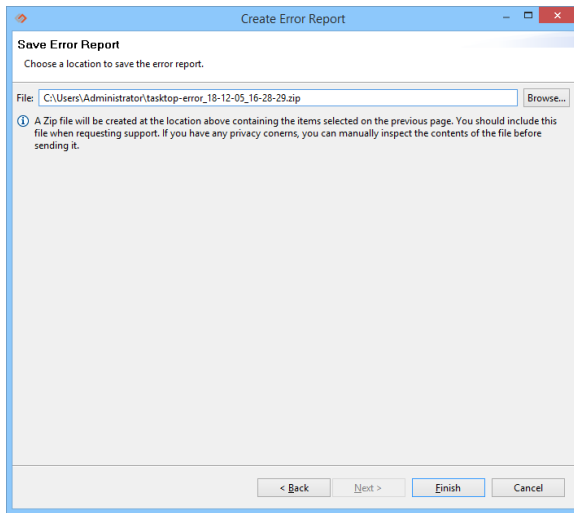
Select the "Create Error Report..." option from the menu that appears.



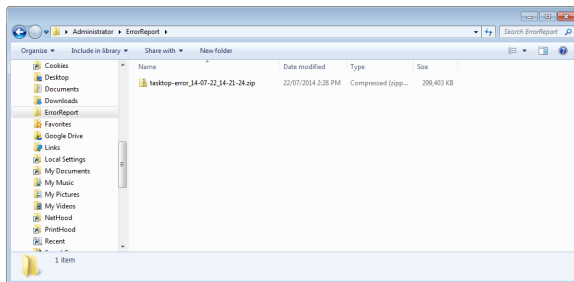
In the dialog that appears, select which components of the error report should be included. Each component can be selected, and a summary of the information included in that component will be shown. Once the desired components to include have been selected, click **Next**.



Choose a location to save the report. Click **Finish** to generate the error report.

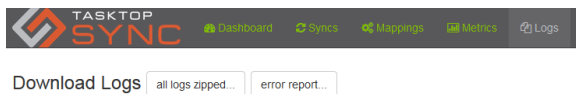


The error report can now be sent to Tasktop support as a .zip file, found in the aforementioned location.



## Tasktop Sync Web UI

On the Logs page of the Tasktop Sync Web UI, click the button labelled "error report..."



Wait for Tasktop Sync to generate the error report, this may take a few minutes. The error report will automatically download via the browser.

The error report can now be sent to Tasktop support as a .zip file, found in the downloaded location.

Error reports generated in the Tasktop Sync Web UI are not configurable, they exclude the Tasktop Sync database as well as all but the last seven days of synchronization logs.

## Disclaimer

Error reports may contain confidential information. Before handling or transmitting the error report, please ensure that these procedures are in compliance with your organization's policy regarding sensitive and confidential information.

## Accessing and Understanding the Tasktop Sync Logs

Tasktop Sync maintains detailed logs of all operations it performs while processing synchronizations. These logs are helpful in determining the cause of synchronization errors. However, because they are so detailed, they can sometimes be hard to understand. This section explains how to access these logs, and how to pick out relevant information.

There are three primary places to look at logging information:

1. The first source of logging information is the **Console** view. This log will show output in real time as Tasktop Sync is running; however, the default verbosity of the console view typically shows only high-level logging messages, and the console view is cleared each time Tasktop Sync is restarted. For this reason, the console view is useful for quickly determining if a problem has just occurred by looking through the backscroll of the view. Note that the verbosity of the **Console** view is limited by the verbosity of the global logging threshold.
2. The primary source of detailed logging information by task is stored in the **Sync Log** editor. See the [Sync Log](#) for details on opening this log viewer.
3. Finally, the on-disk log files contain the most detailed logging information. See the [Log File](#) for details on locating these log files. Note that these log files can be particularly difficult to read because they can contain interleaved messages as synchronizations are processed in parallel.

## Reading Tasktop Sync Logs

Each line of the synchronization log follows the same format:

```
[LogLevel] [Date Time] Message
```

For example, each synchronization log starts off by printing the version of Tasktop Sync used to complete the synchronization:

```
[DEBUG] [2012-09-13 14:59:34.901] Tasktop Sync Version: 4.0.0.qualifier
```

This is a DEBUG level log message that occurred on 2012-09-13 at time 14:59:34.901.

Each Tasktop Sync log also follows a consistent format for reporting details of the synchronization. This format is:

```
Tasktop Sync Version Information
RefreshSource
RefreshProxy
UpdateProxyState
UpdateSourceState
Done
```

These regions of the log files are delineated by log messages such as:

```
[DEBUG] [2012-09-13 14:59:35.787] Transitioning task using UpdateProxyState: https://rtc3.example.com:9443/ccm/resource/itemOid/com.ibm.team.workitem.WorkItem/_pnrAMP3tEeG0daazVVx5RA (5774)
```



This message shows which section of the log file is being entered, as well as the URL and task ID of the source task being processed.

The most interesting sections of the log are UpdateProxyState and UpdateSourceState. These sections detail determining which mappings should apply, as well as applying the relevant mappings. These sections also include the new values that were stored in the proxy task as a result of the synchronization process. Further, these sections print out each task attribute and its corresponding value before applying mappings and after applying mappings, which makes it easy to determine the initial and final state of the tasks retrieved from the repositories and submitted to the repositories.

Since UpdateProxyState and UpdateSourceState follow the same conventions, this section describes the logging output of only the UpdateProxyState.

The first two logging messages printed after entering the UpdateProxyState section are all the task attributes and their initial values for both tasks associated with the synchronization. These appear as:

```
[DEBUG] [2012-09-13 14:59:35.787] -begin source-
[DEBUG] [2012-09-13 14:59:35.787] SyncItem: https://rtc3.example.com:9443/ccm/resource/itemOid/com.ibm.team.workitem.WorkItem/\_pnvAMP3tEeG0daazVVx5RA
isIncoming : true
itemType : task
BgClosingDate:
archived:
category: _jnk04_nAEeCOcd6Rixv_mw
contextId: _jF9WI_nAEeCOcd6Rixv_mw
correctedEstimate:
creationDate: 1347573294249
creator:
...
[DEBUG] [2012-09-13 14:59:35.787] -end source-
```

and

```
[DEBUG] [2012-09-13 14:59:35.787] -begin proxy-
[DEBUG] [2012-09-13 14:59:35.787] SyncItem: http://hpalm.example.com:8080/qcbin;SYNCHRONIZER;RyansArea-DEF2076
isIncoming : false
itemType : BUG
BG_ACTUAL_FIX_TIME:
BG_ATTACHMENT:
...
[DEBUG] [2012-09-13 14:59:35.787] -end proxy-
```

This is followed by two more sections: "computing synchronization plan", and "applying synchronization plan". These sections are delineated by the messages:

```
[DEBUG] [2012-09-13 14:59:35.787] ** Computing synchronization plan (source -> proxy)
```

and

```
[DEBUG] [2012-09-13 14:59:35.804] ** Applying synchronization plan (source -> proxy)
```

Computing the synchronization plan is where Tasktop Sync determines which mappings to apply or skip; log messages in this section include detailed information about why a particular mapping is or is not applied. This is the section to consult if you are unsure why a particular mapping is not being applied.

Applying the synchronization plan is where Tasktop Sync applies the mappings from a task to its proxy task. This section contains detailed logging information about which attributes are being changed as part of the synchronization process and the new values that will be submitted to the repository.

Log messages outputted when computing the synchronization plan look like:

```
[DEBUG] [<Date> <Time>] [<Action>, <Reason>]: <MappingType> [sourceId=<sourceId>, sourceKey=<sourceKey>, targetId=<targetId>, targetKey=<targetKey>, caster=<caster>, targetAttributeStrategy=<strategy>]
```

This message describes which mapping is being tested. It is common for one of `sourceId` or `sourceKey` and `targetId` or `targetKey` to be null as mappings specify only one source or target per mapping. The `caster` attribute specifies which caster is being tested; by default, this is `CopyCaster`, which copies the value from source to target without modification. `targetAttributeStrategy` specifies which synchronization strategies were defined for this mapping.

The most interesting sections of this message are the `<Action>` and `<Reason>` sections. `<Action>` is either `Adding Mapping` or `Skipping Mapping`, depending on whether Tasktop Sync will apply or skip this particular mapping. The `<Reason>` section describes the reasons why this action will be taken. For example, here are some sample synchronization plan computation messages:

```
(1) [DEBUG] [2012-09-13 14:59:35.803] [Adding Mapping, No Conflict, Modified Attributes]:  
DefaultTaskAttributeMapping [sourceId=task.common.summary, sourceKey=null, targetId=task.  
common.summary, targetKey=null, caster=com.tasktop.internal.sync.core.caster.  
CopyCaster@119cb48, targetAttributeStrategy=CREATE]  
(2) [DEBUG] [2012-09-13 14:59:35.804] [Skipping Mapping, Source Attribute Not Changed]:  
DefaultTaskAttributeMapping [sourceId=null, sourceKey=description, targetId=null,  
targetKey=BG_DESCRIPTION, caster=com.tasktop.sync.core.caster.  
HtmlCopyCaster@4470e488, targetAttributeStrategy=CREATE]  
(3) [DEBUG] [2012-09-13 14:59:35.804] [Skipping Mapping, No Matched Attributes]:  
DefaultTaskAttributeMapping [sourceId=null, sourceKey=BG_RESPONSIBLE, targetId=null,  
targetKey=BG_RESPONSIBLE, caster=com.tasktop.sync.core.caster.PersonCaster@4fabbfd2,  
targetAttributeStrategy=CREATE]  
(4) [DEBUG] [2012-09-13 14:59:35.804] [Skipping Mapping, No Matched Attributes]:  
DefaultTaskAttributeMapping [sourceId=null, sourceKey=task.common.key, targetId=null,  
targetKey=BG_USER_01, caster=com.tasktop.internal.sync.core.caster.CopyCaster@14be9cdb,  
targetAttributeStrategy=CREATE]
```

Line 1 shows a mapping that Tasktop Sync will apply. The reason that this mapping applies is because Tasktop Sync has detected that applying it results in no conflicts, and that applying this mapping results in modified attributes (`No Conflict, Modified Attributes`). Note that both the source and target attributes of this mapping are `task.common.summary`.

Line 2 shows a mapping that will be skipped (Skipping Mapping) because the source attribute of the mapping (description) has not changed (Source Attribute Not Changed).

In this particular example, only one mapping will be applied, so the applying synchronization plan section is relatively straightforward. These messages look like:

```
[DEBUG] [<Date> <Time>] <TargetTask>#<AttributeName> <- <ValueList>
```

This message describes the target task and attribute which is being changed, and the values that are being stored in this attribute as a result of applying the mapping. Following our above example, the logging message for this synchronization would be:

```
[DEBUG] [2012-09-13 14:59:35.804] http://hpalm.exmaple.com:8080/qcbin;SYNCHRONIZER;ProjectArea#BUG::2076  
(DEF2076)##BG_SUMMARY <- [[New Summary]]
```

This message shows that the BG\_SUMMARY attribute on DEF2076 is getting a new value: "New Summary".

Finally, the last messages of the UpdateSourceState section will print out the values of the task attributes that are sent to the repositories as a result of the synchronization, and will print them out in the same format that the initial state of the tasks were logged.

## Appendix: Tasktop Sync Error Codes

The following sections explain the error codes used by Tasktop Sync and its installers.

### Tasktop Sync Installer for Windows

Error Code	Description	Troubleshooting Tips
TTSS W100 01	An older version of Tasktop Sync is already installed on this computer.	Uninstall any existing installations of Tasktop Sync and restart the installer.
TTSS W100 02	Tasktop Sync is running.	Close any open instances of Tasktop Sync and restart the installer.
TTSS W200 01	The specified TFS client repository directory does not exist.	Make sure that the TFS repository path passed in to the installer is spelled correctly and points to an existing directory.
TTSS W200 02	The specified TFS client repository directory does not appear to be a valid TFS repository.	Make sure that the TFS repository path passed in to the installer points to a directory where a TFS repository zip file was unzipped.

TTSS W200 03	There was a problem installing TFS into Tasktop Sync. See error logs for more details.	Please see the install.log file in the target directory for more information.
TTSS W300 01	The specified Rational Team Concert client repository directory does not exist.	Make sure that the Rational Team Concert repository path passed in to the installer is spelled correctly and points to an existing directory.
TTSS W300 02	The specified Rational Team Concert client repository directory does not appear to be a valid Rational Team Concert repository.	Make sure that the Rational Team Concert repository path passed in to the installer points to a directory where a Rational Team Concert repository zip file was unzipped.
TTSS W300 03	There was a problem installing Rational Team Concert into Tasktop Sync. See error logs for more details.	Please see the install.log file in the target directory for more information.

## Tasktop Sync Installer for Linux

Error Code	Description	Troubleshooting Tips
TTSSL 10001	This computer is not running a Linux operating system.	Run the installer on a Linux OS.
TTSSL 10002	The installer was not run with a 64-bit Java JRE on a Linux operating system.	Make sure a 64-bit Java JRE is installed on your Linux OS, and use that 64-bit JRE to run the installer jar file. ( <a href="#">Read more here.</a> )
TTSSL 10003	A version of the product is already installed.	Uninstall the existing version before installing a new one.
TTSSL 20001	The specified TFS client repository directory does not exist.	Make sure that the TFS repository path passed in to the installer is spelled correctly and points to an existing directory.
TTSSL 20002	The specified TFS client repository directory does not appear to be a valid TFS repository.	Make sure that the TFS repository path passed in to the installer points to a directory where a TFS repository zip file was unzipped.
TTSSL 20003	The specified TFS client repository path is a relative path. Only absolute paths are accepted.	Make sure that the TFS repository path passed in to the installer is an absolute path, and restart the installer.
TTSSL 30001	The specified Rational Team Concert client repository directory does not exist.	Make sure that the Rational Team Concert repository path passed in to the installer is spelled correctly and points to an existing directory.

TTSSL 30002	The specified Rational Team Concert client repository directory does not appear to be a valid Rational Team Concert repository.	Make sure that the Rational Team Concert repository path passed in to the installer points to a directory where a Rational Team Concert repository zip file was unzipped.
TTSSL 30003	The specified Rational Team Concert client repository path is a relative path. Only absolute paths are accepted.	Make sure that the Rational Team Concert repository path passed in to the installer is an absolute path, and restart the installer.

## Additional Information

This section provides additional information about specific error codes.

### TTSSL10002

The installer is available as a self-extracting .bin file which includes a 64-bit JRE. If you run the .bin file, it will use that bundled 64-bit JRE to run the installer.

## Appendix: Notices

### Copyright

The material in this guide is Copyright © 2012, 2013, 2014, 2015, 2016, 2017, 2018, 2019 Tasktop Technologies.

### Patents

Covered by one or more of the following: US Patent No. 9,459,839 and US Patent No. 9,342,512.

### Trademark Acknowledgements

See [www.ibm.com/legal/copytrade.shtml](http://www.ibm.com/legal/copytrade.shtml).

## After Completing the Wizard

# Installation Guide

## Preparation

### System Requirements

The minimum server configuration needed to run Tasktop Sync is:

- 4 GB system memory
- 1 GHz processor, 2 cores
- 30 GB free disk space
- A supported operating system
  - 32-bit or 64-bit Windows; Windows Server 2007, 2008, 2012, 2016 or 2019 are recommended.
  - Red Hat Enterprise Linux 5, 6, 7 & 8; GTK3 and X Windows are required.
    - To force GTK3, add the following to the TasktopSync.ini file before `-vmargs:`  
`--launcher.GTK_version`  
`3`
  - SUSE Linux Enterprise Server 11, 12, 15; GTK3 and X Windows are required.
    - To force GTK3, add the following to the TasktopSync.ini file before `-vmargs:`  
`--launcher.GTK_version`  
`3`

### Accounts

Before installing Tasktop Sync, ensure that login credentials for the following user accounts are available:

- On the Tasktop Sync installation server:
  - An account with administrative privileges that will be used to configure Tasktop Sync
- On each repository to be accessed by Tasktop Sync:
  - An account that Tasktop Sync will use to create and update tasks
  - An account that Tasktop Sync will use to append logging comments when required

### Required Files

Before installing Tasktop Sync you will need the following files:

- The Tasktop Sync installation executable
- A Tasktop Sync license file (**tasktop.license**)
- Any vendor-specific downloads specified in the [Certified Connectors](#) document

## Backup Installation Customizations

Installing Tasktop Sync will overwrite the following customizations you may have made to a previous install:

- TasktopSync.ini: Ensure any changes made to this file are backed up so they can be applied again after installing a new version of Tasktop Sync.
- Windows service log on credentials: If there is an existing installation of Tasktop Sync as a Windows service, and the service is set up to run under a specific account, ensure the account name and password are available to be entered again after upgrading to a new version.

Note that the working directory containing Tasktop Sync's configuration files (task mappings, repository settings, cache data, etc.) will **not** be overwritten by an install. Of course, it's a good idea to back these up on a regular basis anyway.

## Vendor-specific Requirements

Consult the **Installation requirements** section for all vendors you wish to use with Tasktop Sync in [Connector Documentation](#). These sections will identify additional information you will need during installation of Tasktop Sync.

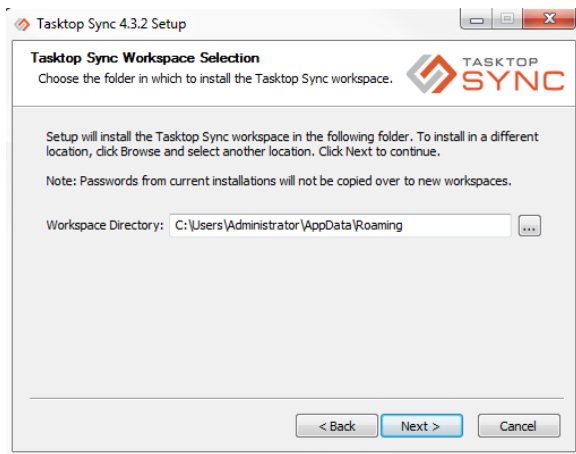
## Test Environment

We **strongly** recommend setting up a test environment, using snapshots of both repositories. Getting the mapping between the task attributes on both systems is tricky, and you are highly likely to change your mind about the configuration before you have finished. It is better to change your mind on a test environment than a production environment.

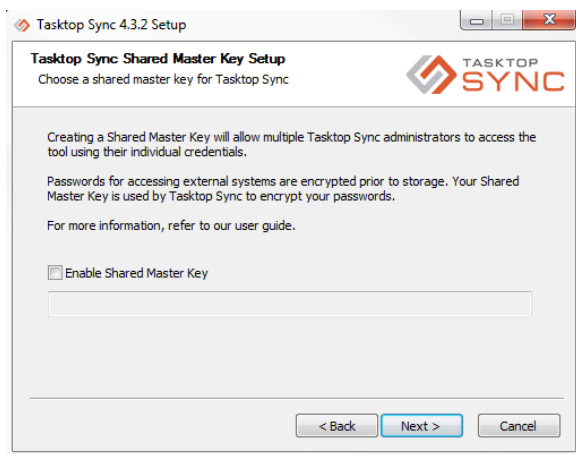
In your test environment, you probably want to turn off email notifications on your task repositories, so that you do not bury people with notifications as you test the synchronizer.

## Installation on Windows

You will be provided with an installation package for Tasktop Sync as a standard Windows installer. The installer will provide instructions for how to install Tasktop Sync as server deployment and service deployment. There are several pages in the installation dialog which you do need to pay attention to. The first is the workspace selection page:

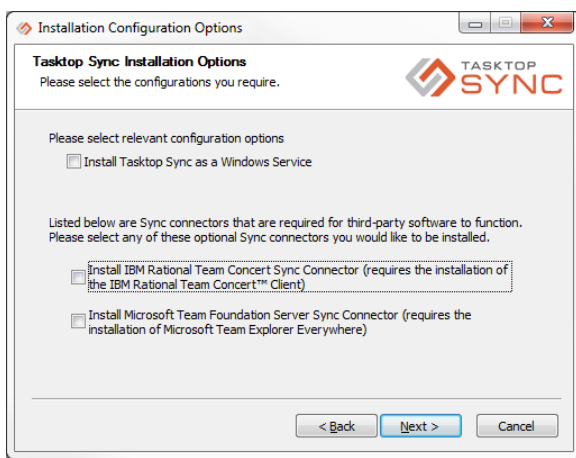


This option will allow you to select the location of the Tasktop Sync workspace. By default, the workspace is placed within the installing user's AppData directory.



This page allows you to enable and specify a shared master key for Tasktop Sync. Enabling this feature will allow multiple Windows user accounts to administer Tasktop Sync.

For more information on this feature, please refer to the [Shared Master Key](#) section.



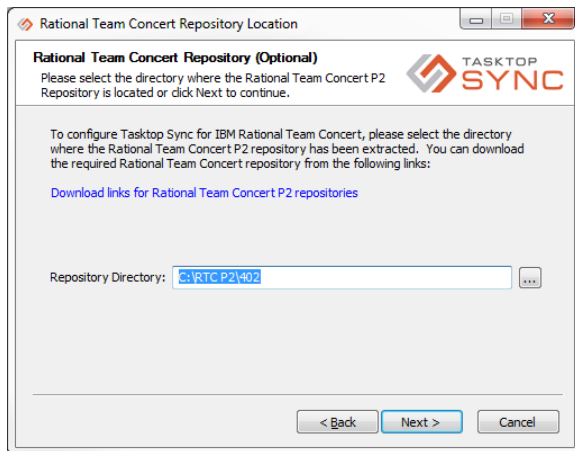
If you want to run as a Windows service, you need to check the top box. If you are evaluating Tasktop Sync or only wish to use the Quick Start wizard we recommend not running as a Windows service as



there are a few less steps involved to access the user interface. Further details on installing Tasktop Sync as a service are available in the [Installing Sync as a Windows Service](#) guide.

If you are using IBM Rational Team Concert or Microsoft Team Foundation Server, you also need to check the appropriate boxes here. If you do not check the boxes now and later decide to use one of these repositories, you will have to reinstall Tasktop Sync to install the appropriate connectors.

If you check any of the boxes, the next page will contain links to download the plugin for the repository you select (or the first repository you selected):



Click on the link for your repository to download it (if you have not done so previously), and then specify the directory where to find it. Alternatively, you may run the installer from the Windows command prompt and pass in the repository location through a parameter. The accepted command line parameters are:

- **-RTC** for the IBM Rational Team Concert connector
- **-TFS** for the Microsoft Team Foundation Server connector

For example, running **TasktopSetup-sync.exe -RTC="C:\Downloads\RTC-Client-p2Repo-4.0\repo"** will tell the installer to automatically install the IBM Rational Team Concert connector with Tasktop Sync, using the given repository directory.

**NOTE: Getting the repository version exactly correct is important.** For example, selecting 4.0.3 if you have a 4.0.1 repository will not be productive, and you will not be able to change it without reinstalling Tasktop Sync.

## Shared Master Key

The Shared Master Key may be enabled during installation and is used to encrypt passwords in Tasktop Sync. Instead of user specific encryption, the Shared Master Key will be used as a master password for all Tasktop Sync administrators.

The Shared Master Key is stored in plain text format within the workspace. Upon installation, only Windows Administrators are granted access to the key file. It is important to ensure that appropriate Windows access permissions are placed on the Tasktop Sync workspace.

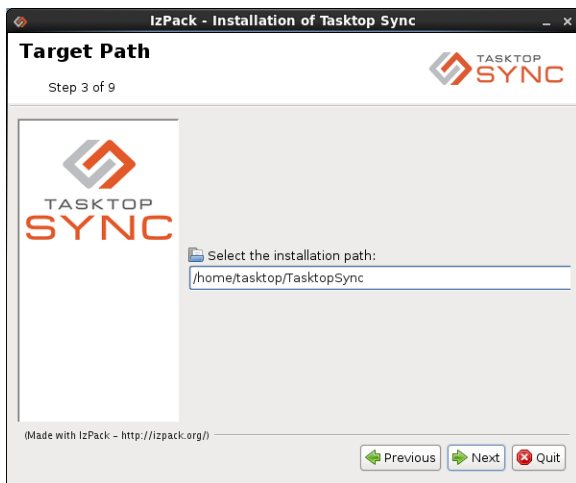
## Installation on Linux

**NOTE:** In the following instructions the installer file is referred to as `TasktopSetup-linux.bin` for convenience. The actual file distributed by Tasktop will be called something like `TasktopSetup-sync-lin64-3.5.0.20140320-0607-RELEASE.bin`.

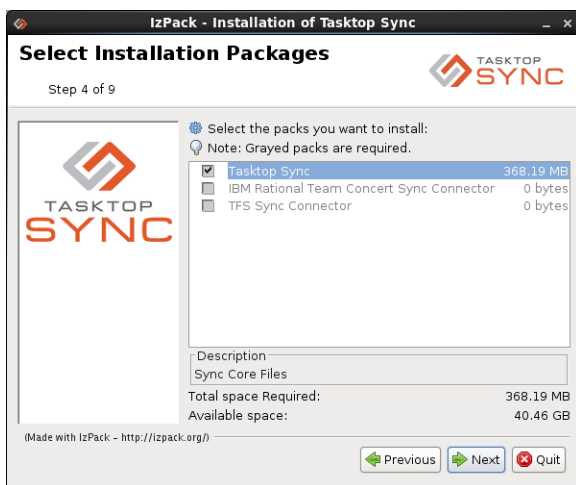
1. Open a terminal and change to a directory that the currently logged in user has write access to by typing "`cd <path to directory>`" in the terminal.
2. Ensure that the installer .bin file is executable by typing "`chmod u+x <path to installer>`" in the terminal.
3. Run the installer:
  - a. (Optional) Installation with IBM Rational Team Concert:
    - i. Ensure that there is an IBM Rational Team Concert client p2 repository zip file available on the computer. If not, you can download the Rational Team Concert 4.0 repository from <https://jazz.net/downloads/rational-team-concert/releases/4.0/RTC-Client-p2Repo-4.0.zip>. Note that this zip file must be unzipped to a directory on the local machine.
    - ii. Use the optional command line argument "`-DRTC`", passing in the IBM Rational Team Concert p2 repository directory, when running the installer as described in step 3.3 (e.g. `TasktopSetup-linux.bin -DRTC="/media/sf_shared/RTC-Client-p2Repo-4.0"`).
  - b. (Optional) Installation with Microsoft Team Foundation Server:
    - i. Ensure that there is a Microsoft Team Foundation Server client repository available on the computer. If not, you can download one from <http://download.microsoft.com/download/F/0/4/F04E054B-9DB5-4C24-AF84-DF1A290F5C73/TFSEclipsePlugin-UpdateSiteArchive-12.0.1.zip>. Note that this file must be unzipped to a directory on the local machine.
    - ii. Use the optional command line argument "`-DTFS`", passing in the Microsoft Team Foundation Server p2 repository directory, when running the installer as described in step 3.3 (e.g. `./TasktopSetup-linux.bin -DTFS="/media/sf_shared/TFS-Client-p2Repo-10.0"`).
  - c. Type the path to the installer (e.g. `./TasktopSetup-linux.bin <optional args>`) in the terminal and press enter to launch the Tasktop Sync installer.



4. Click Next.
5. Review the license agreement.
6. Specify the target directory.



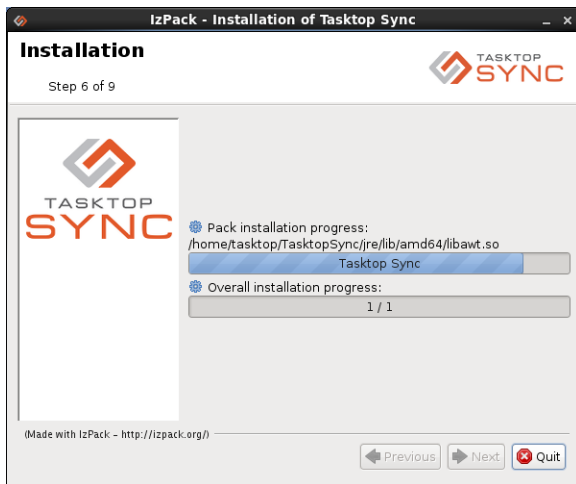
7. Confirm packages to install.



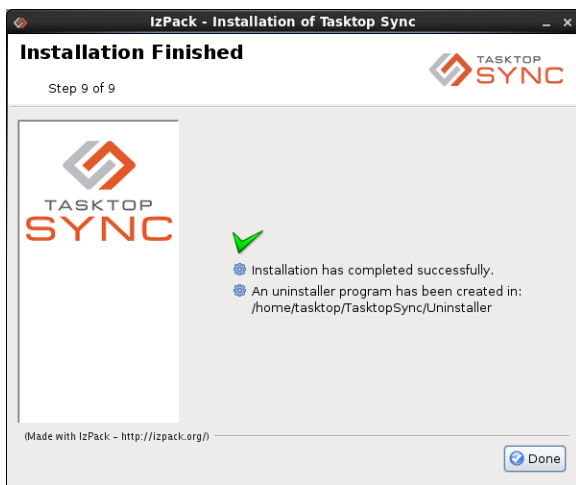
8. Specify shortcuts to be created.



9. Wait for installation to finish.



10. Confirm that the installation was completed successfully.



See also [Linux Installer Limitations](#) .

# Post-Installation Steps

Ensure that the following installation customizations you may have made to a previous install of Tasktop Sync are applied:

- **TasktopSync.ini** -; Edit this file to apply any custom settings.
- Windows service log on credentials -; If you are running Tasktop Sync as a Windows service, make sure that the login details used from a previous installation are entered again.

## Starting Tasktop Sync

The method for starting Tasktop Sync is slightly different depending on whether Tasktop Sync was installed as a Windows Service or not.

## Standard Installs

In a desktop installation, you can bring up the Tasktop Sync Dashboard by right-clicking on the Tasktop Sync icon in your system tray and selecting "Open Tasktop Sync." If this icon isn't visible, it means that Tasktop Sync isn't running, and you can instead start it from the start menu.

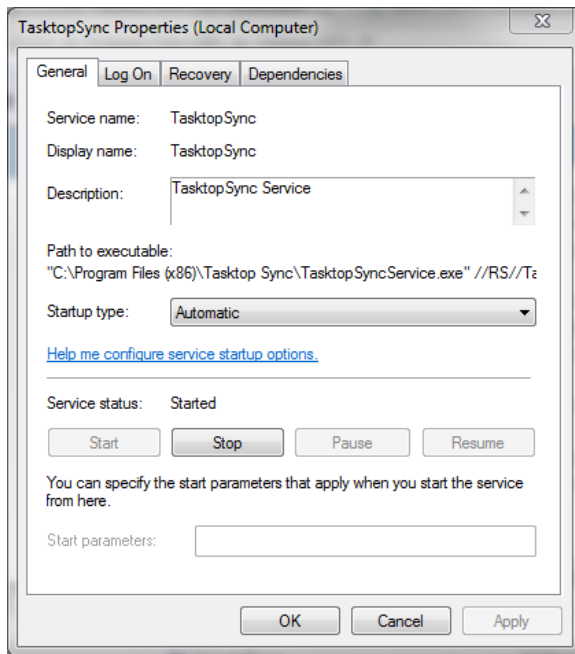
## Windows Service Installs

See the [Windows Service How-to](#) document.

### Starting and Stopping the Service

Starting and stopping the Tasktop Sync service can be done using the Tasktop Sync Service Manager installed as part of the installation process. Because interacting with Windows services requires administrative permissions, you will be asked for an administrator password when running this application.

After running this program you will be presented with a dialog:



This dialog contains buttons allowing you to start and stop the service.

## Initial Setup and License Activation

The first time you run Tasktop Sync, you will need to do two things. The first is to initialize the secure storage that Tasktop Sync uses to store configured repository passwords. You will be prompted to create a master password and recovery questions for accessing secure storage the first time Tasktop Sync is run.

The second is to import the license file you have been provided (e.g. TasktopSync.license). To import your license file, look for the Tasktop logo in the top left of your Tasktop Sync window. Select the down-arrow drop-down menu beside this icon and select "Preferences..." This will pop-up a dialog window which allows you to import your license file. Click the "Import from File..." button and select the license file you extracted from your installation package. This should populate the License dialog box with the details of your particular license. Finish the import by clicking the "OK" button.

# Installing Sync as a Windows Service

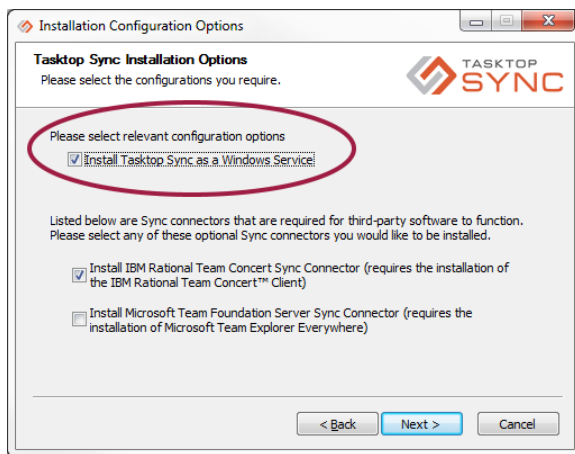


This document describes how to install and configure Tasktop Sync as a Windows service.

## Installation

Detailed instructions on system requirements, license activation and installation preparation are available in the [Installation Guide](#). Note that you will need to complete the steps on [user setup](#) before you can import the license file.

To install as a Windows service, make sure to check the option for it provided in the installation options page of the wizard.



## User Setup

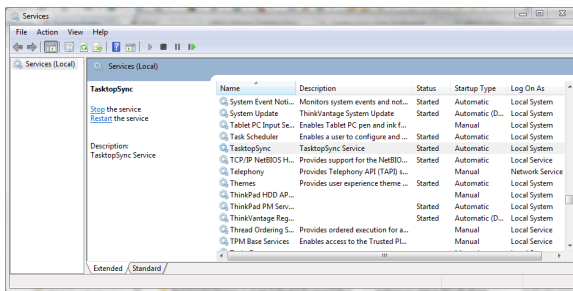
When installed as a Windows service, Tasktop Sync should not run as the Local System account and should be run as a specific user account with the following permissions:

- permission to run a service
- read and write permission in Tasktop Sync's workspace directory and all its sub-directories (default location: C:\Documents and Settings\All Users\Application Data\TasktopSync)
- read permission on the install directory and all its sub-directories (default location: C:\Program Files\Tasktop Sync)
- write permission on just the top level install directory

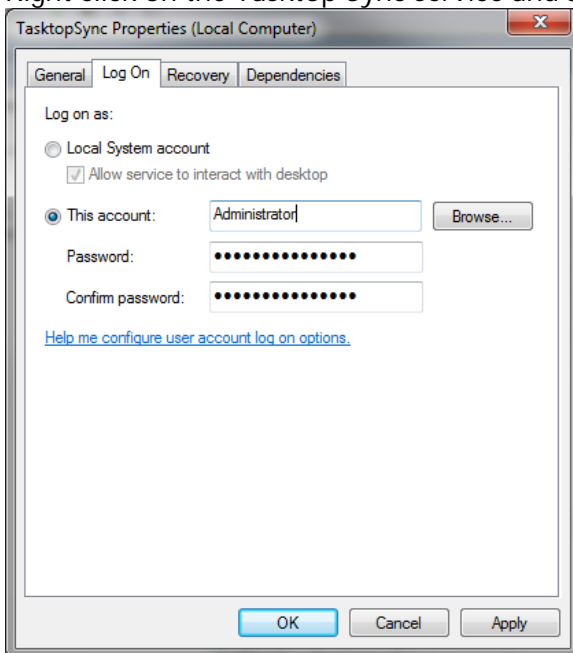
If that is not possible, consult the [Known Limitations](#) below for a workaround.

To configure Tasktop Sync to use the Administrator user account:

- Open the Windows Services application by going to Administrative Tools -> Services.
- Scroll down to the Tasktop Sync Service as shown:



- Right click on the Tasktop Sync service and select properties, you should see the following dialog:



- Select the "This account" radio button and enter in the Administrator account and the password.
- Select "OK".
- Navigate to the directory where Tasktop Sync has been installed (typically @C:\Program Files (x86)\Tasktop Sync@).
- Right-click on the @TasktopSync.exe@ file and select Properties.
- Under the Compatibility tab, click the "Change Settings for all users" button.
- Select the "Run this program as an administrator" radio button under the Privilege Level heading and click "Apply".

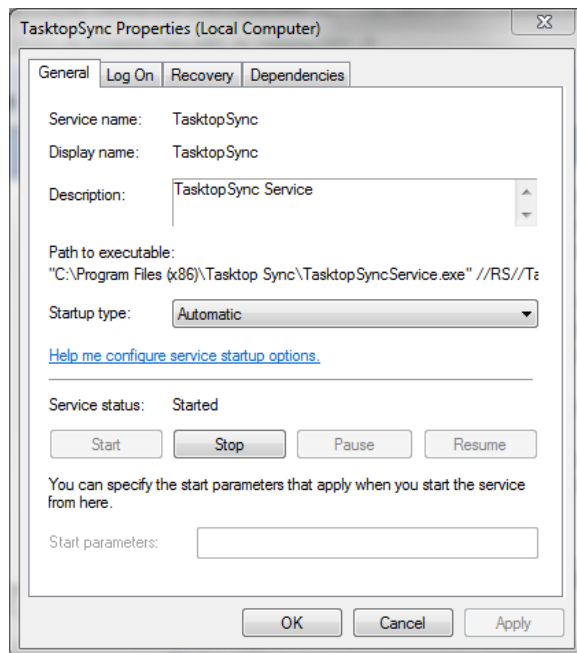
Note: You must use the Windows Service application to configure the services user account (you cannot use the Tasktop Sync Service Manager). This known limitation is described in the [Known Limitations](#).

## Task Mapping Configuration and Maintenance



To access the Tasktop Sync user interface, you must login as the user account used to run the Tasktop Sync service.

Before you can access the UI you must shutdown the Sync service if it is running. To check whether the service is running and shut it down, run the Tasktop Service Manager application (from the Windows start menu) installed as part of the installation process. You will see the following dialog:



If the service status is started, you should shutdown the service by clicking the "Stop" button. Once the service has been stopped, access the Sync user interface by running the Tasktop Sync application installed as part of the installation process (typically @C:\Program Files (x86)\Tasktop Sync\Tasktop Sync.exe@). If after running the Sync application you receive an error that another process is using the workspace, this indicates that the service has not been shutdown or has not finished shutting down.

For further details on using the Tasktop Sync UI, please consult the user guide.

Once the task mapping configuration has been completed and Sync has been verified as working you can resume running it as a service in the background. To do this, exit the Tasktop Sync UI by closing the window or selecting "Exit" from the Tasktop menu.

You may now use the Tasktop Sync Service Manager to restart Sync in the background by clicking "Start."

## Data Directory Location

Tasktop Sync stores log files and other information in a data directory. The location of this directory varies depending on which operating system Sync has been installed on.

- On Windows 2008 Server, Windows Vista, Windows 7: \$TASKTOPSYNC\_HOME = @C:\ProgramData\TasktopSync@

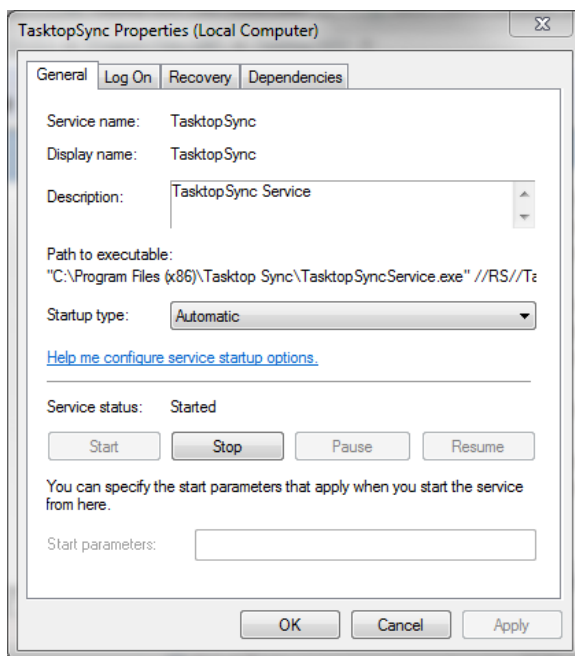
- On Windows XP, Windows 2003 Server: \$TASKTOPSYNC\_HOME = @C:\Documents and Settings\All Users\Application Data\TasktopSync@

Note that these directories are different than the directories used by default if Sync is not installed as a service. See [migrating](#) for details.

## Starting and Stopping the Service

Starting and stopping the Sync service can be done using the Tasktop Sync Service Manager installed as part of the installation process. You will be asked for an administrator password when running this application because interacting with Windows services requires administrative permissions.

After running this program you will be presented with a dialog:



This dialog contains buttons allowing you to start and stop the service.

## Migrating From an Application Only Install

If you have previously installed and configured Tasktop Sync as an application only (i.e. not as a Windows service), it's likely that your configuration folder is in a different location than where the service is expecting. To migrate your configuration to a service install:

- Copy your existing configuration files to the data directory as specified [above](#) . (e.g. copy @C:\Users\<User>\AppData\Roaming\TasktopSync\workspace@ to @C:\ProgramData\TasktopSync\workspace@.
- If you'll be running Tasktop Sync as a service under a different user account than you were previously then you'll need to launch the user interface to enter your passwords again.

## Known Limitations

### General

#### **The Tasktop Sync user interface does not start and stop the Windows service.**

The Tasktop Sync user interface is being generated by the service itself, so the service must be started and stopped outside of the Tasktop Sync user interface.

**Work Around:** See the section on [starting and stopping](#)

#### **Tasktop Sync's user interface is not accessible if the service is already started.**

The Tasktop Sync configuration files can only be accessed by one process at a time so it's not possible to access the user interface from the application while the service is running.

**Work Around:** The service must be stopped before starting the application. See the section on [starting and stopping](#) the service.

#### **Setting the log on account in the Tasktop Sync Service Manager does not work.**

The service manager will not save changes made to the log on account.

**Work Around:** Open the Windows Services application by going to Administrative Tools -> Services. Scroll down to the Tasktop Sync Service, right click on it, and select properties. Enter account credentials in the Log On tab.

#### **Credentials in secure storage are not shared when switching the user account Tasktop Sync is being run under.**

The secure storage used by Tasktop Sync is tied to the user who is running Tasktop Sync. If Tasktop Sync is configured when running as an one user, then switched to run as another user, it will be using a new secure storage and none of the previously entered passwords will be available.

**Work Around:** When changing user accounts, start the user interface and ensure that all repository passwords are entered again and saved.

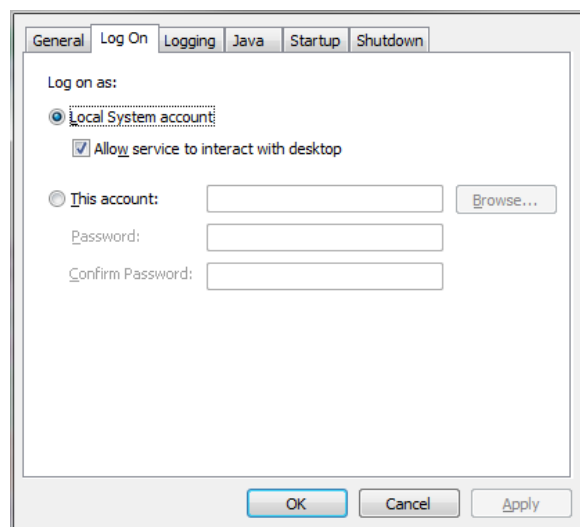
### **When Running as a Local System User**

It's strongly recommended that Tasktop Sync be run using a standard account and not the Local System user (see [User Setup](#) ). If this is not possible for some reason there are a number of limitations that will be encountered:

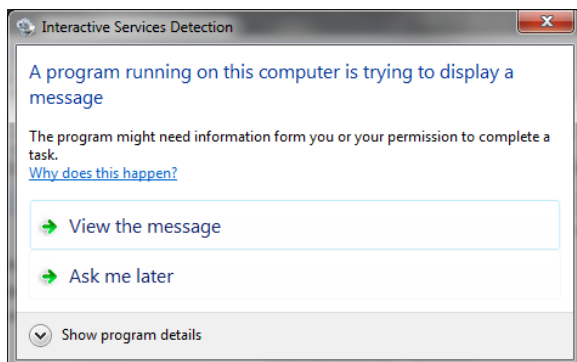
## Running the service as a Local System user prevents accessing the user interface when running as an application.

Windows enforces file system permissions such that files accessible by services run as the Local System user are not accessible by programs run as applications. In some cases Windows may seem to allow access to the same file at the same location in the file system, but in fact the files can be different for the service and the application. This can happen even if both users have Administrator privileges. The result of this is that Tasktop Sync when run as an application cannot access the configuration files when run as a service under the Local System account.

**Work Around:** To access Tasktop Sync's user interface when it's running as a service under the Local System account you need to enable the service to interact with the desktop. This can be done in the "Log On" section of the service manager. Open the service properties (right-click on the service in the Windows service list and select properties), select the "Log On" tab, and check "Allow service to interact with the desktop".



Starting the service at this point will pop up a dialog asking for permission to access the service's interface. Click "View the message" to access the Tasktop Sync user interface.



## When running as the Local System user the Tasktop Sync user interface is not visible in screen sharing sessions.

The interactive desktop feature of Windows services does not allow its window to be visible across remote screen sharing sessions.

**Work Around:** Screen sharing will work in conjunction with an RDP session. Instead of running the conferencing software on the same machine as Tasktop Sync, run it on a different machine and RDP into the Tasktop Sync machine. The screen will be visible over the RDP session and the RDP session can be screen shared over the conferencing software.

## Tasktop Sync's user interface is not accessible if the service is already started using the Local System user.

Once the service has been started it's not possible to force the user interface to activate when running as a Local System user.

**Work Around:** Ensure the service has [permission to interact with the desktop](#) , then [restart](#) the service.

## In RDP sessions the Tasktop Sync user interface is only available on the "console session" of a Windows server.

When logging into a Windows server using RDP, even as the Administrative user, the RDP login is not the console session on the server. This can cause problems when attempting to configure Tasktop Sync when it is installed as a Windows service. The Tasktop Sync dashboard is only available on the console session of a Windows server when it has been installed as a service. The symptom of this problem is that Tasktop Sync dashboard does not appear after selecting "Allow service to interact with the desktop" in the Tasktop Sync Service Manager and restarting the service.

**Work Around:** Instead of using the standard Remote Desktop Connection client on your workstation, open up a command prompt and use the following command:

```
mstsc /admin /v:tasktopsync.example.com
```

You should replace **tasktopsync.example.com** with the hostname of the computer running Tasktop Sync in the above command. This command will open a remote desktop session to the Windows server that uses the console session which allows Tasktop Sync configuration to function normally.