Docker-based Build Farm for ROS

Tully Foote (OSRF), Dirk Thomas (OSRF), Dejan Pangercic (Bosch),
Daniel Di Marco (Bosch), Arne Hamann (Bosch)

Developed by OSRF, sponsored and tested by Bosch.

Custom Build Farm - Motivation

What does it do?

- automatically build .deb files from your packages in order
- continuous integration (unit tests)
- autodocumentation (doxygen, sphinx, epydoc, ...)

OSRF & GitHub are awesome, why would you want your own build farm?

- host your code on your own servers (i.e. you don’t want to or are not allowed to use public github)
- distribute your proprietary ROS packages (only) to customers
- keep specific package versions (e.g. for stability)

Why it is better (than the old build farm)

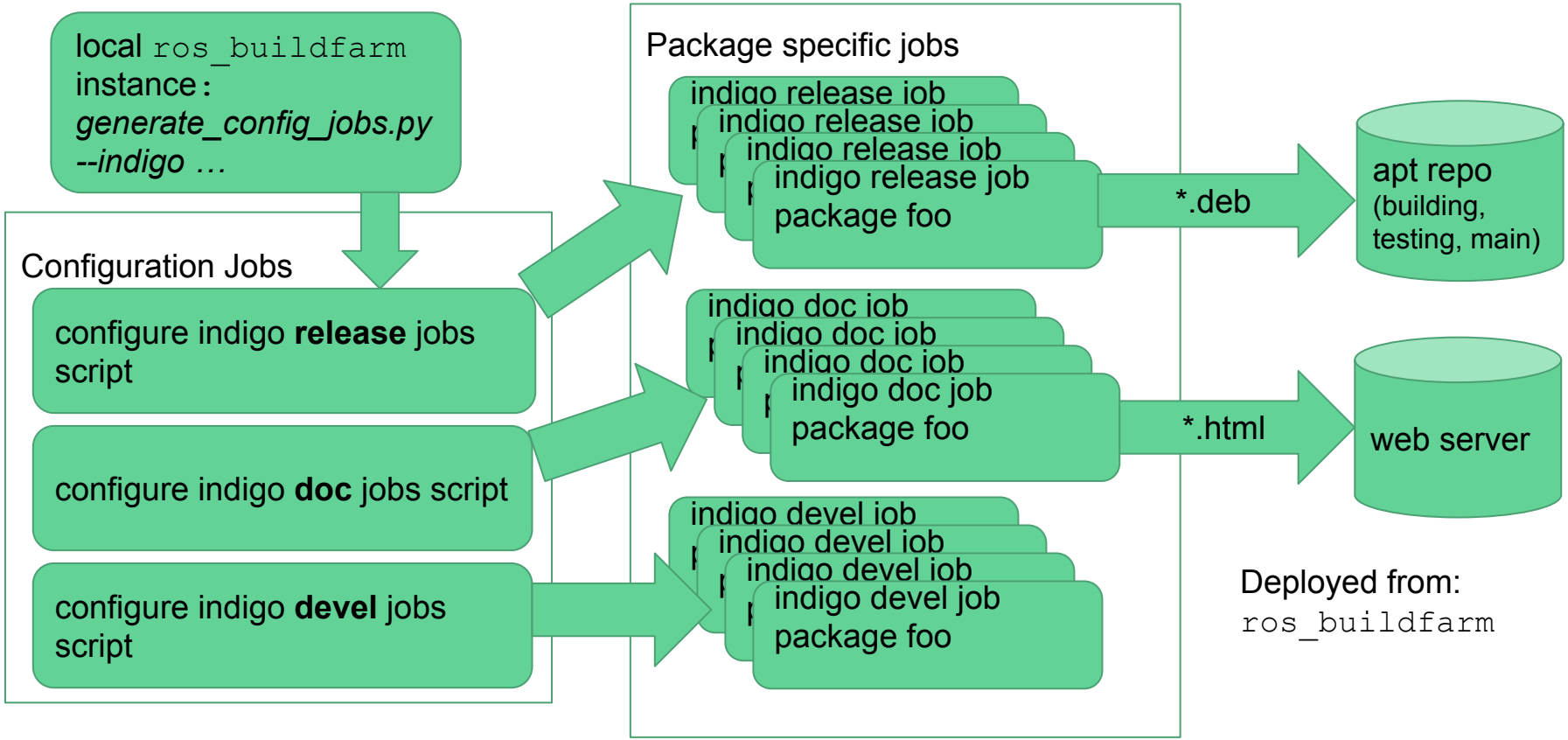
- perfectly reproducible builds, also in parallel (thanks to Docker)
  - also locally on your dev machine (pre-release jobs)
- allow hosting source code on non-public servers
- scripted deployment & update
  - old build farm installation was not reproducible
- simplify deployment of custom setups, more customization options
- black/white-listing packages
- build non-catkin packages

Overview - Hardware

Jenkins Master	Jenkins Slave (n instances)	Repository/Web Host
 Jenkins **Master + Plugins** 	 Jenkins **Slave** 	 Apache  **Reprepro**

Deployed from `buildfarm_deployment`

Overview


local ros_buildfarm instance:
generate_config_jobs.py --indigo …
Configuration Jobs
configure indigo release jobs script
configure indigo doc jobs script
configure indigo devel jobs script
Package specific jobs
indigo release job
indigo release job
indigo release job
indigo release job
package foo
indigo doc job
indigo doc job
indigo doc job
indigo doc job
package foo
indigo devel job
indigo devel job
indigo devel job
indigo devel job
package foo
*.deb
*.html
apt repo
(building, testing, main)
web server
Deployed from:
ros_buildfarm

Jenkins Jobs

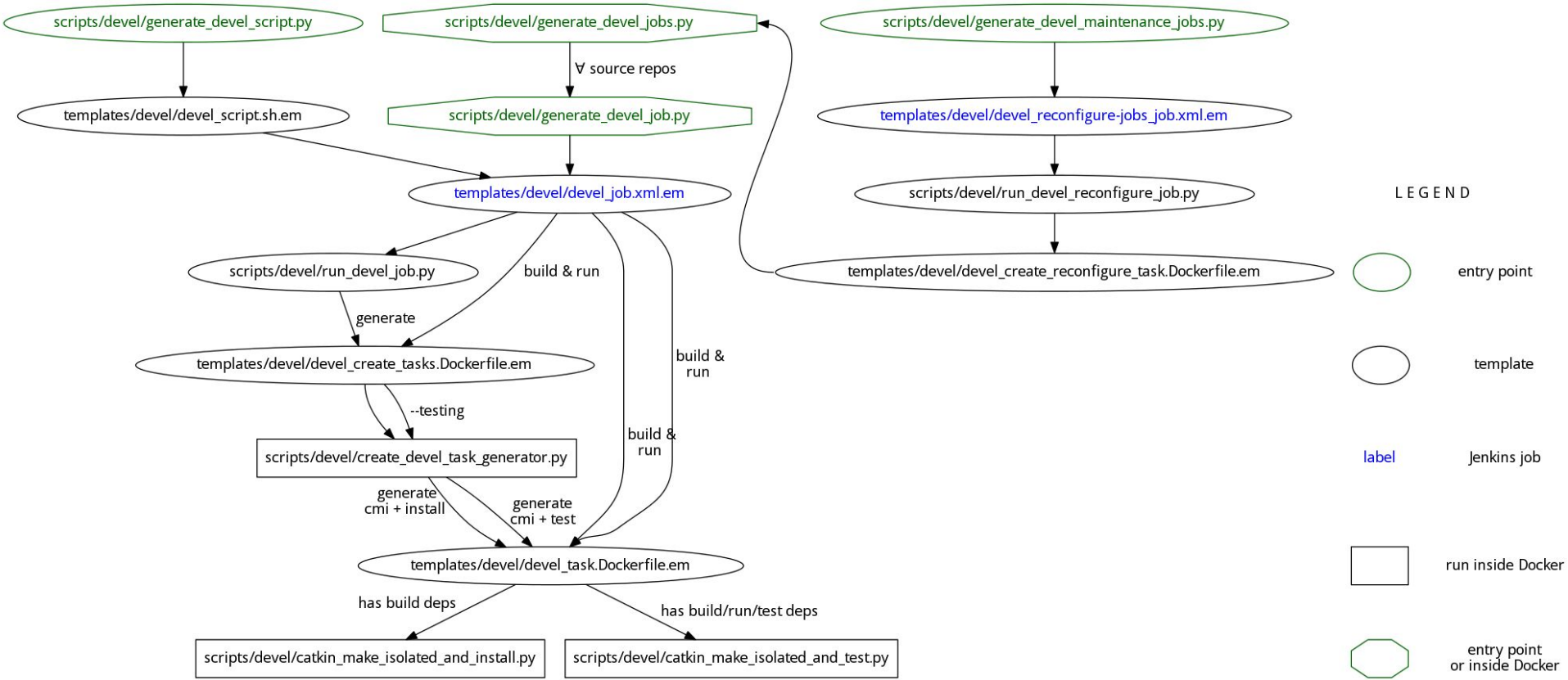
Management jobs:

- **rosdistro cache**: recreate binary rosdistro cache
- **import_upstream**: call reprepro-updater to fetch upstream .deb packages
- **check_slaves**: check disk space on jenkins slaves
- **release-status-page**: create overview page on the repo server
- ***-reconfigure/-trigger-jobs**: update/run build jobs
- **sync-packages**: move packages from building to testing

Build jobs:

- **devel**: build & run tests
- **release**: build binary .debs
- **source**
- **doc**

Devel Job - Components Interaction



Initial Setup (Deployment)

deployment: “bootstrap” your build farm environment, only done once (ROS agnostic)

- fork & adapt ros_buildfarm_deployment_config
  - insert information about your servers, ssh keys, jenkins login
- check it out on your servers, run
  - `./install_prerequisites.bash`; `./reconfigure.bash master | slave | repo`

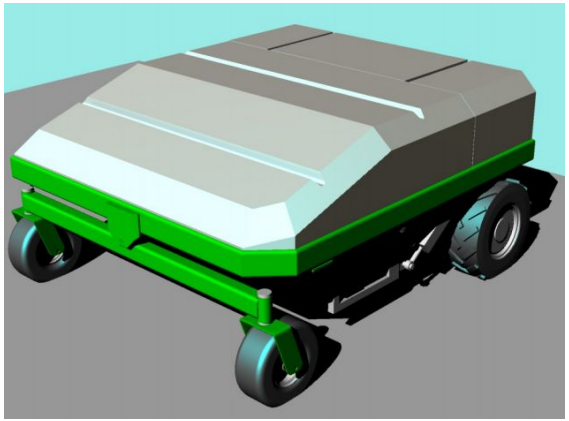
jenkins job configuration:

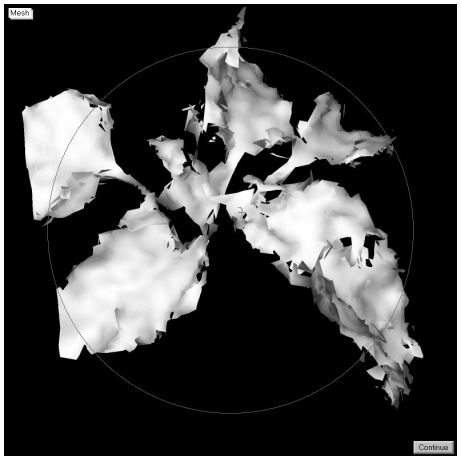
- fork & adapt ros_buildfarm_config
  - create your own distribution, e.g. deepfield-indigo
- if you want to build on top of existing packages:
  - fork & adapt rosdistro
  - add your distribution, list of packages (as generated by bloom-release)

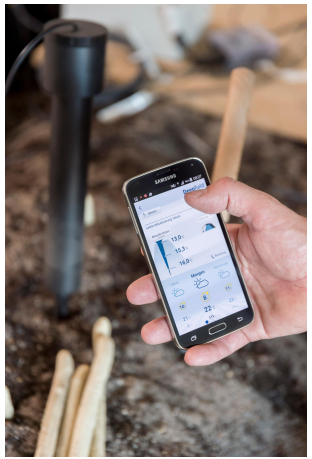
```
distributions:
  indigo:
    doc_builds:
      default: deepfield-indigo/doc-build.yaml
      released-packages-without-doc-job:
        deepfield-indigo/doc-released-build.yaml
    notification_emails:
      - daniel.dimarco@de.bosch.com
    release_builds:
      default: deepfield-indigo/release-build.yaml
    source_builds:
      default: deepfield-indigo/source-build.yaml
```

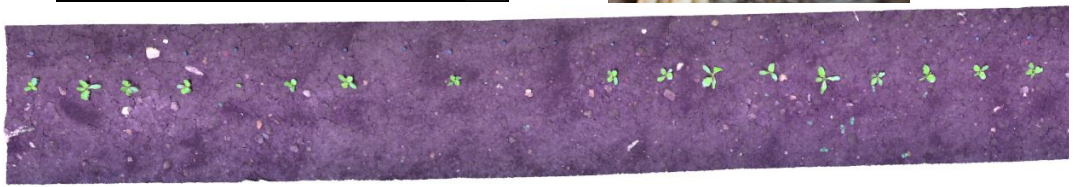
About **Deep**Field Robotics

- Corporate Start-Up within Robert Bosch GmbH
- ~ 20 people with robotics and/or agricultural background









Our Use Case

- our agricultural robot BoniRob is fully ROSified
- deliver proprietary ROS packages (& updates) to customers

Until recently:

- `catkin_make install`
- `tar -cf …`

➜ cumbersome, easy to mess up updates

better: `apt-get install ros-indigo-deepfield`

Overview - DF Setup

Setup:

- 3 VMs on a local server: jenkins-master, jenkins-slave, repository
- custom built proprietary drivers added as rosdeps
- external webserver (Google Cloud) where docs, rosdistro, repositories are mirrored
  - accessible via https, one set of credentials per user
- additional jenkins server for testing, static code analysis

Challenges:

- custom rosdep packages
  - i.e. drivers we must not make publicly available
- private repositories on GitHub Enterprise (and Atlassian Tools?)

rosdistro Custom Dependencies

If your source code builds on standard ROS packages:

- by default, buildfarm builds all packages from source
- so, dependencies in the package.xml can be resolved
- but we don't have e.g. roscpp source packages & just want to use the pre-built pkgs from OSRF

➡ Use Mike Purvis' rosdep-generator[1]:

- generates rosdep files for OSRF buildfarm pkgs (i.e. map rospack name to debian name)
- e.g.

```
actionlib: {fedora: ros-indigo-actionlib, ubuntu: ros-indigo-actionlib}
actionlib_msgs: {fedora: ros-indigo-actionlib-msgs, ubuntu: ros-indigo-actionlib-msgs}
```

- put the resulting rosdep files in your rosdistro repository

[1] https://github.com/mikepurvis/rosdep-generator

Private Repos & GitHub Enterprise

Just replace github.com with your enterprise instance in buildfarm_deployment_config

Still some assumptions wrt. code hosting platform (i.e. public GitHub):

- unauthenticated downloads from `raw.github.com`: rosdistro, buildfarm_config
  ➜ put buildfarm_config, rosdistro on repo web server
- checkouts from public readable git repositories
  - create OAuth tokens
  - put into checkout url in rosdistro distribution
- open pull requests (bloom-release)
  - adjust rosdistro manually

Caveats

Some steps are not automated & need to be triggered manually:

- trigger import_upstream when new upstream packages are released
- sync packages from testing to main

“docker pull” hangup (v 1.6.2)

- update to 1.8.2 seems to have fixed this

A good overview over the tools involved is (highly) recommended:

- Puppet
- Jenkins & Groovy scripting
- git-buildpackage
- Docker
- bloom

Misc. Questions for Discussion

- when does OSRF switch to the new build farm?
- difference to buildbot-ros (bird-eye view)
  - scalability
- users (that we know of):
  - Fraunhofer IPA
  - Yujin
  - Bosch
  - Aldebaran

Documentation

Wiki Instructions

- http://wiki.ros.org/buildfarm

Example Jenkins and Repository instances

- http://54.183.65.232/
- http://54.183.26.131:8080/

Mailing List

- https://groups.google.com/forum/#!forum/ros-sig-buildfarm