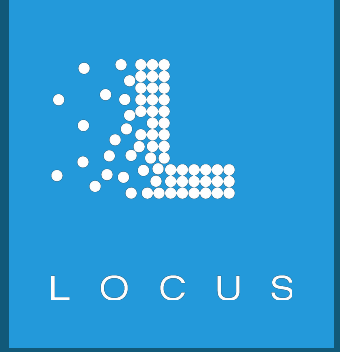
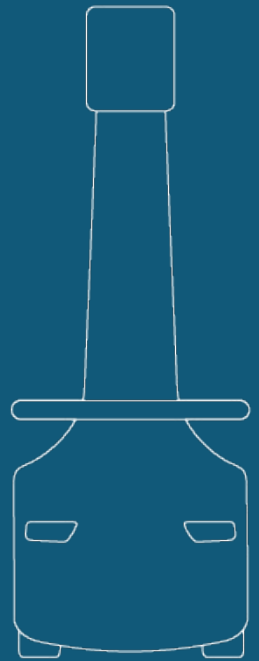


Tailor CI: How Locus ships robots at scale

Paul Bovbel

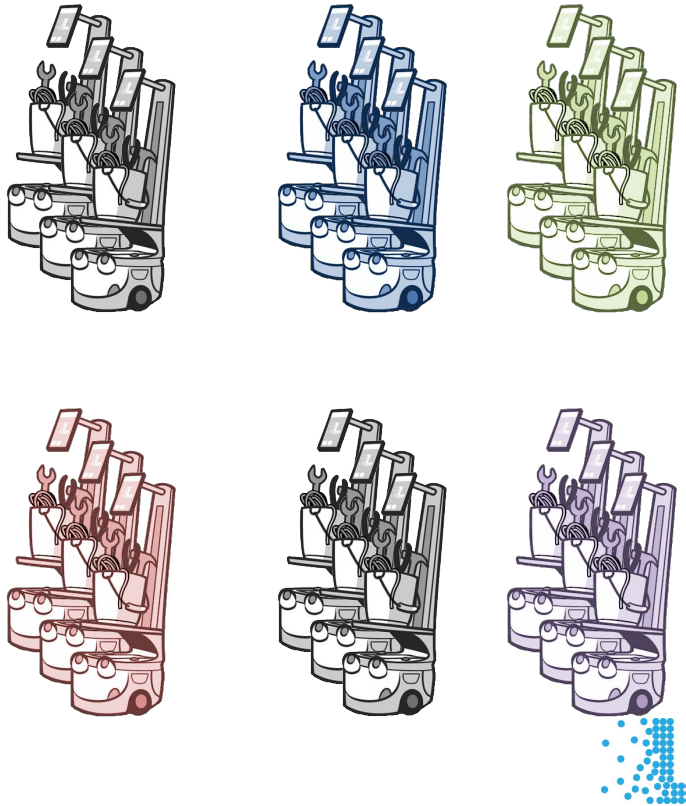
Gary Servin



What does scale mean anyway?

- Some number of sites `A`
- Some number of robots `B`
- Some variety of software releases `C`

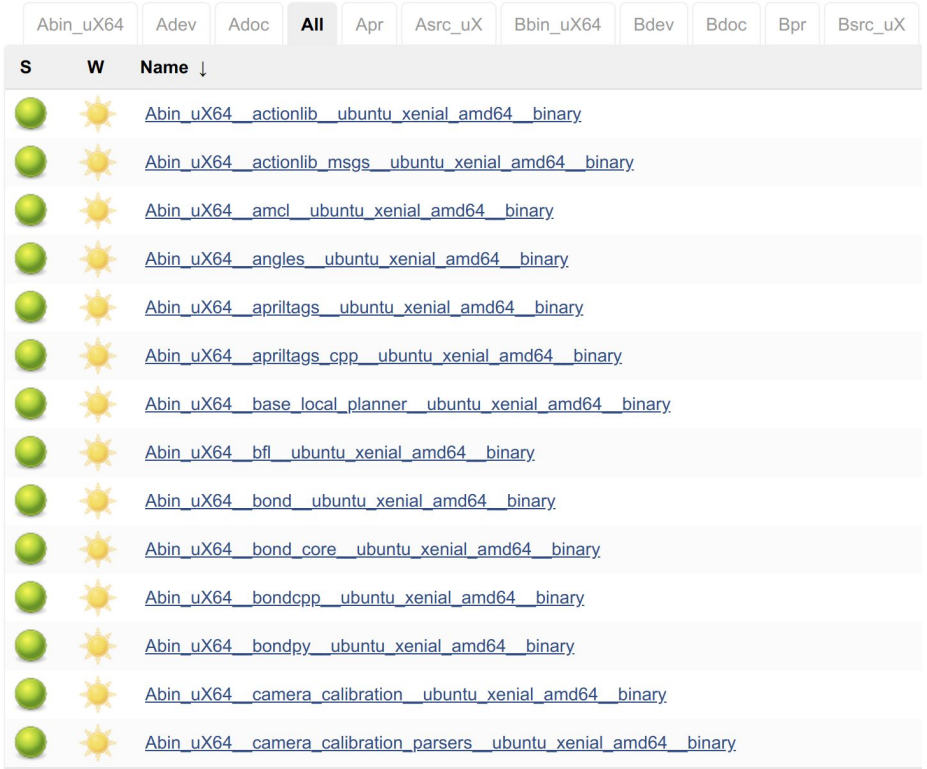
```
madness ~= O(A * log(B) * C)
```



The Old Way

Some combination of

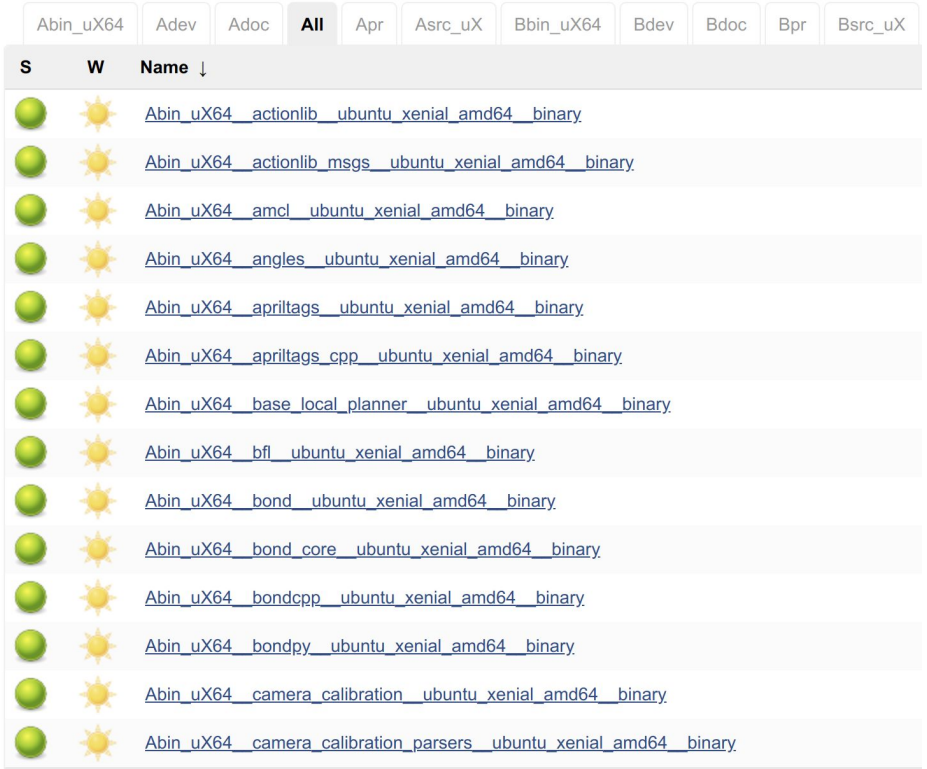
- packaging via ros_buildfarm
- incremental deploys via ansible
- source overlays



The Old Way

Downsides:

- no rolling release or nightlies
- slow top-to-bottom builds
- release state is distributed amongst many repositories
- incremental deploys are flaky

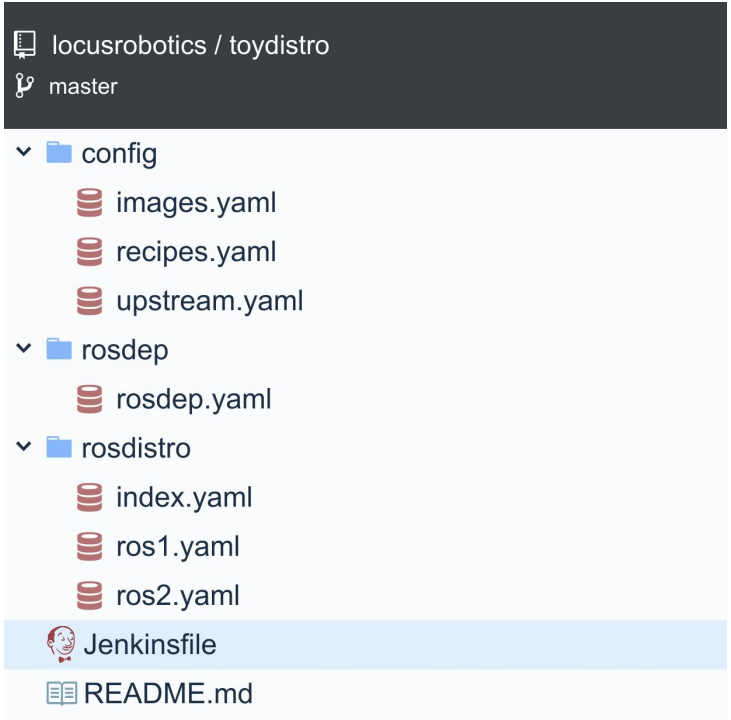


The New-ish Way!

Let's…

- build and package all our sources into one artifact.
- generate images for our targets containing this artifact
- build a nightly from the tip of every package's `master` branch.
- store all release state in one source of truth rather than multiple repositories.
- branch a release off any point in the source of truth.

Top-Level Configuration

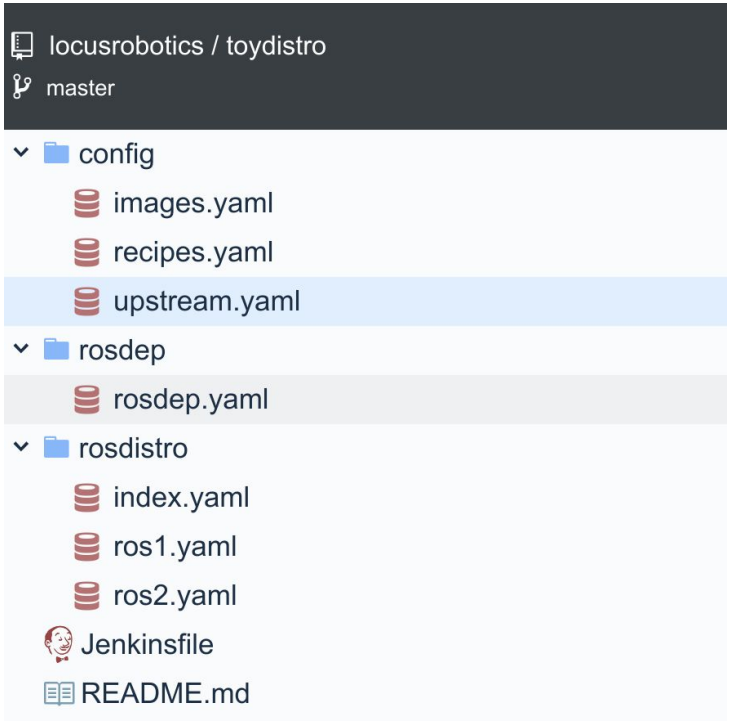



```
#!/usr/bin/env groovy
versions = [
  tailor_meta: "0.1.12",
  tailor_distro: "0.1.15",
  tailor_image: "0.1.26",
]


library('tailor-meta@' + versions['tailor_meta'])_
tailorDistroPipeline(
  versions: versions
)
```

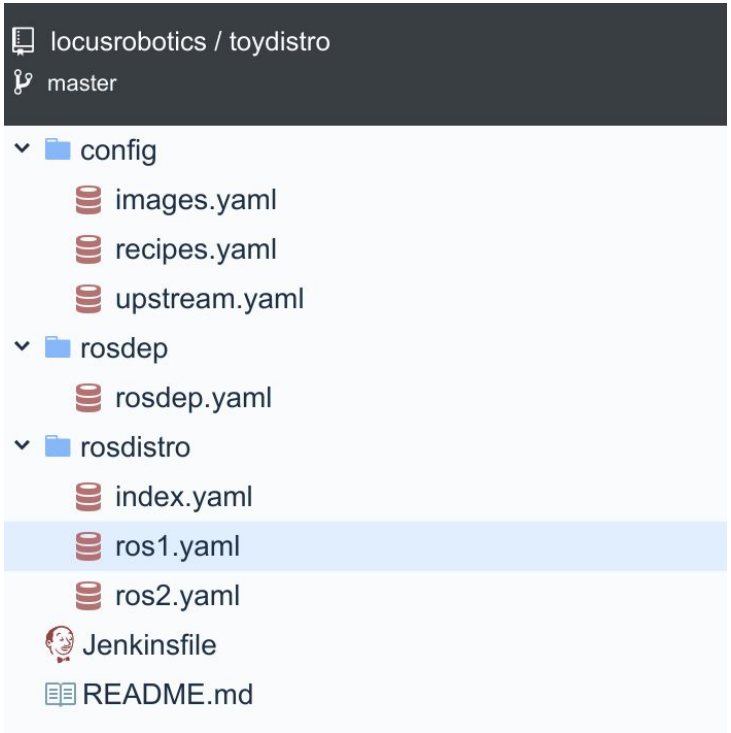
Tailor CI Pipeline

Mirror Configuration



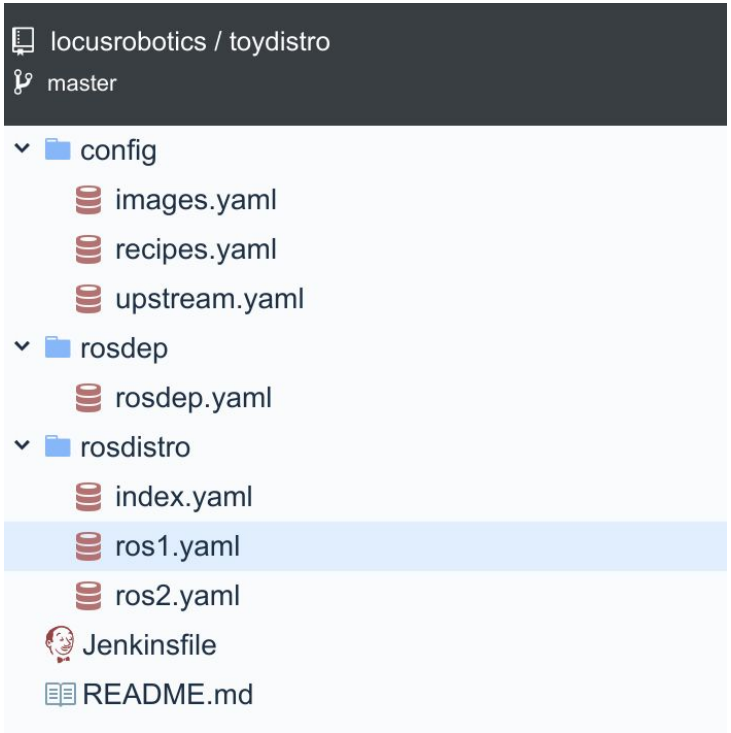

```yaml
---
mirrors:
  ros:
      url: http://repositories.ros.org/ubuntu/main/
      distributions:
        - {{ distribution }}
      components:
        - main
      keys:
        - C1CF6E31E6BADE8868B172B4F42ED6FBAB17C654
      filters:
        - Name (% python*-catkin-pkg)
        - Name (% python*-catkin-tools)
        - Name (% python*-rosdep)
        - Name (% python*-vcstool)
        - Name (% python-pyside* ) | Name (% libpyside* ) | Name (% pyside* )
        - Name (% libshiboken* ) | Name (% shiboken* )
```

Sources Configuration



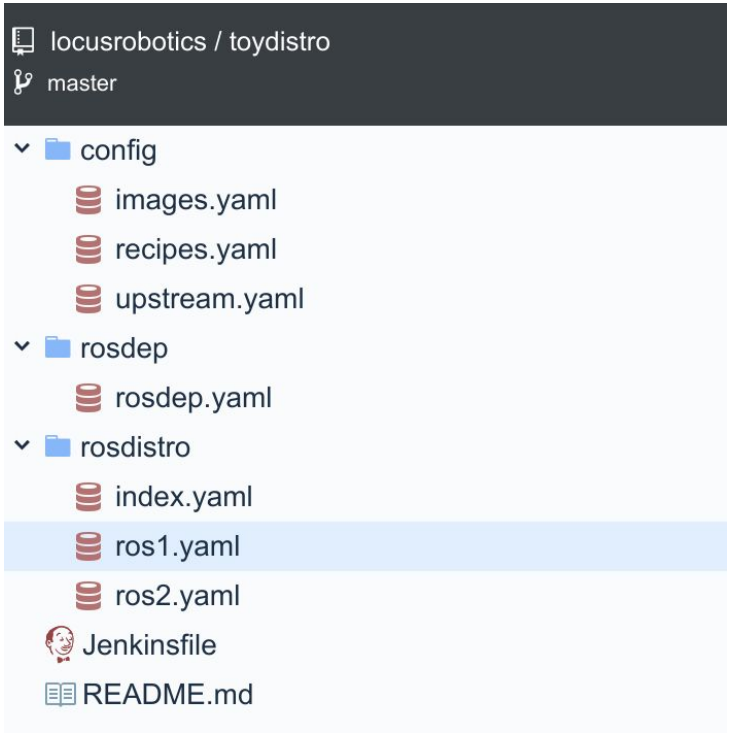

```yaml
---
repositories:
 secret_sauce:
   source:
     type: git
     url: https://github.com/locusrobotics/secret_sauce.git
     version: devel
     test_commits: true
```

Sources Configuration



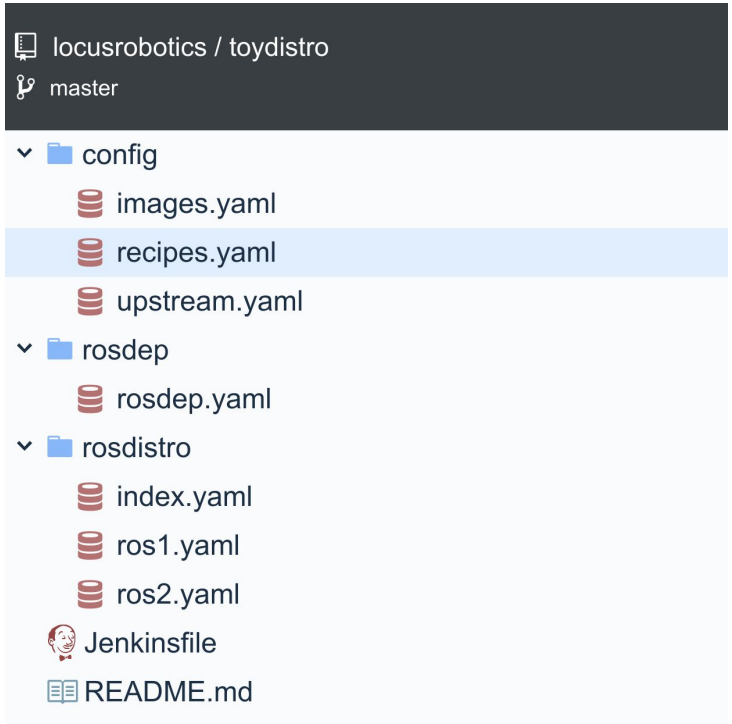

```yaml
---
repositories:
 secret_sauce:
   release:
     tags:
       release: '{{ version }}'
     url: https://github.com/locusrobotics/secret_sauce.git
     version: 19.10.0
   source:
     type: git
     url: https://github.com/locusrobotics/secret_sauce.git
     version: release/19.10
     test_commits: true
```

Sources Configuration



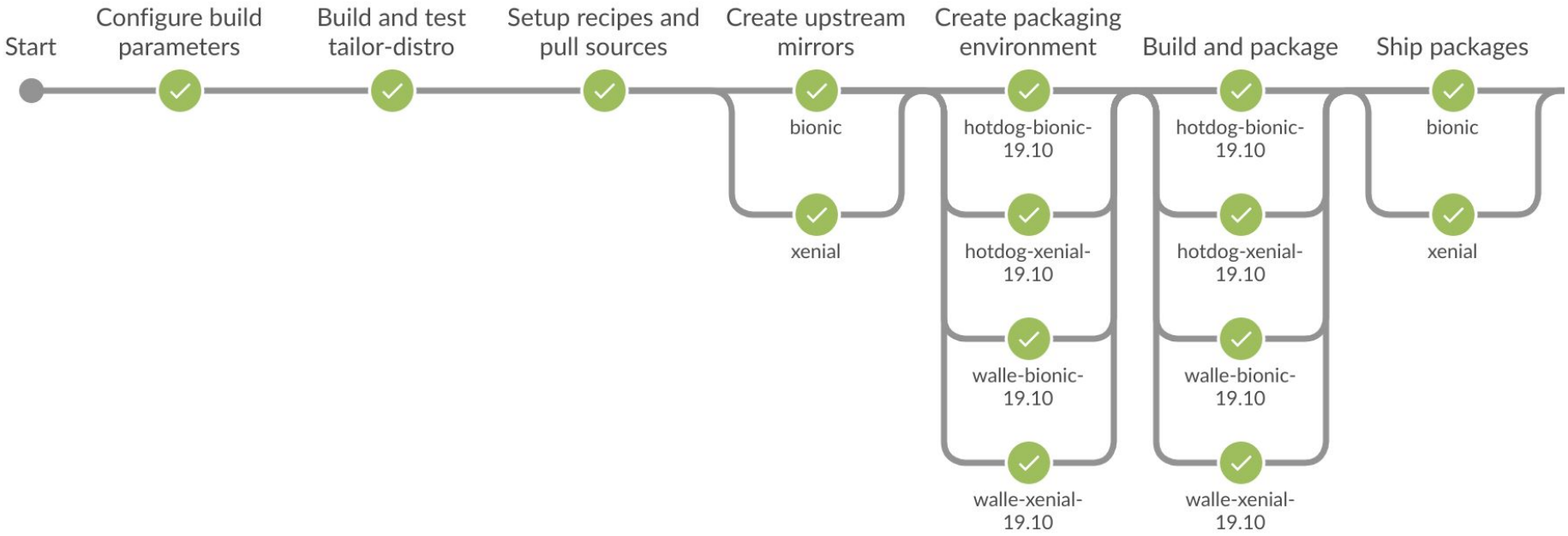

```yaml
---
repositories:
 fastrtps:
   release:
     tags:
       release: release/{{ upstream }}/{{ package }}/{{ version }}
     url: https://github.com/ros2-gbp/fastrtps-release.git
     version: 1.7.2-0
   source:
     type: git
     url: https://github.com/eProsima/Fast-RTPS.git
     version: master
```

Packaging Configuration

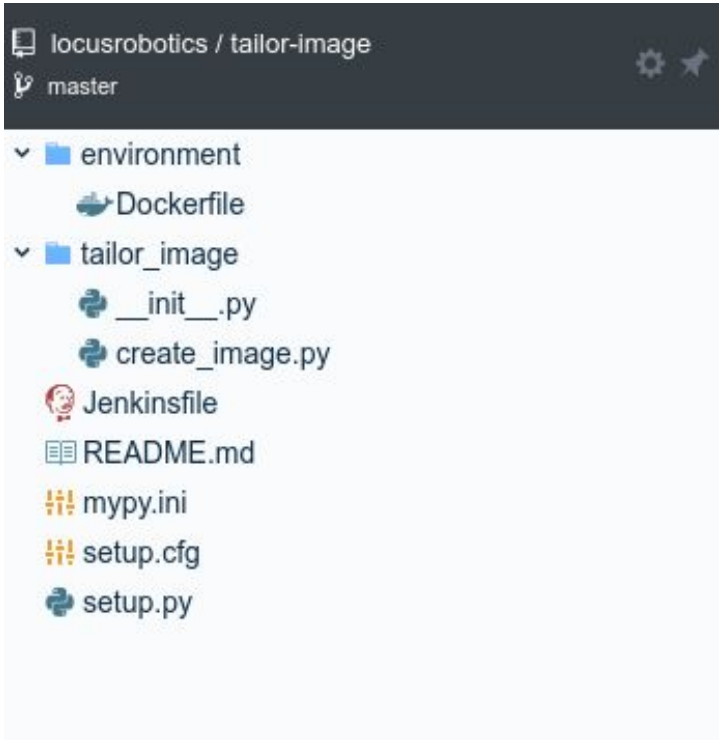



```yaml
---
flavours:
  walle:
    description: mobile trash compactor
    distributions:
      ros1:
        root_packages:
          - diff_drive_controller
      ros2:
        root_packages:
          - heart_of_gold
os:
  ubuntu:
    - xenial
    - bionic
common:
  organization: locusrobotics
  cxx_standard: 14
  cxx_flags:
    - DNDEBUG
```

Packaging Pipeline

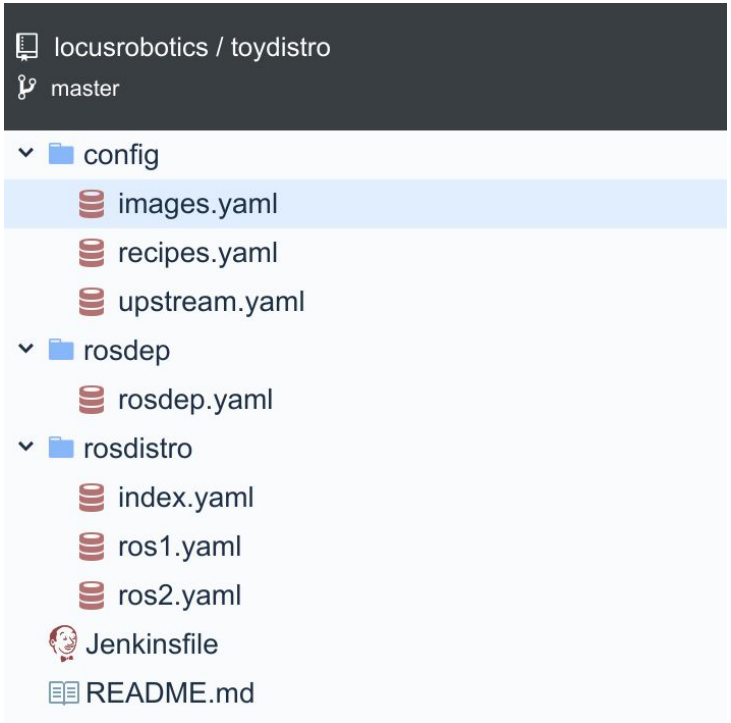


tailor-image



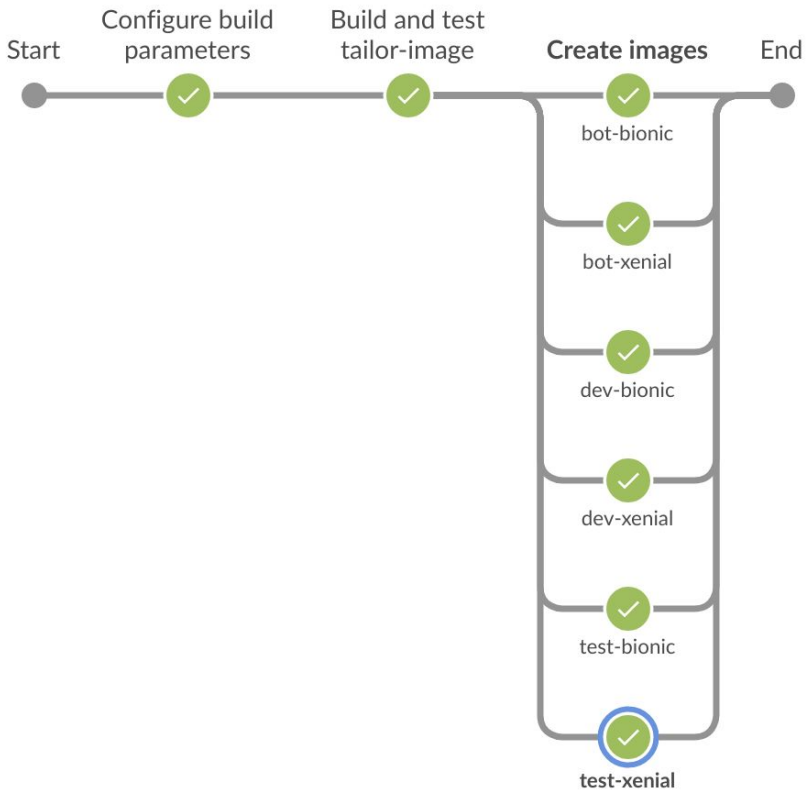
- Python code to generate image artifacts
- Reads configuration from a yaml file
- Currently supports:
  - docker images for test pipelines and developer images
  - bare metal images for robots
- Uses packer to build images
  - Bare metal image build uses packer with the qemu-chroot builder to speed up the image creation process

Image Configuration



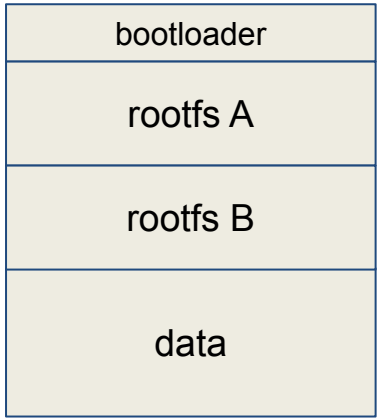

```yaml
---
images:
 dev:
   build_type: docker
   distro: ros1
   package: robot_deploy
   provision_file: developer.yaml
   description: Image for off-board testing and development.
 bot:
   build_type: bare_metal
   distro: ros1
   package: robot_deploy
   provision_file: bare_metal.yaml
   base_image: $distribution-server-cloudimg-amd64-disk1.img
   description: Image for production robots.
```

Image Pipeline



How to consume images

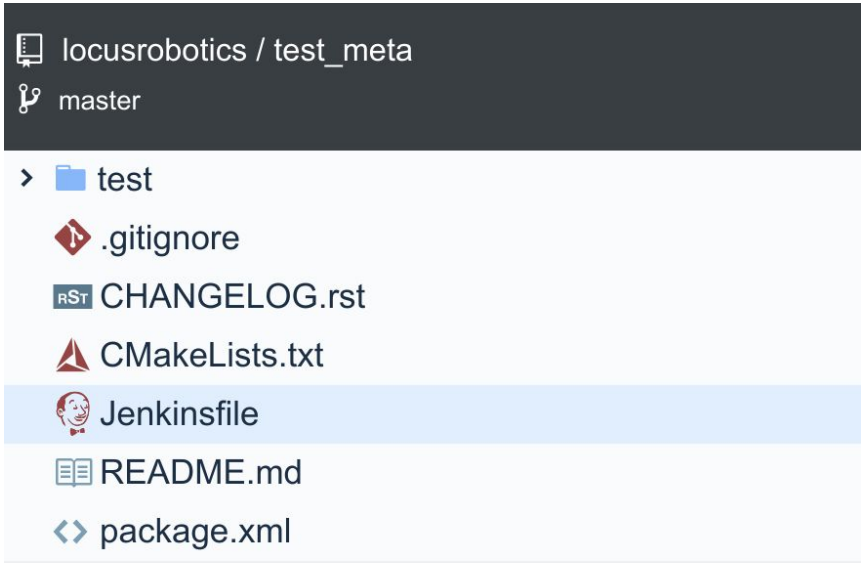
- flash image directly to a hard drive
  - Useful for production
- OTA (Over-The-Air) updates
  - Used by Android, embedded systems, etc
  - A/B partition scheme for the root filesystem
  - Persistent data partition overlayed on top of the current active partition
  - Allows to rollback to a previous working version
  - Allows to run the update as a background process



tailor-meta

- In charge of creating/setting pipelines
- Setups build PRs tests for managed repos (optional)
- Handles updating github settings (optional)
  - Disables wiki
  - Disables projects
  - Only allow squash merge
  - Protect branches and set minimum number of reviewers

PR Builds




```
#!/usr/bin/env groovy
@Library('tailor-meta@0.1.12 ')_
tailorTestPipeline(
 // Name of job that generated this test definition.
 rosdistro_job: '/ci/toydistro/master',
 // Distribution name
 rosdistro_name: 'ros1',
 // Release track to test branch against.
 release_track: '19.10',
 // Release label to pull test images from.
 release_label: '19.10-rc',
 // OS distributions to test.
 distributions: ['xenial', 'bionic'],
 // Branch of tailor_meta to build against
 tailor_meta_branch: '0.1.12',
 // Master or release branch associated with this track
 source_branch: 'master',
 // Docker registry where test image is stored
 docker_registry: 'https://us-east-1.amazonaws.com/locus-toydistro'
)
```

In Closing

- The entire CI pipeline is available for anyone to use:
  - https://github.com/locusrobotics/toydistro (sample distribution)
  - https://github.com/locusrobotics/tailor-distro
  - https://github.com/locusrobotics/tailor-image
  - https://github.com/locusrobotics/tailor-meta
- Built around GitHub API, AWS Services (S3, ECS, EC2)
  - Happy to take patches to support other VCS systems or cloud providers.
- Send us feedback!
  - Better documentation in progress - will publish on Discourse.

Questions?

Management tools

```
# Checkout a release branch for the toydistro repository
cd ~/toydistro
git checkout -b release/19.10

# Gather all unpinned packages
packages=$(tailor_manage query --distro ros1 --unpinned)

# Run catkin_generate_changelog and catkin_prepare_release on all unpinned repos
tailor_manage release --distro ros1 --release 19.10 $packages

# Push release, this builds toydistro-walle-19.10-rc
git push -u origin release/19.10

# Tag release, this builds toydistro-walle-19.10.0
git tag -a 19.10.0 -m 19.10.0
git push --tags
```