Continua CI Repositories

Repositories

A repository in Continua CI is a reference to an external Version Control System (VCS) that provides a way for Continua CI to interact with that VCS. Adding a repository and assigning it to a Configuration allows the user to make use of the VCS's content in Continua CI. Some examples of using a repository in Continua CI include:

- Automatically trigger a build whenever a check-in occurs in a tracked VCS.
- Referencing components of the VCS in build actions.
- Monitor and record changes in the repository. These include changesets that triggered a build or changesets that have been added since the previous build was run.
- Show changesets that were committed by the current user. For a user to correctly view their own changes, they will first need to configure their Repository User Mappings.
- Link a repository to an Issue Connector. This allows you to automatically link repository check-in comments to an issue in your issue tracking system.

Repository Caches

Unlike some other continuous integration servers like FinalBuilder Server, Continua CI manages the retrieval of your source code. When you create a repository in Continua CI, it points the Continua server to your VCS. The polling frequency of each repository can be set to specify how often the Continua server polls the VCS for changes.

Continua maintains several caches of your source code: one on the Continua server and one on each agent that has previously used that repository. When Continua detects a new commit/check-in/changeset in your repository, it updates the server cache, then each agent that has that repository cached updates its cache from the server. This means that when it's time to run a build the source is most likely already on the Agent.

The first time that an agent uses a repository its cache needs to be created. Depending on the size of your source, this can mean that the initial stage on that agent can take significantly longer than usual. After the cache is created though, keeping it updated is usually very quick (unless you add large binaries etc).

Building the right code base

The cache isn't just a bunch of files - it is versioned. This makes it easy for us to get the right version of the source to build.

When a build is initialised, either manually or from a trigger, Continua will check if there are any new changesets for each repository in the configuration and, if it finds any, update the repository cache.

At this point the latest changeset for each repository will then be associated with the build. Continua will then try to reserve an agent to run the first build stage. After this point the build may need to wait for an agent to be available and further changesets may be detected, however all build stages will always use the source code from the latest changeset associated with the build.

For example, lets say you have a Repository Trigger with no quiet period set. Given the following steps:

1. A new commit (Change1) is detected, which queues up a new build.
2. When the build is just starting, a new change (Change2) is detected in your repository, which is automatically synchronised to all of the agents.
3. Your build starts running on an agent.

Change2 is in the agent's cache - but we don't want to build it. We want to build the code that triggered the build. So, we get the version of the cache that contains the most recent changeset associated with the build. In this case, that will be the changeset that triggered the build (Change1). Change2 will have triggered its own build, which will build the version of the source associated with it.

Note if the build is initialised by a Repository Trigger and a quiet period set has been set, Continua will continue to associate any new changesets it detects on the triggering repository with the same branch as the triggering changeset until the quiet period has ended. The build will then continue with each stage using the latest changeset associated with the build before the quiet period ended.

For example, lets say you have a Repository Trigger with no quiet period of 1 minute, which means that the build will wait 1 minute after its triggered to see if any other changes come in. Given the following steps:

1. A new commit (Change1) is detected, which queues up a new build.
2. During the quiet period, a new commit (Change2) is detected, and this is added to the changesets associated with the build.
3. Just after the quiet period has ended, (and the build is just starting) a new change (Change3) is detected in your repository, which is automatically sync'd to all of the agents.
4. Your build starts running on an agent.

We then get the version of the cache that contains the most recent changeset associated with the build. In this case, that will be the last changeset detected during the quiet period (Change2).

This also allows us to run stages on different agents. For example, your Build stage might run on Agent1 and your Installer stage on Agent2. When the Installer stage gets to Agent2, it gets the same version of the cache that was used for the Build stage. Without this versioned cache, we would need to run every stage on the same Agent to ensure that the correct source was used.

Repository Architecture

The diagram below demonstrates the architecture used by Continua CI.