Continuous Integration and Delivery at NSIDC

Julia Collins
The Goal

Continuous integration and Continuous delivery

?
Continuous integration: Merge changes frequently into shared code base (and run unit/integration tests) to catch conflicts early.

Requires:

- Code repository
- Automated build (build all commits)
- Automated tests
- If fail, fail fast
Definitions

**Continuous delivery:** A software development discipline where you build software in such a way that the software can be released to production at any time.

http://martinfowler.com/bliki/ContinuousDelivery.html
Definitions

Continuous deployment: Passing builds are automatically deployed to the production environment

Development Context

Iterative development with frequent feature releases
Jenkins (was Hudson)
Jenkins

Open source (MIT license)
Core installation great for running scripts
Can run on a schedule, poll SCM for changes
Can be extended via plugins. Many (many many) plugins already exist.

Cons: Plugin developers not necessarily writing with other (unrelated) plugin behaviors in mind.
“Artifact”

The by-product of the build; the testable entity

tar file

jar file

SCM tag or revision number

a running virtual machine
We had no artifact repository

* **Solution:** Roll our own
* **Result:** Overhead added to job configurations to support the artifact handling
One day you look around . . .

http://xkcd.com/1319/
But really, automation is good

- Consistent steps, no matter who’s running the build
- History and reproducibility
- Build configuration conveys information about application environment and structure
Meanwhile . . .

Moving from svn towards git

* Bitbucket repositories
* Take advantage of git workflows
Branching Strategies

- **Master (trunk):** Always in deployable state
- **Feature A**
- **Feature B**

Merge master into Feature A
Many jobs, one build server

* Applications using different versions of same language or libraries
* Conflict between projects needing upgrades vs. those frozen at a particular version
* Would like option to build, test and run applications on different OS than default
The Wish List

Feature workflow:

* The ability to build/test/approve a feature branch.
* The ability to merge master into a branch prior to build/test.
* The ability to merge all successfully integrated (built/tested) feature branches into master for review within the QA environment.
* The ability to push the local master (with all approved feature changes) to origin/master on the remote repository, along with tags indicating released code.
The Wish List

Build and deployment:

* The ability to create multiple build environments to satisfy different application needs for language versions and supporting libraries.

* A continuous build environment that produces/manages deployment-ready artifacts, including managing the state of approval (or not).

* Deployment: Ops must be able to identify the approved artifact(s) and run the required steps for deployment.

* Deployment: Ops must be able to easily roll back to a previous release.
Road Map

* Stick with Jenkins, OR
* Transition to a different build server
Jenkins in the future

* Use existing plugins to provide needed functionality (artifact handling, master/slave implementation for multiple build agents)
* Contribute to plugin development
* Move in-line shell commands to application task runners (remove redundancies)
Bamboo (Atlassian)

Build Plan

Run in order

Stage 1

Stage 2

Stage 3

Job 1

Run in order

task 1

task 2

task 3

Job 2


task 1

task 2

task 3

Job 3


task 1

Jobs run in parallel (different agents)
Bamboo
Bamboo
The Critical Facts

Jenkins: “free”

TeamCity

- Free edition: 3 build agents, 20 build configurations
- $299: 1 additional build agent, 10 additional configs
- $1999+: Enterprise options

Bamboo

- Hosted vs. local options
- $10: No remote agents, 10 jobs
- $800: 1 remote agent, unlimited jobs
- $2200: 4 remote agents, unlimited jobs
Lessons Learned

* Build configurations tend to proliferate.
* With some effort, Jenkins can support many requirements.
* It’s not just using the right tool, it’s how you use it.
References

- **Jenkins**: http://jenkins-ci.org
- **TeamCity**: http://www.jetbrains.com/teamcity/
- **Bamboo**: https://www.atlassian.com/software/bamboo
- **CI**: http://martinfowler.com/bliki/ContinuousDelivery.html

Julia Collins
collinsj@nsidc.org