Developing and Operating MyTardis at Monash University

James Wettenhall and Manish Kumar
Monash eResearch Centre
https://www.monash.edu/eresearch

In collaboration with:
Amr Hassan, Wojtek Goscinski, Samitha Amarapathy, Keith Schulze, Chaitanya Manapragada, Grischa Meyer, Steve Androulakis, Jason Rigby, Andrew Perry, Anitha Kannan, Lance Wilson, Andrew Mehnert, Andrew Janke
Objectives

Developing MyTardis

MyTardis Codebase
Contributing to Mytardis
Maintaining MyTardis
Continuous Integration

Operating MyTardis

Infrastructure as code
Managing Mytardis deployment at Monash
Developing MyTardis at Monash University

MyTardis

- Open source project hosted at https://github.com/mytardis/mytardis
- Licensed under GPL V3.0
- Latest release is v3.9.0
- Documentation http://mytardis.readthedocs.org
  - User documentation
    - Organising data, Accessing data, sharing and publishing data
  - Administrator documentation
    - Installation, Settings, Migration, enable/disable apps
  - Developer documentation
    - Architecture, Tests, source code
Developing MyTardis at Monash University

Contributing to MyTardis

• Feature requests
• Reporting Bugs
  • https://github.com/mytardis/mytardis/issues
• Submitting Pull Request
  • https://github.com/mytardis/mytardis/blob/develop/CONTRIBUTING.rst
• Help users and contributors via Slack/Github
  • https://mytardis.slack.com/
Developing MyTardis at Monash University

Maintaining MyTardis

Automated Python Security and Dependency Updates
Developing MyTardis at Monash University

Maintaining MyTardis

Automated Python Security and Dependency Updates

- [https://pyup.io/](https://pyup.io/)
Enforcing Coding Standard and error detection

- https://www.pylint.org/
Developing MyTardis at Monash University

Maintaining MyTardis

Code review and code quality monitoring
  • Codacy https://www.codacy.com/
Developing MyTardis at Monash University

Maintaining MyTardis

Code review and code quality monitoring
Developing MyTardis at Monash University

Maintaining MyTardis

Code review and code quality monitoring
Developing MyTardis at Monash University

Maintaining MyTardis

Automated testing
  • Unit Tests
    • ./test.py
Automated testing

- Unit Tests
  - ./test.py
Developing MyTardis at Monash University

Maintaining MyTardis

Automated testing

- BDD
  - ./test.py behave
Developing MyTardis at Monash University

Maintaining MyTardis

Automated testing
  • BDD
    • ./test.py behave
Developing MyTardis at Monash University

Maintaining MyTardis

Automated testing
  • Qunit tests
    • Npm test
Developing MyTardis at Monash University

Maintaining MyTardis

Automated testing
  • Qunit tests
  • Npm test
Developing MyTardis at Monash University

Maintaining MyTardis

Test coverage

• [https://coveralls.io/](https://coveralls.io/)
Continuous Integration

“Continuous Integration doesn’t get rid of bugs, but it does make them dramatically easier to find and remove.”

— Martin Fowler, Chief Scientist, ThoughtWorks
Developing MyTardis at Monash University

Continuous Integration

• How we do it
  • Developers check out code into their private workspaces
  • When done, commit the changes to the repository
  • The CI server monitors the repository and checks out changes when they occur
  • The CI server builds the system and runs unit and integration tests
Continuous Integration

- How we do it
  - The CI server releases deployable artefacts for testing
  - The CI server assigns a build label to the version of the code it just built
  - The CI server informs the team of the successful build
  - If the build or tests fail, the CI server alerts the team
  - The team fixes the issue at the earliest opportunity
  - Continue to continually integrate and test throughout the project

Developing MyTardis at Monash University
Continuous Integration

- **Semaphore**
  - PR submitted
  - Codacy check
  - Semaphore build
  - PR ready to be merged
Developing MyTardis at Monash University

Continuous Integration

- **Semaphore**
  - PR submitted
  - Codacy check
  - Semaphore build
  - PR ready to be merged
Developing MyTardis at Monash University

Continuous Integration

- **Semaphore**
  - PR submitted
  - Codacy check
  - Semaphore build
  - PR ready to be merged
Developing MyTardis at Monash University

Continuous Integration

- **Semaphore**
  - PR submitted
  - Codacy check
  - Semaphore build
  - PR ready to be merged
Continuous Integration

- Semaphore
Developing MyTardis at Monash University

MyTardis Release process

- Minor release (Monthly)
- Major release (as needed)
- Security and patch release (anytime within release cycle)

<table>
<thead>
<tr>
<th>Day of the Month</th>
<th>Tasks</th>
</tr>
</thead>
<tbody>
<tr>
<td>8th Feature Freeze day:</td>
<td></td>
</tr>
<tr>
<td>• Announce creation of series-x.y branch and feature freeze on #general MyTardis Slack channel</td>
<td></td>
</tr>
<tr>
<td>• Create meta issue to track release on <a href="http://github.com/mytardis/release">http://github.com/mytardis/release</a> issues</td>
<td></td>
</tr>
<tr>
<td>• Merge any final Pull Requests labelled with this milestone into develop that have been approved (i.e., reviewed and have approval status).</td>
<td></td>
</tr>
<tr>
<td>• Create series-x.y branch from develop</td>
<td></td>
</tr>
<tr>
<td>• Tag and release X.Y.0-rc1</td>
<td></td>
</tr>
<tr>
<td>14-24th Tag and release subsequent RCs as needed. Generally should aim for at least a couple. (as needed)</td>
<td></td>
</tr>
<tr>
<td>22nd Release day:</td>
<td></td>
</tr>
<tr>
<td>• Cherry-pick any final patches for features in this release that have been merged into develop.</td>
<td></td>
</tr>
<tr>
<td>• Ensure all tests on series-x.y are passing on Semaphore</td>
<td></td>
</tr>
<tr>
<td>• Tag X.Y.0 version from series-x.y</td>
<td></td>
</tr>
<tr>
<td>• Check that all tests for tag X.Y.0 pass</td>
<td></td>
</tr>
<tr>
<td>• Create github release</td>
<td></td>
</tr>
</tbody>
</table>
Objectives

Developing MyTardis

- MyTardis Codebase
- Contributing to Mytardis
- Maintaining MyTardis
- Continuous Integration

Operating MyTardis

- Infrastructure as code
- Managing Mytardis deployment at Monash
Operating MyTardis at Monash University

Infrastructure as Code

- Nectar cloud (Based on OpenStack)
- Heat
- Salt
Operating MyTardis at Monash University

Infrastructure as Code

• Nectar cloud (Based on OpenStack)
  • provides
    • flexible scalable computing power
    • computing infrastructure
    • software and services

• Heat
• Salt
Operating MyTardis at Monash University

Infrastructure as Code

• Nectar cloud (Based on OpenStack)
• Heat
  • Heat template
  • Heat create stack
  • Heat update stack
• Salt

```yaml
heat_template_version: 2013-05-23
description: Simple template to deploy a single compute instance
resources:
  my_instance:
    Type: OS::Nova::Server
    properties:
      image: cirros-0.3.3-x86_64
      flavor: m1.small
      key_name: my_key
      networks:
        - network: private-net
```
Operating MyTardis at Monash University

Infrastructure as Code

- Nectar cloud (Based on OpenStack)
- Heat
  - Heat template
  - Heat create stack
  - Heat update stack

```python
deploy_all.py --deployment qat --salt_master_ip=118.138.233.109 --ssh_key_name=cloud-nodes --security_group=open-to-all --celery_n_servers 2 --celeryd_filters_n_servers 2 > qat-store-monash.json
```

```bash
heat stack-create qat-store-monash --template-file qat-store-monash.json
```
Operating MyTardis at Monash University

Infrastructure as Code

- Nectar cloud (Based on OpenStack)
- Heat
  - Heat template
  - Heat create stack
  - Heat update stack
- Salt
Operating MyTardis at Monash University

Infrastructure as Code

• Nectar cloud (Based on OpenStack)
• Heat
• Salt
  • Configuration Management
    • Run commands on managed systems
    • Define re-usable configurations
    • Apply commands and configurations to specific systems
Operating MyTardis at Monash University

Infrastructure as Code

- Salt
  - Configuration Management
    - Run commands on managed systems
    - Define re-usable configurations
    - Apply commands and configurations to specific systems

```
salt -E '.+(celery|web|sftpd).+' cmd.run 'ls -l /home/mytardis'
```
Operating MyTardis at Monash University

Infrastructure as Code

- Salt
  - Configuration Management
    - Run commands on managed systems
    - Define re-usable configurations
      - Salt state
      - Apply commands and configurations to specific systems

Salt `*web*` state.apply
Operating MyTardis at Monash University

Infrastructure as Code
- Nectar cloud (Based on OpenStack)
- Heat
- Salt
  - Configuration Management
    - Run commands on managed systems
    - Define re-usable configurations
  - Apply commands and configurations to specific systems
MyTardis has been supported by:

- [http://www.mytardis.org/](http://www.mytardis.org/)
- [https://github.com/mytardis/mytardis](https://github.com/mytardis/mytardis)
- [https://mytardis.readthedocs.io](https://mytardis.readthedocs.io)
- [store.star.help@monash.edu](mailto:store.star.help@monash.edu)