Development Process

This is a proposed way of working, under discussion on the mailing list. See http://markmail.org/message/xoyjw2sduenlytwm

Commit to master through PR only was decided on here: http://comments.gmane.org/gmane.comp.apache.cloudstack.devel/62223

Summary

Apache CloudStack is a broad and encompassing project covering many areas that require in-depth technical knowledge. As such, it is not possible for one person to know and/or test all components of CloudStack in depth. Miscommunications and misunderstandings between developers can lead to poor quality in CloudStack's code and cause community to deliver a unstable release. Knowledge lost from one release to the next can cause regressions in the current release. This page details the development process the CloudStack community must follow in order to ensure the community delivers high quality releases. This page is a must read for all developers and testers.

Process Principles

This process is developed with the following principles.

- Continuous integration is key to a stable code base.
- Automation is the only insurance the community has against regressions in releases.
- Constrain the actual hardware testing to testing CloudStack's physical/virtual orchestration.
- Use simulators to speed up testing of CloudStack's self-service business logic.
- Properly performed code reviews are key to catching problems early.

Development Process

Infrastructure Setup

The development infrastructure is setup as follows.

- Continuous Integration Testing is always running on master and the current release branch.
- Developers can request from Jenkins that automation be run on their branch to make sure the new code and new testcases did not break continuous integration.
- No merges are allowed without a successful run of the automation.

Checkin and Merge Process

The process is very simple.

1. Fork the code on Github and send a pull request (even if you are a committer)
2. Assign the JIRA issue to yourself, change the status to "In Progress", add the branch information into the issue.
3. When the work is completed, do the following things.
    a. Send the pull request and then ask two colleagues to review and give their "LGTM".
    b. Testing will be done on your pull request, the Travis CI output will appear on the pull request.
4. Once review and automation are completed successfully, a committer will merge the branch to the release branch or master.
5. Once the merge is completed, change the status of the issue to "Resolved."

Review Process

- It is up to the developer to find the right people to review his/her code changes.
- It is within the reviewer's right to decide whether they are the appropriate reviewer for a code change.
- If they are not or if the review requires technical expertise from others, the reviewer can either refer the developer to another reviewer or pull in another reviewer.

Automation Process

- Developers uses Jenkins to schedule automation test (BVT) to be performed on their branch.
- If the BVT is not completed successfully, the BVT uploads all logs collected and the testcase that failed.
- Developers can fix their code/tests and repeat until the BVT is successfully completed.
- If a developer cannot figure out the problem from the logs uploaded, they can add more logging to more precisely pinpoint the error and reschedule the BVT on their branch.

Continuous Integration

The BVT will be run on master and the current release branch on a continuous basis. If the BVT fails, the commits submitted between the last successful BVT run and the failed BVT runs are reverted. The developers who submitted the commits are notified of the revert and can merge their changes again when they have figured out the problem on their own branch.

Developer Responsibilities

As a code contributor to Apache CloudStack, regardless of if you have committer or contributor status, your responsibilities are as follows:

Acquiring Knowledge

- Understand how to use JIRA to communicate your work with release managers, users, developers, and testers.
- Understand CloudStack's architecture and where you should or should not make changes.
- Understand CloudStack's simulator and how to run it and how to change it to test your code.
- Understand CloudStack's Marvin framework and how to use it to write automated tests.
- Understand CloudStack's code check-in process.
- Understand CloudStack's review process and your role in it as the reviewer and the code submitter.

JIRA

Communicating what issues you're working on and how you're working on it is an important part of participating in the community. We do that on the mailing list. However, you should use JIRA to update issue status and add comments with conclusions drawn from mailing list discussions. This allows release managers, users, and testers to follow the issue better, to understand if the issue is being worked on, to get a better feel of whether an issue will be resolved in time for the release.

Writing Automated Tests

Automated tests ensures that your feature will not be regressed. Anyone who submits a code change that breaks an existing automated test is responsible for fixing that automated test or revert the code change.

Q&A

Q: I'm a contributor and not a committer. How do I participate in this process?

A: You can send a pull request on github. When two other collaborators give a LGTM, a committer will merge your pull request.

Q: Who is providing this jenkins?

A: Currently, the jenkins instance will be provided by Citrix, mainly because we're still working out an infrastructure testing lab in ASF. When that is ready, Citrix is willing to donate the setup necessary to run continuous integration to ASF.

Q: I like to set this up in my own organization so that we can run CI on our own development branches. Can I do this?

A: The setup itself is publicly available. The people who made this possible: Amogh, Bharat, Santhosh, are all on the mailing list. You can communicate with them via the mailing list and ask them help you with the setup. We will try to document this setup as much as possible once it's available.

Q: I currently have a Jenkins slave that runs tests for my own code on every build. Can you add my slave to your jenkins?

A: Currently, no. There's a lot of issues involving jenkins in Citrix making call to outside of Citrix that really I'm not willing to go resolve. For now, you just have to run that with the ASF jenkins.