Skytap Automation Pack for IBM Rational Team Concert

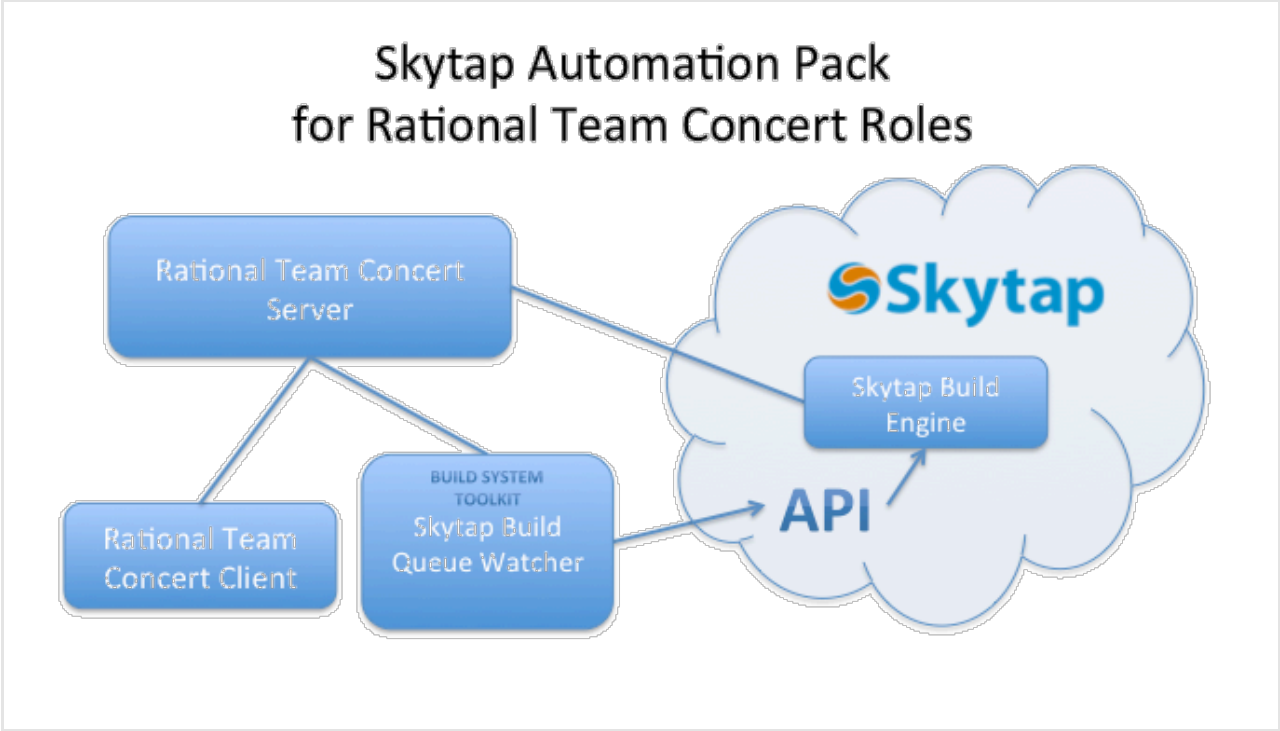
This Automation Pack enables easy integration of your Skytap environment with IBM's Rational Team Concert lifecycle management features. The integration enables IT operations and development teams to rapidly provision Rational Team Concert build machines on demand in Skytap, and make use of scalable, elastic, on-demand cloud resources to meet fluctuating demand for build machine capacity. Additionally, the virtual machines that are provisioned on demand using the Skytap Automation Pack can also be configured to shut down automatically so organizations only pay for the virtual machine build resources that they use.

The automation pack can be downloaded here (http://www.skytap.com/downloads/Skytap Automation Pack for Rational Team Concert.zip).

Contents

IBM Rational Team Concert Integration Roles

The Skytap/IBM Rational Team Concert integration provides dynamic provisioning of Rational Team Concert build environments in Skytap. As illustrated below there are four roles in this integration:



- **Jazz Team Server** – The Jazz Team Server is available from IBM and serves as the database repository for your controlled objects and work items.
- **IBM Rational Team Concert client** – The IBM Rational Team Concert client is available from IBM and provides the user interface for IBM Rational Team Concert. The IBM Rational Team Concert client allows users to request builds specified in project build definitions.
- **Skytap Build Engine** – The Skytap Build Engine, available from Skytap, is a process that monitors specified Jazz build queues in Skytap. At user-defined intervals, the Skytap Build Engine loops through a list of configured Jazz build queues looking for build requests to execute. If a candidate build request is found, the Skytap Build Engine claims that build result and launches the Rational Team Concert Jazz Build Engine process to execute the build. If no build requests are found, the Skytap Build Engine waits for a user-defined amount of time and then checks again for build requests. After the specified number of intervals where no build request candidate is identified, the Skytap Build Engine will shut down and optionally shut the machine down. The Skytap Build Engine is implemented as a wrapper around the IBM Jazz Build Engine and is installed into a virtual machine in Skytap. In addition to having the Skytap Build Engine installed, this virtual machine should also be configured to execute the desired type of build.

- **Skytap Build Queue Watcher** - The Skytap Build Queue Watcher, available from Skytap, is a utility that monitors Jazz build queues and interacts with the Skytap API to dynamically instantiate and start Skytap Build Engines when it detects build requests collecting in those build queues. At user-defined intervals, the Skytap Queue Watcher enumerates all of its configured build rules. For each configured set of build rules, the Skytap Queue Watcher loops through the Rational Team Concert build definition queues defined in that build rule, and if the configured threshold number of builds requests has been reached or exceeded, the Skytap Queue Watcher uses the Skytap API to instantiate and start one or more Skytap

Build Engine environments in Skytap to handle those builds. The Skytap Queue Watcher process monitors the Skytap Build Engines it has instantiated and, when a Skytap Build Engine shuts down, the Skytap Queue Watcher interacts with the Skytap API to delete that Build Engine configuration. The Skytap Build Queue Watcher can be installed on a dedicated machine, an existing Rational Team Concert Client, or on any other machine that has ongoing connectivity and access to the Jazz Team Server.

Skytap/Rational Team Concert Integration Installation and Configuration Overview

The workflow for the installation and configuration for the Skytap/Rational Team Concert Integration in an existing Rational Team Concert environment is as follows:

1. Create a new Skytap environment.
2. Add a virtual machine to the environment and configure it so that is has the necessary software and environment required for a particular build definition(s). If a single VM can support multiple build definitions, consider using one Skytap environment that has been configured for all of the required types of builds.
3. Ensure the Skytap environment has connectivity to the IBM Jazz Team Server.
4. Install the Skytap Build Engine onto the virtual machine and configure it as described in the Skytap Build Engine Installation and Configuration section below.
5. Shutdown the virtual machine and create a Skytap Template out of the Skytap Build Engine Configuration. Make a note of the template ID(s), as you will need this information when configuring the Skytap Queue Watcher.
6. Repeat steps 1-4 for each build definition for which you need to configure a distinct build machine.
7. Install the Skytap Queue Watcher on a machine that has ongoing connectivity to the IBM Jazz Team Server as described in the Skytap Queue Watcher Installation and Configuration section below.
8. Configure the Skytap Queue Watcher to monitor the desired build queues as described in the Skytap Queue Watcher Installation and Configuration section below.
9. Start the Skytap Queue Watcher.

Skytap Build Engine Installation and Configuration

The Skytap Build Engine Watcher is implemented as a Java JAR file. To run successfully in production it is advisable, though not required, to run the Skytap Build Engine as either a Windows Service or a *nix Daemon.

Skytap Build Engine Prerequisites

The following items need to be installed prior to installing and configuring the Skytap Build Engine.

1. **A Java 6 JDK**

- A JDK is required on Windows if the Windows Service provided with the Skytap/Rational Team Concert integration is used.
- A JDK is recommended for all platforms.

2. **IBM Rational Team Concert Build Toolkit**
   - The IBM Rational Team Concert Build Toolkit is available from https://www.jazz.net (https://www.jazz.net/). You will need a free account on jazz.net to access and download it.
   - The IBM Rational Team Concert Build Toolkit should be the same version as your IBM Rational Team Concert Jazz Team Server.
3. Any other applications or utilities that are required for the JBE build definitions to be supported by this build engine.

Skytap Build Engine Windows Service Installation

The Skytap Build Engine is an executable Java JAR file. The following instructions are provided as an example of how to install the Skytap Build Engine as a Windows Service using the Apache Commons Daemon. The components used in the following example are provided as-is in the Examples directory of the Skytap Automation Pack for Rational Team Concert.

1. Locate the Examples/Windows directory in the distribution of the Skytap Automation Pack for Rational Team Concert.
2. Unzip the SkytapBuildEngineService.zip archive into a new directory.
3. Inside the Examples/Windows folder you will find the following files: a. config.properties i. This is the configuration file; see Configuring the Skytap Build Engine for information on what this file contains. b. install.bat i. This is a batch file for installing as a service. c. SkytapBuildEngine.exe i. Windows Java Service wrapper (Apache Commons Daemon). d. SkytapBuildEngine.jar i. Java code that is the Build Engine. e. SkytapBuildEnginew.exe i. Windows Java Service wrapper configuration tool (Apache Commons Daemon). f. uninstall.bat i. Removes the windows service.
4. Edit the config.properties file to define your Skytap Build Engine behavior as described in the Configuring the Skytap Build Engine section below.
5. If you are going to run the Skytap Build Engine interactively (i.e. for configuration testing) and not as a Windows Service you can stop here.
6. If you want to run the Skytap Build Engine as a service, make the following changes to the install.bat file:
   a. If any of your paths below include spaces, put a set of double quotes around the path.
   b. Change --Install=c:rtcjazzSkytapBuildEngine.exe to match the path to the SkytapQueueWatcher.exe file.
   c. Change --LogPath=c:rtcjazzskytap_logs to the full path location of where you want to log files to land.
   d. Change --ClassPath=c:rtcjazzBuildEngine.jar to match the location of SkytapQueueWatcher.jar file and also change c:rtcjazzbuildsystembuildtoolkit* to match the full path to where you installed the build toolkit.

e. Change –Startup=manual to –Startup=auto if you want the Skytap Queue Watcher service to automatically start when the machine boots.

f. Save your changes.

7. Run the install.bat file as administrator.
8. Start the SkytapBuildEngine Service.

Skytap Build Engine *nix Installation

Most *nix systems are unique in how they initialize and run daemons. While there is some commonality with the init v5 systems, it is difficult to create a single java daemon to support all the possible variants of these systems. The Windows install section above lists the required files as well as the installation instructions for the Skytap Build Engine. You will, however, need to build a customized daemon for your specific platform utilizing a utility such as jsvc from Apache Commons Daemon.

Configuring the Skytap Build Engine

You configure the Skytap Build Engine by modifying parameters in the config.properties file. Each Build Engine parameter is on a single line. Parameter syntax is as follows: = Skytap Build Engine parameters that can appear in the config.properties file are as follows:

- jbePath
    - Full path to the IBM Rational Team Concert JBE.
- repository
    - This is the full IBM Rational Team Concert URL, e.g. repository=https://jazz:9443/ccm (https://jazz:9443/ccm).
- userId
    - The user id to use when logging onto the IBM Rational Team Concert Jazz Team Server to execute the specified build. This user must be a member of the Rational Team Concert project that is being built and must have a Rational Team Concert developer license.
- passwordFile
    - Location of the file that contains the encrypted user password.
- pass
    - An unencrypted version of the user’s password (it is recommended to use the passwordFile property instead of listing the password with this property in plain text).
- buildDefinitionIds
    - A comma separated list of build definitions that this particular build engine should check for builds against (and if found, build them).
- emptyLoops
    - Number of times of non-build loops before shutting down.
- sleepTime

    - Number of seconds between checks for builds.
- shutdownAfterExecution
    - True/false - should the OS be shutdown upon exit.
- loglevel
    - set this value to “info” for minimal logging. Set it to “verbose” for verbose logging. If this parameter is omitted, the default is “info”

Here is an example of a Skytap Build Engine config.properties file:

```
jbePath=/opt/jazz/jazz/buildsystem/buildengine/eclipse/jbe.sh repository=https://services-uswest.skytap.com:25584/ccm (https://services-uswest.skytap.com:25584/ccm) userId=bill passwordFile= pass=bill buildDefinitionIds=junit,junit.fetchCodeAndBuild emptyLoops=10 sleepTime=60 shutdownAfterExecution=false loglevel=info
```

Skytap Queue Watcher Installation and Configuration

The Skytap Queue Watcher is implemented as a Java JAR file. To run successfully in production it is advisable, though not required, to run the Skytap Queue Watcher as either a Windows Service or a *nix Daemon.

Skytap Queue Watcher Prerequisites

The following items need to be installed prior to installing and configuring the Skytap Queue Watcher.

1. A Java 6 JDK
    1. A JDK is required on Windows if the Windows Service provided with the Skytap/Rational Team Concert integration is used.
    2. A JDK is recommended for all platforms.
2. IBM Rational Team Concert Build Toolkit
    1. The IBM Rational Team Concert Build Toolkit is available from https://www.jazz.net (https://www.jazz.net/). You will need a free account on jazz.net to access it.
    2. The IBM Rational Team Concert Build Toolkit should be the same version as your IBM Rational Team Concert Jazz Team Server

Skytap Queue Watcher Windows Installation

The Skytap Queue Watcher is an executable Java JAR file. The following instructions are provided as an example of how to install the Skytap Queue Watcher as a Windows Service using the Apache Commons Daemon. The components used in the following example are provided as-is in the Examples directory of the Skytap Automation Pack for Rational Team Concert.

1. Locate the Examples/Windows directory in the distribution of the Skytap Automation Pack for Rational Team Concert.

2. Unzip the SkytapQueueWatcherService.zip archive into a new directory.
3. Inside this new directory you will find the following files:
    1. **config.properties:**This is the configuration file for the Skytap Queue Watcher. For more information on this file, see the Configuring the Skytap Queue Watcher section below.
    2. **install.bat:** This is a batch file for installing as a service.
    3. **SkytapQueueWatcher.exe:** Windows Java Service wrapper (Apache Commons Daemon).
    4. **SkytapQueueWatcher.jar:** Java code that is the QueueWatcher.
    5. **SkytapQueueWatcherw.exe:** Windows Java Service wrapper configuration tool (Apache Commons Daemon).
    6. **uninstall.bat:** Removes the windows service.
4. Edit the config.properties file to define your Skytap Queue Watcher configuration as described in the Configuring the Skytap Queue Watcher section below.
5. If you are going to run the Skytap Queue Watcher interactively and not as a Windows Service you can stop here.
6. Open up the install.bat file and make the following changes to match your prerequisite installations (if installing as a windows service). Note: If any of your paths below include spaces, put a set of double quotes around the path!
    1. Change --Install=c:rtcjazzSkytapQueueWatcher.exe to match the path to the SkytapQueueWatcher.exe file.
    2. Change --LogPath=c:rtcjazzskytap_logs to the full path location of where you want to log files to land.
    3. Change --ClassPath=c:rtcjazzSkytapQueueWatcher.jar to match the location of SkytapQueueWatcher.jar file and also change c:rtcjazzbuildsystembuildtoolkit* to match the full path to where you installed the build toolkit.e. Save your changes.
7. Run the install.bat file as administrator.
8. Start the SkytapQueueWatcher Service.

Skytap Queue Watcher *nix Installation

Most *nix systems are unique in how they initialize and run daemons. While there is some commonality with the init v5 systems, it is difficult to create a single java daemon to support all the possible variants of these systems. The Windows install section above lists the required files as well as the installation instructions for the Skytap Queue Watcher. You will, however, need to build a customized daemon for your specific platform utilizing a utility such as jsvc from Apache Commons Daemon.

Configuring the Skytap Queue Watcher

You configure the Skytap Queue Watcher by modifying parameters contained in the config.properties file. Each configurable parameter in the config.properties is on a line by itself. The configurable values in the config.properties file are as follows:

- repository
    - The full IBM Rational Team Concert Jazz Team Server URL, i.e. - https://jazz:9443/ccm (https://jazz:9443/ccm) .
- userId
    - The user id to use when logging onto the IBM Rational Team Concert Jazz Team Server. This user must be a member of the Rational Team Concert project that is being built and must have a Rational Team Concert developer license.
- passwordFile
    - Location of a file that contains the encrypted user password.
- pass
    - An unencrypted version of the user's password (we recommend use of the passwordFile property to avoid exposing the unencrypted user password with this property).
- sleepTime
    - The number of seconds to wait between checks for new pending builds.
- skytapuser
    - The Skytap username to use with the SkytapAPI.
- skytapapikey
    - The Skytap API Key (or password if API Key has not been setup) for the Skytap user specified.
- skytapmaxconfigs
    - The maximum number of active Skytap environments this QueueWatcher can start and have running (set to 0 for unlimited).
- loglevel
    - Set this value to "info" for minimal logging. Set it to "verbose" for verbose logging. If this parameter is omitted, the default is "info".
- nettype
    - Set this value to "vpn" for VPN network connectivity. Set it to "icnr" for connectivity between Skytap environments.
- netid
    - For networks of type VPN, set this value to the VPN ID. For networks of type ICNR, set this value to the ID of the Skytap environment containing the Rational Team Concert server and the Skytap Queue Watcher.
- buildRules
    - The buildRules parameter contains the values that control when the Skytap Queue Watcher will instantiate a Skytap Build Engine. The buildRules parameter is a vertical-bar separated list that includes the following values:
        - Build Definition Ids - a semi-colon delimited list of build id(s) to which this rule applies. For example junit;junit2;junit3.

- Pending build count – the number of pending build requests required to trigger instantiation of a Skytap Build Engine.
- Skytap template Id – This identifies the Skytap Template to use for instantiating the appropriate Skytap Build Engine if the pending build count is met or exceeded.
- Max number of active Skytap environments – This is the maximum number of Skytap Build Engines this build rule can have running at any given time (this parameter is optional, if omitted or set to 0, then the maximum number of environments is limited by the value in the skytapmaxconfigs parameter.
- buildRules=junit|1|222123
    - Would watch the junit build definition queue, once a pending build was found, template 222123 would be created as an environment with no limit on the number of environments created unless skytapmaxconfigs has been reached).
- buildRules=junit;junit.2|1|222123
    - Same as above, but junit and junit.2 build definition queue's have the same ruleset applied to them.

```
Here is an example of a Skytap Queue Watcher config.properties file:
repository=https://http://host1.skytap.example:9443/ccm
(https://http//host1.skytap.example:9443/ccm) userId=bill passwordFile= pass=bill sleepTime=15
loglevel=info nettype=icnr netid=595708 skytapuser=tmilligan
skytapapikey=8694aac56591616e223ad4b4a758ee71f7ae3162
buildRules=junit;junit.fetchCodeAndBuild|1|222741|2 skytapmaxconfigs=2
```

VPN Requirements for Rational Integration

If you are using a Skytap VPN to access the Skytap Build Engine from your on-prem network, you will need to adhere to the following requirements:

The subnet mask for the Skytap VPN must be set to allow the Skytap Queue Watcher process to modify the third octet in the Subnet. Typically this would be a mask of 16 to 23 bits. In the example below, the mask is set to 20 bits, allowing a maximum of 15 build engines.

The following table provides a mapping of subnet mask bits to the maximum number of build engines possible:

Subnet Bits	Maximum Skytap Build Engines
16	255
17	127
18	63
19	31
20	15
21	7
22	3
23	1