Triggering Synchronization with an Internal Scheduler

Teamwork Server has a built-in mechanism to trigger tasks at various periods and with various parameters.

This mechanism can be used for triggering synchronization between servers.

The tasks are specified by *.properties files located in the schedule subdirectory in the Teamwork Server install directory. Each *.properties file describes one task to be run at the defined period(s). There is an example file (synchronization.properties.example) which can be examined and copied/renamed to produce the synchronization task(s). For example, you can create two files (syncDE_GB.properties and syncDE_US.properties) to describe two synchronizations, one for synchronization from a remote server on the GB site and another for synchronization from the remote server on the US site.

Teamwork Server constantly monitors the schedule subdirectory for any changes in the *.properties files. Any changes take effect immediately. All other files with different endings, such as the synchronization.properties.example outlined above, are ignored.

When scheduled tasks are run, they are run on behalf of the system user of the Teamwork Server. This is the reason it is not necessary to specify the user on the target server when running synchronization using the scheduler; however, you must specify the user on the target server when running synchronization from the command line utility.

Properties files are specified in Java property file format.

Properties files carry the following task parameters:

- **com.nomagic.teamwork.synchronization.Task.** This parameter describes the type of task to be run. Currently, the synchronization task is the only type of task that can be scheduled. This task type is specified by the com.nomagic.teamwork.synchronization.SynchronizationJob value of the property.
- **com.nomagic.teamwork.schedule.cron.** This parameter specifies the frequency of task running. The parameter syntax corresponds to the Unix cron task definition syntax. This syntax allows very flexible descriptions of task running frequency and pattern. For example, the value: "0 30 01 ? * MON-FRI" triggers a synchronization task at 01:30 AM Monday to Friday. For more information about this parameter, please see the documentation of Quartz framework at http://quartz-scheduler.org/files/documentation/Quartz-2.1.x-Documentation.pdf. See the chapter "CronTrigger Tutorial." Alternatively, you can read the manual of your Unix system.
- **com.nomagic.teamwork.synchronization.SourceServer.** This parameter specifies the synchronization source server and port. It is a mandatory parameter for a synchronization task.
- **com.nomagic.teamwork.synchronization.SourceUser.** This parameter specifies the user on the synchronization source server (the server providing the data) used to access and fetch the project data.
- **com.nomagic.teamwork.synchronization.SourcePassword.** This parameter describes the password of the user on the synchronization source server used to access and fetch the project data. Since the password is stored in plain text, please ensure that the *.properties file has the correct and safe permissions.
- **com.nomagic.teamwork.synchronization.Use Secure Connection.** This parameter specifies that SSL connection shall be used between the target server and source server for data transfer. SSL certificates must be set up correctly on both the source and target servers. For more information, see Secured Connection tab.
- **com.nomagic.teamwork.synchronization.CategoryPrefix.** This parameter specifies the prefix prepended to each remote category name in the target (this) server. For example, if the category name in the source server is “Requirements,” and the prefix is “REMOTE/,” in the target server this category will be named “REMOTE/Requirements.”

Once triggered by the scheduler, synchronization runs to completion (or to unrecoverable error). The progress of synchronization can be monitored in the server log file. There is no facility to stop these jobs by user command (except for full server stop). If the synchronization job takes longer than the schedule period, and the next scheduled event happens before the previous job has finished, the event is suppressed. The second synchronization is NOT started in parallel; the running synchronization is allowed to continue and finish normally.