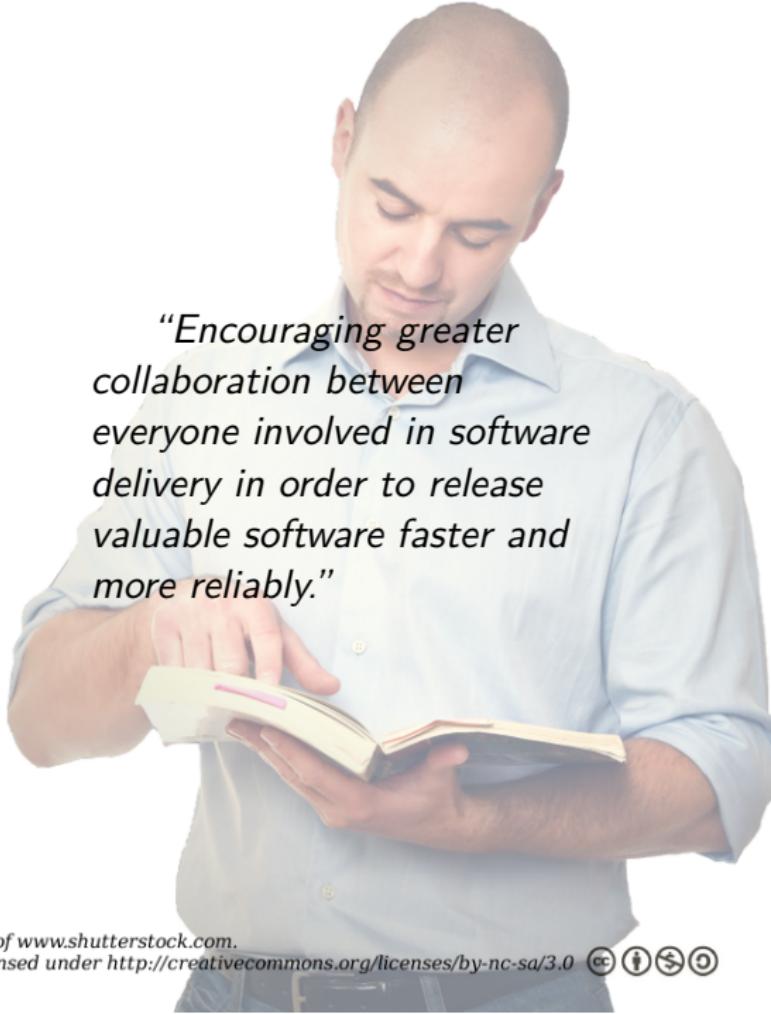
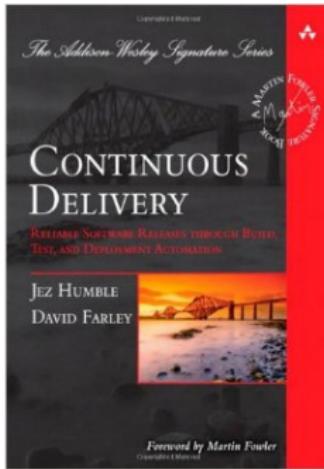


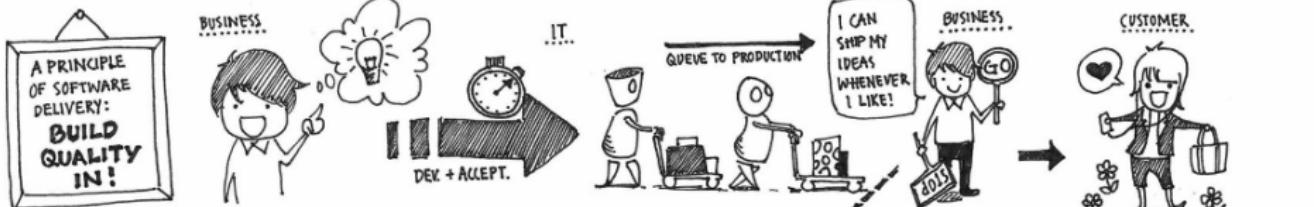


What is this all about?

- A brief introduction to Continuous Delivery.
- How each tool fits in the big picture.
- The approach I propose, without the pain.
- Help to get you started on your own.
- Advance towards Continuous Deployment.



"Encouraging greater collaboration between everyone involved in software delivery in order to release valuable software faster and more reliably."



CONTINUOUS DELIVERY

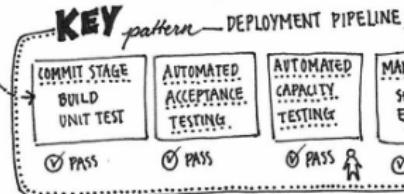
BY JEFF HUMBLE & DAVID FARLEY

A CLOSER LOOK
COMMIT STAGE

- CREATING EXECUTABLE CODE MUST WORK. VERIFIES THAT THE SYNTAX OF YOUR SOURCE CODE IS VALID

- UNIT TEST PASS

- FULFILL CERTAIN QUALITY CRITERIA SUCH AS TEST COVERAGE AND OTHER TECHNOLOGY-SPECIFIC METRICS



EXAMPLE

DONE
MEANS
RELEASED

CHANGE

CREATE NEW
INSTANCE
OF PIPELINE

- CHANGE IN
 - EXECUTABLE CODE
 - CONFIGURATION
 - HOST ENVIRONMENT
 - DATA

CHANGES
PIPELINE 1
PIPELINE 2
PIPELINE 3



• ANY CHANGE IS A TRIGGER • FAST • ACT ON IT

“

ENCOURAGING GREATER COLLABORATION
BETWEEN EVERYONE INVOLVED IN
SOFTWARE DELIVERY IN ORDER
TO RELEASE VALUABLE
SOFTWARE FASTER
AND MORE RELIABLY.

”

if it hurts, do
it more frequently

BENEFITS



EMPOWERED - IN CONTROL
LOW STRESS - SMALL RELEASES

REDUCING ERRORS
- CONFIG MGT.
- VERSION CONTROL



DEPLOYMENT FLEXIBILITY
- EASY TO
START APPLICATION IN NEW
ENVIRONMENT



PRACTICE MAKES
Perfect

SEEMS LIKE THE AUTHORS CAN'T STRESS IT ENOUGH. IT'S EVERYWHERE THROUGHOUT THIS BOOK.



VERSION
CONTROL



AUTOMATE
ALMOST
EVERYTHING

Bots welcome

- Speed up the release of new features.
- Special focus on risk: automate everything!
- Advance towards Continuous Deployment.
- No need for code freeze.
- Automated tagging.

“Apache Maven is a software project management and comprehension tool. Based on the concept of a project object model (POM), Maven can manage a project’s build, reporting and documentation from a central piece of information.”

maven

<http://apache.maven.org>

Convention

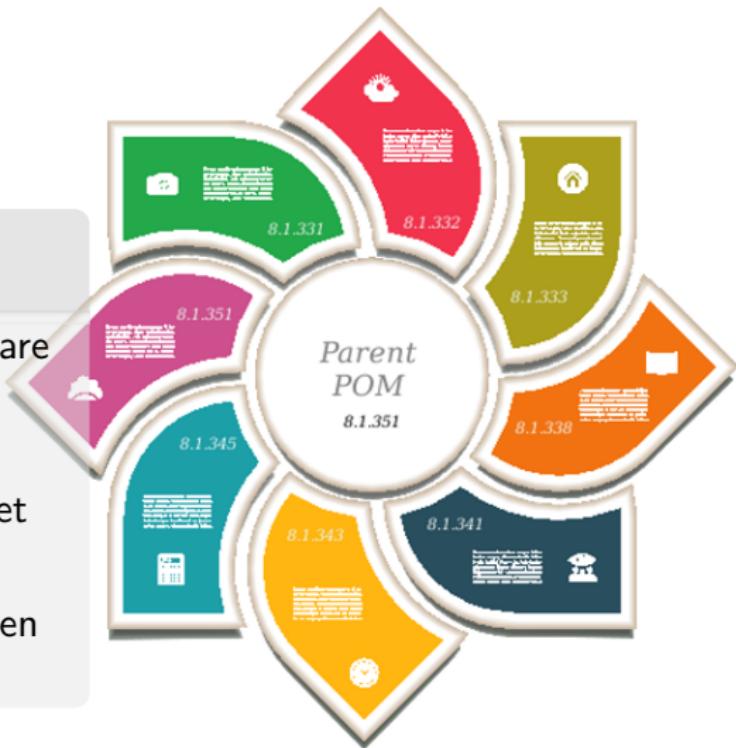
- All logic is isolated in its own module.
- No multi-module projects, unless for WARs.
- All modules inherit from a common, logic-less module: the parent POM.

According to Maven, a parent...

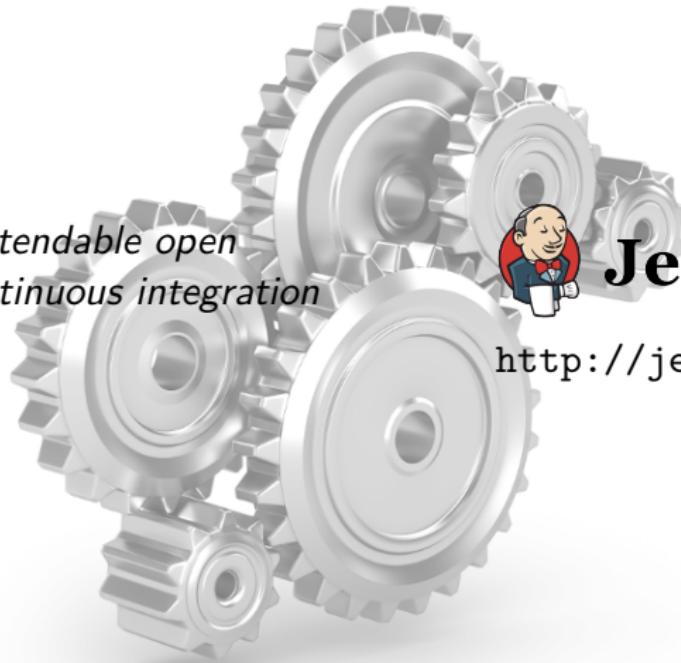
- Declares all dependencies in one place.
- Ensures all modules use the same versions of the dependencies.
- Defines common configurations for maven plugins.
- Simplifies child poms.
- Uses Maven properties to specify the versions of in-house modules.

Versions are unique and time-ordered

- All in-house modules, share the same version: `latest-SNAPSHOT`.
- All local environments get always up-to-date code.
- Versions are resolved when generating releases.



*"An extendable open
source continuous integration
server."*



Jenkins

<http://jenkins-ci.org>

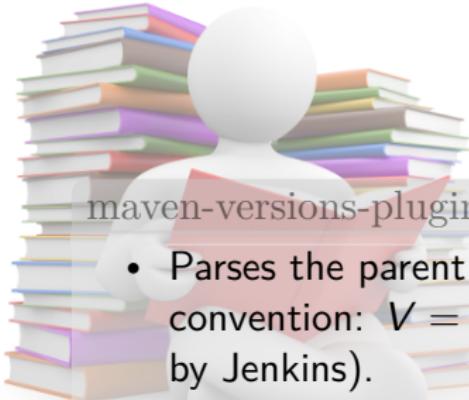
Generates unique version numbers

- Helper job to automate the tagging and packaging process.
- Checks out parent-pom code.
- Parameterized job with a single parameter: the name of the module triggering the release.
- Should have higher priority to avoid slot starvation and deadlocks in Jenkins.
- Expects parent-pom to contain two properties: `version.major` and `version.minor`.



One job to rule them all

- When a commit is pushed to the remote repository, Jenkins launches the associated job.
- The job is a Maven job, which runs `mvn deploy`.
- If it succeeds, calls `get-new-version` with its own name as parameter.



maven-versions-plugin magic

- Parses the parent pom and defines a new version using a convention: $V = \text{major}.\text{minor}.BUILD_NUMBER$ (provided by Jenkins).
- Using **maven-versions-plugin**:
 - Points itself to new version V .
 - Uses the latest released versions for all modules.
 - Ensures the version for the triggering module becomes V .
- Builds a release the Maven way, with **maven-release-plugin**.
- Publishes the new pom, with references to the latest released versions of each module.

Madrid 2014/11/22

www.codemotionworld.com

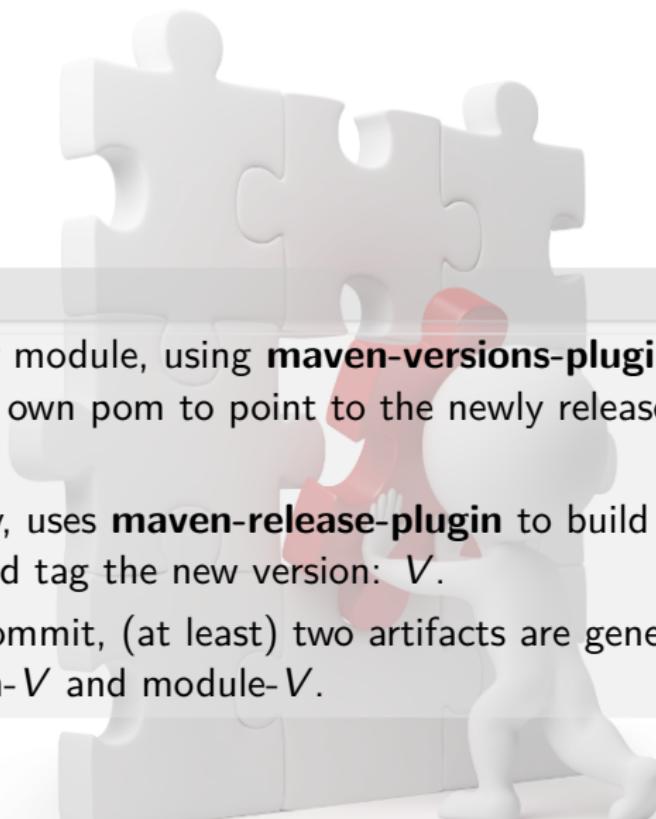


{codemotion}

Images courtesy of www.shutterstock.com.

Slide content licensed under <http://creativecommons.org/licenses/by-nc-sa/3.0>



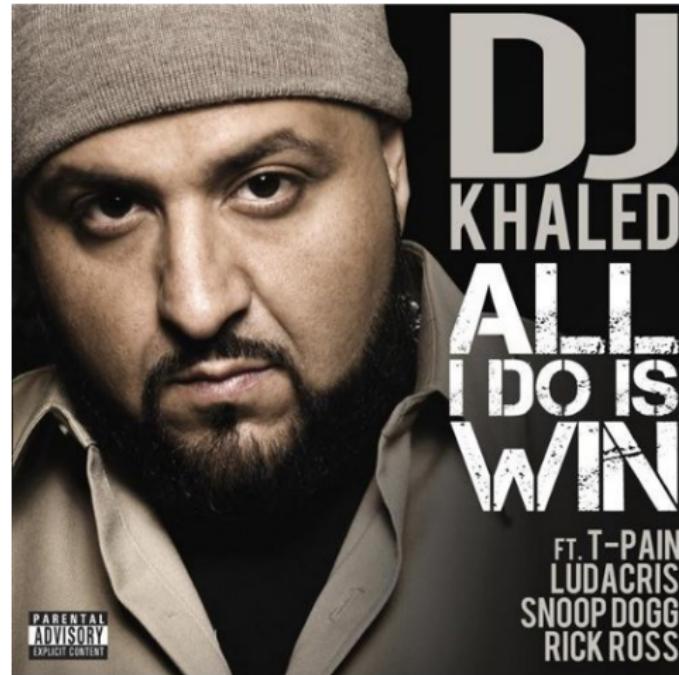
A large white 3D-style puzzle piece is positioned behind a smaller red one. They appear to be interlocking. The background is a light gray.

Finally

- The trigger module, using **maven-versions-plugin** again, updates its own pom to point to the newly released parent pom.
- Accordingly, uses **maven-release-plugin** to build all required artifacts and tag the new version: V .
- For each commit, (at least) two artifacts are generated: parent-pom- V and module- V .

Madrid 2014/11/22

www.codemotionworld.com

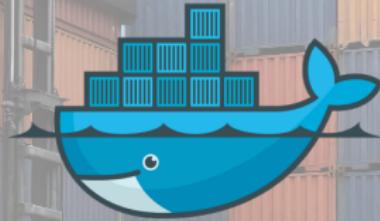




Maven Embedded is too embedded

- Maven jobs in Jenkins run Maven Embedded engine.
- Maven annotates parent jobs as dependencies in the dependency graph.
- For *get-new-version* to work, it cannot be a Maven job: It has to call Maven from the command line.
- Otherwise, it triggers an infinite loop of downstream jobs.

"An open platform for distributed applications for developers and sysadmins."



<http://www.docker.com>

Docker

“The Docker Engine container comprises just the application and its dependencies. It runs as an isolated process in userspace on the host operating system, sharing the kernel with other containers. Thus, it enjoys the resource isolation and allocation benefits of VMs but is much more portable and efficient.”

Packaging applications

- **Image:** Packaged application and dependencies. Ready to launch.
- **Container:** An isolated (process, memory, network, etc.) environment, running an *image*.
- **Volume:** A folder within a container, accessible from the host. Can be directly mapped to a folder in the host.

Running applications

- **Link:** Docker mechanism to help containers communicate with each other. It's defined as `-link container:alias`:
 - *container*: the name of the external, already running container,
 - *alias*: the name used locally in the new container, pointing to the external container. Docker adds it to `/etc/hosts`, and defines some environment properties.
- **Exposed port:** Docker service can map host ports to internal ports, when the container starts.

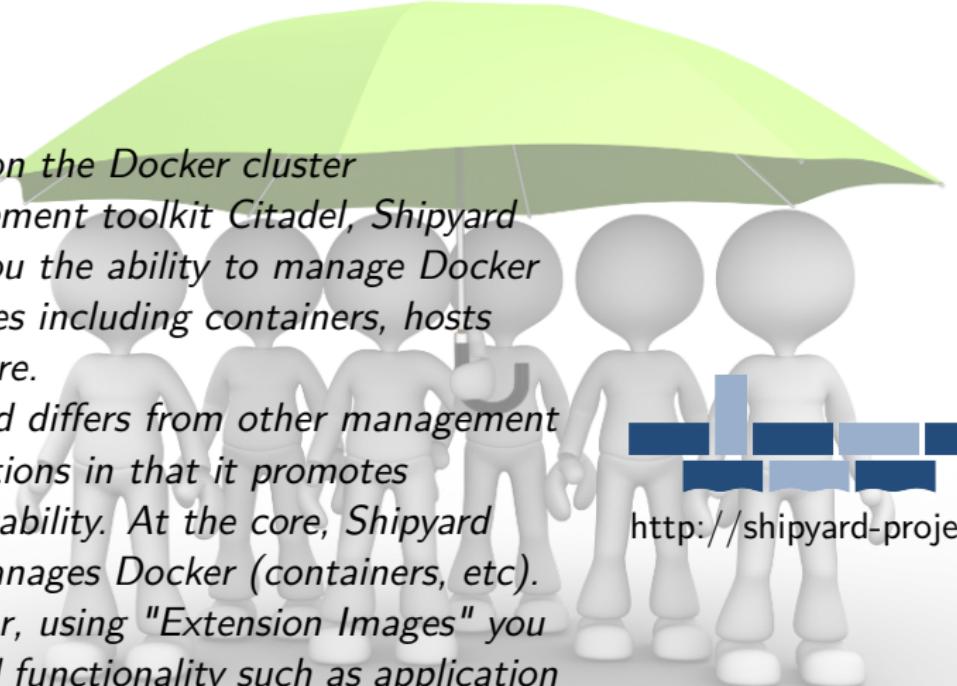
Cleaning things up

- A minimal Ubuntu base image modified for Docker-friendliness.
- Takes care of the problem of:
 - Zombie processes,
 - Logger daemon,
 - Cron jobs.
- Motivation explained in their website: “Your Docker image might be broken without you knowing it”

<https://phusion.github.io/baseimage-docker/>

Variables in Dockerfiles

- Based on wking's approach and code for Gentoo-based images: <https://github.com/wking/dockerfile>
- Modified for phusion-baseimage.
- Enhanced with in-house bash scripting framework: dry-wit.
- Allows placeholders in Dockerfiles.



"Built on the Docker cluster management toolkit Citadel, Shipyard gives you the ability to manage Docker resources including containers, hosts and more.

Shipyard differs from other management applications in that it promotes composability. At the core, Shipyard only manages Docker (containers, etc). However, using "Extension Images" you can add functionality such as application routing and load balancing, centralized logging, deployment and more."



<http://shipyard-project.com>

“Citadel is a toolkit for scheduling containers on a Docker cluster.”



<http://citadeltoolkit.org>

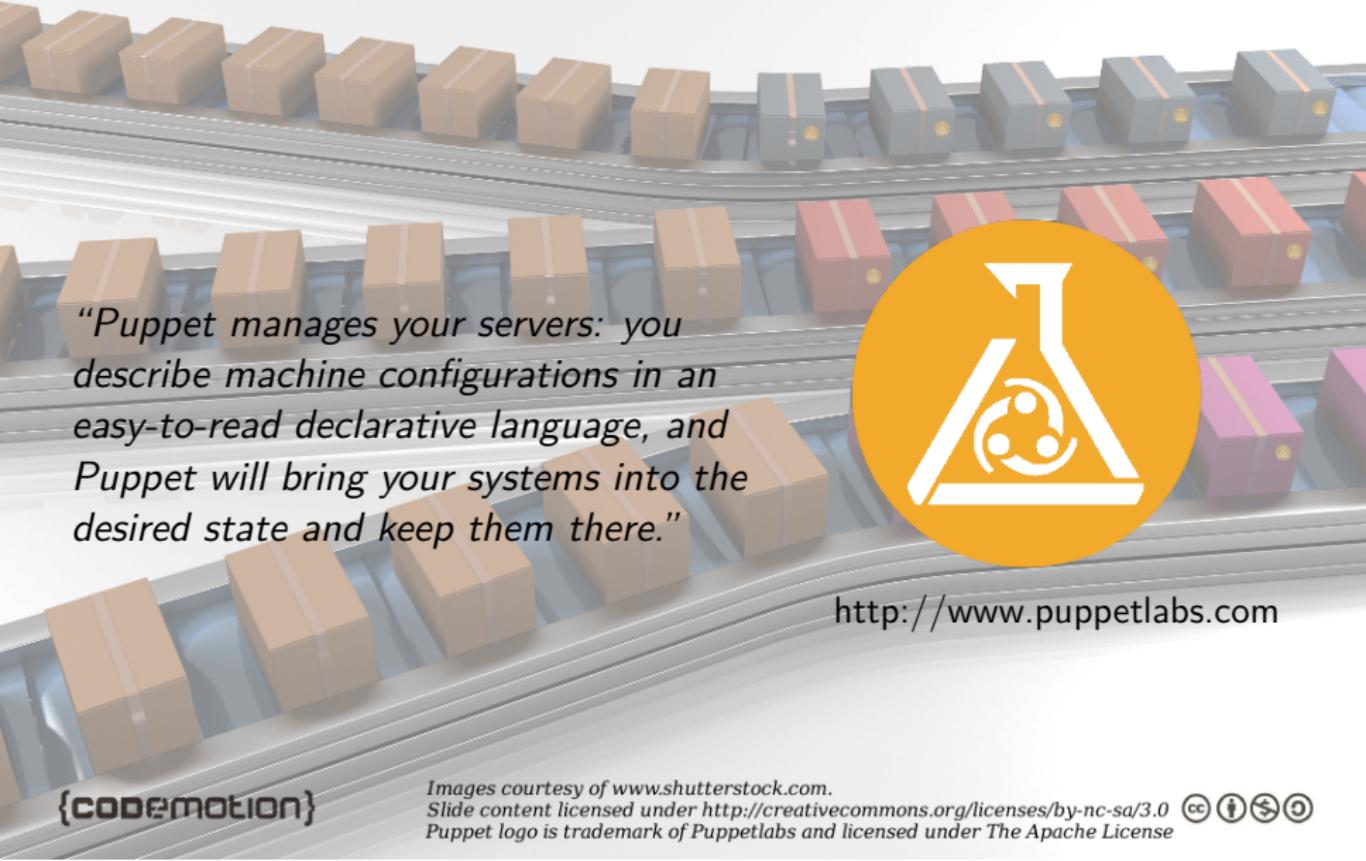
Images courtesy of www.shutterstock.com.

Slide content licensed under <http://creativecommons.org/licenses/by-nc-sa/3.0>

Citadel logo is subject to the Citadel license available at
github.com/citadel/citadel/blob/master/LICENSE



Puppet

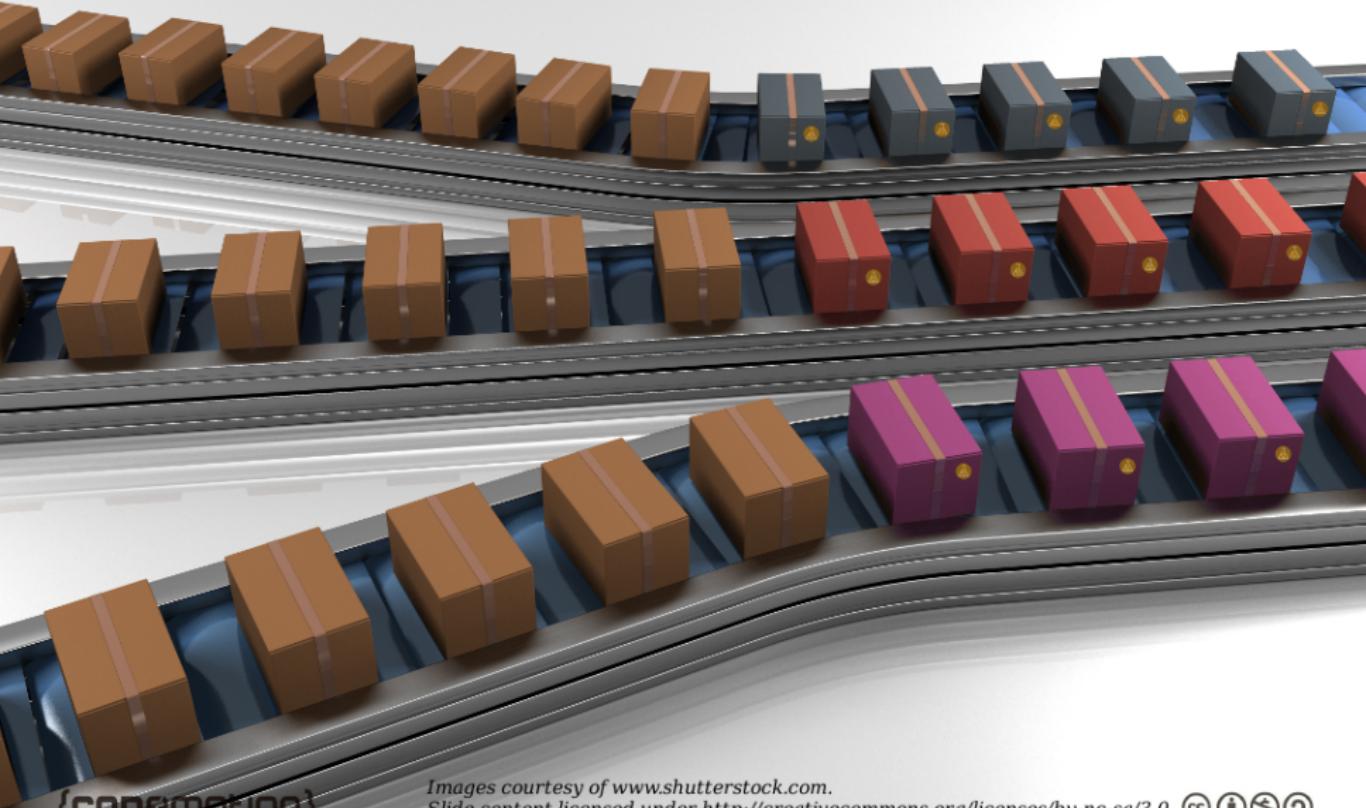


"Puppet manages your servers: you describe machine configurations in an easy-to-read declarative language, and Puppet will bring your systems into the desired state and keep them there."



<http://www.puppetlabs.com>

Puppet



Pros

- Images can be deployed anywhere.
- It doesn't require a convention to map host volumes or data containers.
- Containers can respond to changes propagated via Puppet.

Cons

- Containers take much longer to start.
- Automatic generation, auto-sign, and auto-accept SSL certificates.
- Puppet infrastructure required in production.

Pros

- Containers are stateless.
- Containers launch fast.

Cons

- Containers need to be prepared to read their configuration from plain files.
- The command for launching containers depends on Puppet configuration for that host.
- Puppet infrastructure required in production.

Pros

- Puppet sets up the configuration for environment-aware images.
- No Puppet needed in production: just links to data containers.
- Launching containers do not depend on the host.

Cons

- SSL certificate magic takes place on intermediate Docker images.
- A change in Puppet requires rebuilding the images, replacing the data-containers, and probably the application containers as well.



Tomcat logo is licensed under The Apache License.

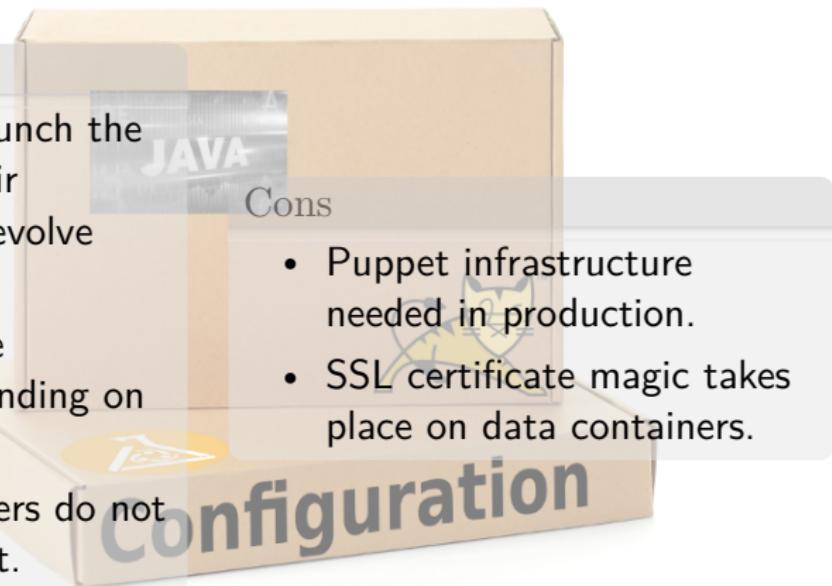
Images courtesy of www.shutterstock.com.

Slide content licensed under <http://creativecommons.org/licenses/by-nc-sa/3.0>

Puppet logo is trademark of Puppetlabs and licensed under The Apache License

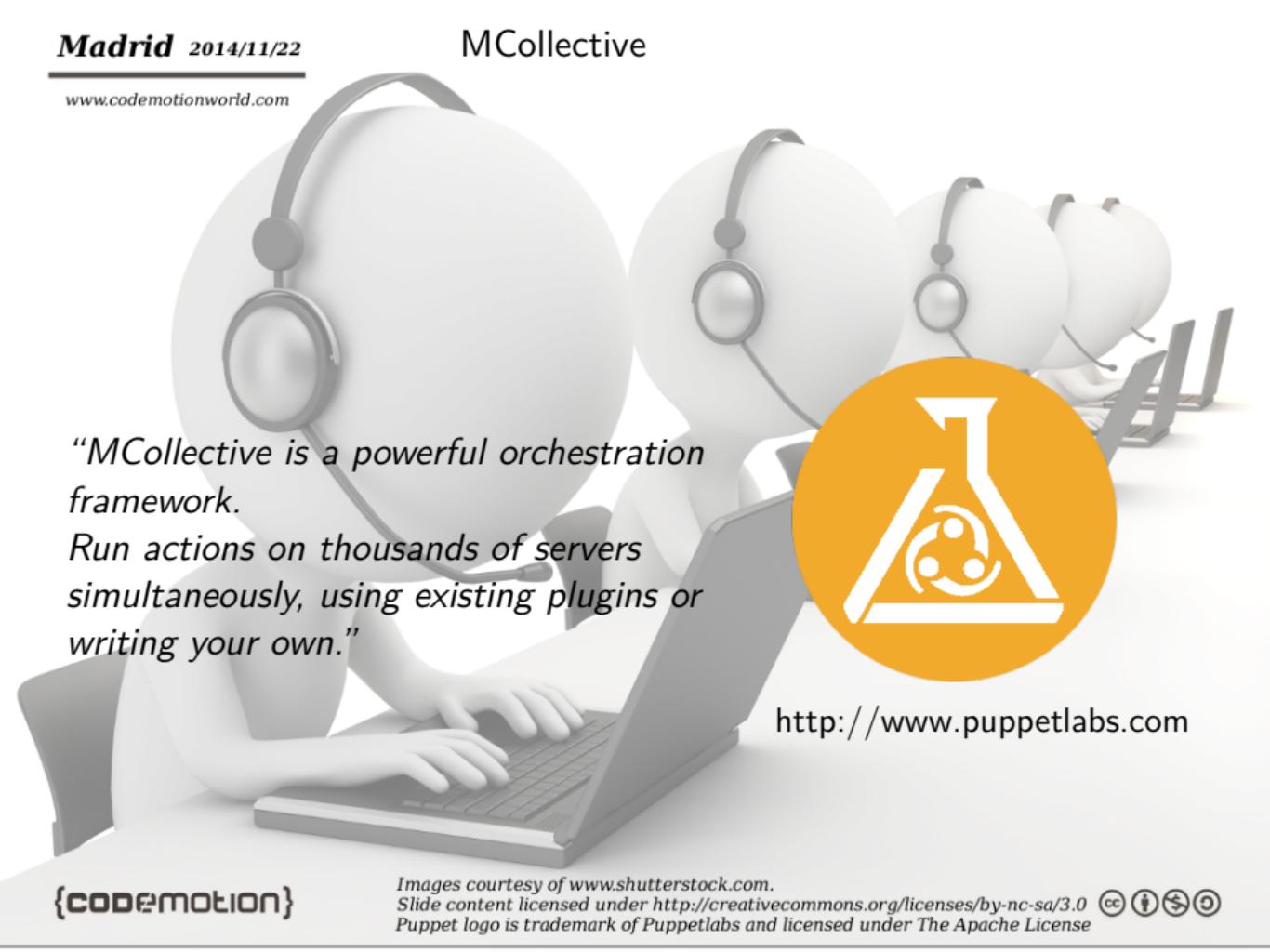
Pros

- Data containers launch the Puppet agent: their configuration can evolve over time.
- Puppet sets up the configuration depending on the environment.
- Launching containers do not depend on the host.



Cons

- Puppet infrastructure needed in production.
- SSL certificate magic takes place on data containers.



"MCollective is a powerful orchestration framework.

Run actions on thousands of servers simultaneously, using existing plugins or writing your own."



<http://www.puppetlabs.com>

ssh in a loop (1)



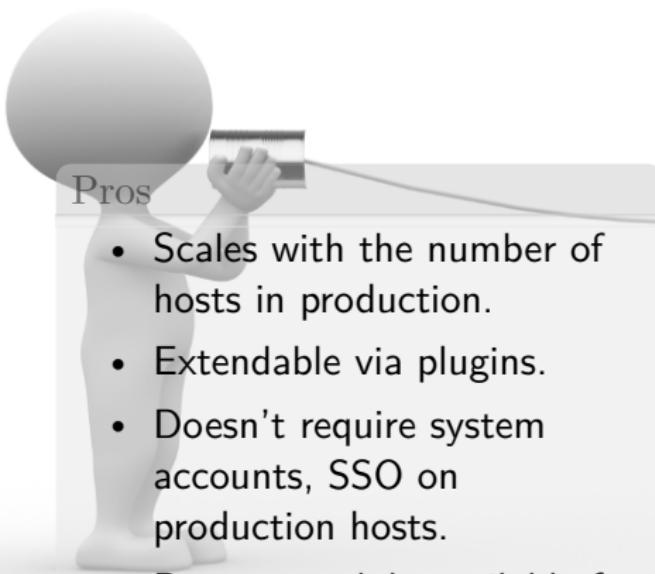
Pros

- Simple and straightforward.
- Fast enough up to a certain number of hosts.
- Easy and cheap to adapt to perform different tasks.
- Scriptable.



Cons

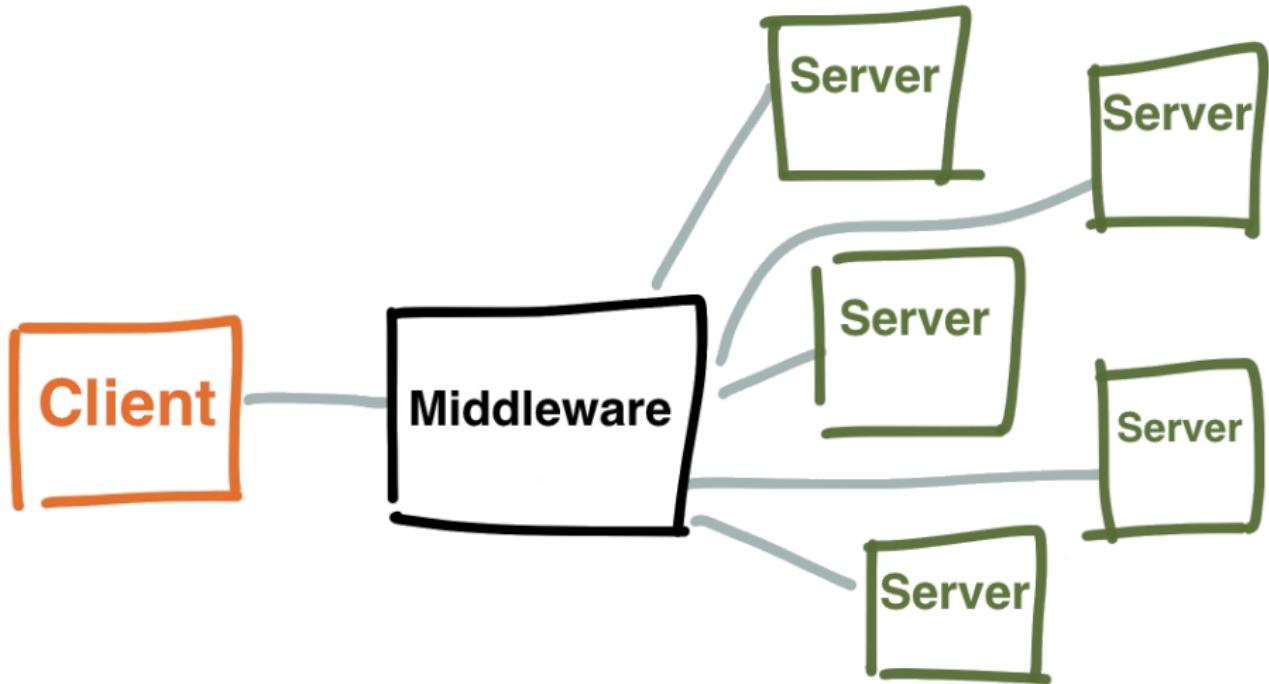
- Scripts with hard-coded host names or IPs.
- Requires way too much information about the production environment.
- Cannot easily run remote commands which expect some kind of interaction.
- When the number of host grows, the risk of overlook reported problems increases.
- Requires dealing with account permissions, SSO, etc.

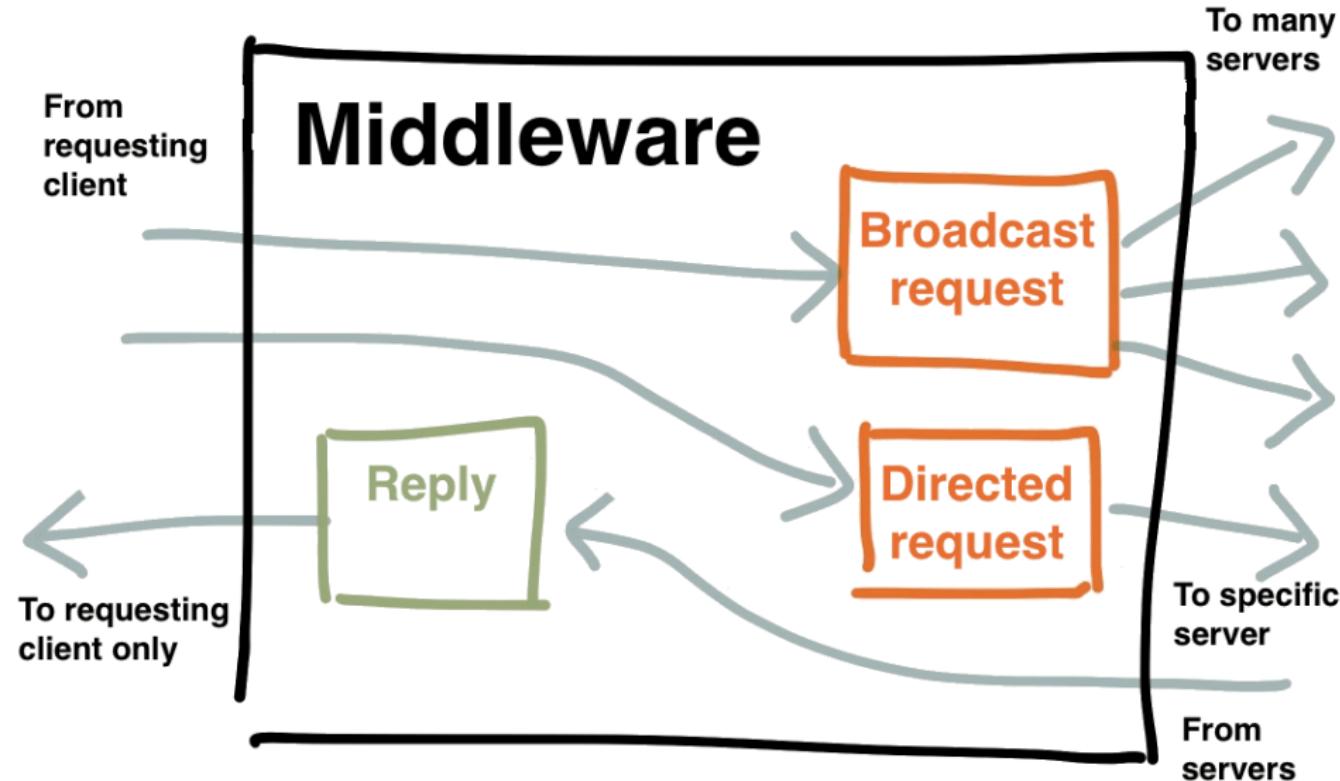
Pros

- Scales with the number of hosts in production.
- Extendable via plugins.
- Doesn't require system accounts, SSO on production hosts.
- Puppet module available for servers.

Cons

- More complex architecture.
- Requires middleware.
- Scaling beyond certain size requires tuning.
- Middleware should be fault-tolerant.
- Misconfigured setups can generate excessive traffic.





First things first

- Clone my repos: <http://github.com/rydnr/dockerfile> and <http://github.com/rydnr/dry-wit>
- Take <http://github.com/rydnr/acmsl-jenkins-configs> as a template for **get-new-version** job.
- Build your custom Delivery Pipeline.
- Make Jenkins generate Docker images and push them to a private index.

Customize and test

- Build mcollective-client and mcollective-server images.
- Install shipyard and mcollective server agent in a test environment.
- Launch docker containers from the mcollective client, via mcollective shell agent.
- Try Interlock in the path to Continuous Deployment!