Continuous Integration and FAIR release management for modular ontologies

Anita Caron, Damien Goutte-Gattat, Nicolas Matentzoglu and Philip Strömer
Logical axioms in ontologies relate entities from diverse ontologies

Hypolysinemia

Quality
PATO

decreased amount

ChEBI
part of

.ũ uberon

Entity
blood

decreased amount

lysine

blood
How do I get terms from external ontologies to re-use them?
How do I make sure that whenever I make a change, I didn’t break anything?
How do I make sure all users can access the ontology in a standardised, FAIR manner?
The ODK is a toolbox and ontology life-cycle management system

docker pull obolibrary/odkfull
Overview

Generate standard git repository

Executable workflows:
- Imports, releases
- Testing
- Uses ODK toolbox

Social workflows:
- Issues
- Pull requests

CI/CD:
- Automatic testing
- Automatic updates
- Diffs

GitLab

GitHub Actions

Travis CI
Workflow: Dependency management

- Repeatable workflows ("refresh-uberon")
- Stay always current with the axiomatisation of upstream sources
- Manage small import modules using use-case optimised computational extraction techniques instead of adhoc term import or importing the whole.

Toolbox
- ROBOT
- OAK
- sssom-py
- owltools
Continuous Integration Testing

- No more broken ontologies on “main”
- No more fear you might “break stuff”
- Rich set of checks:
  - OWL profile checking
  - ROBOT report (incl. many best practices)
  - Customisable with SPARQL-based unit testing
  - Logical consistency

Developer

edit locally

Make pull request

CI System (GH actions) runs ODK checks

Some checks were not successful
2 failing and 1 skipped checks

CI/ontology_qc (pull_request) Failing after 3m
Workflow: Release pipeline

- Repeatable workflows (“prepare_release”)
- Robust pipelines to generate standard release files reliably and quickly
- Following OBO Best practices:
  - Standard serialisations
  - Standard release variants (base, full)
  - Versioning
  - Clear separation of editors and release environment
- Fully customisable (release variants, serialisations, base-iris)
Add getting started guides for other onboarding new curators and developers

Provided detailed documentation on how all workflows work (imports, releases etc)

Document design patterns

User documentation

---

3. Merge to main branch
4. Create a GitHub release

These steps are outlined in detail in the following:

Run a release with the ODK

Preparation:
1. Ensure that all your pull requests are merged
2. Make sure that all changes to main are committed (there are no modified files)
3. Locally make sure you have the latest changes
4. Checkout a new branch (e.g. `git checkout`)
5. You may or may not want to refresh your instance of the ODK
6. Make sure you have the latest ODK installed

To actually run the release, you:
1. Open a command line terminal window at `cato/src/ontology`
2. Run release pipeline: `sh run.sh make pr...` this process can take up to 90 minutes - e.g. PRO or CHEBI.
3. If everything went well, you should see the Cato ontologies are now in ..../... - now you should check the hosting site such as GitHub or GitLab.

This will create all the specified release target ontologies and copy them into your release directory.

---

How to run a release?

Design patterns?

SOPs?
Acknowledgements

Core team

- @matentzn Nicolas Matentzoglu (Semanticly)
- @gouttegd Damien Goutte-Gattat (Flybase)
- @anitacaron Anita Caron (EMBL-EBI)
- @balhoff Jim Balhoff (RENCI)
- @cmungall Chris Mungall (LBNL)
- @dosumis David Osumi-Sutherland (EMBL-EBI)
- @ehartley Emily Hartley (Critical Path Institute)
- @hkir-dev Huseyin Kir (EMBL-EBI)
- @shawntanzk Shawn Tan (EMBL-EBI)
- @ubyndr Ismail Ugur Bayindir (EMBL-EBI)

Funding:

Phenomics First (NIH / NHGRI #1RM1HG010860-01)

Paper in Database:
https://arxiv.org/abs/2207.02056

More than 70 documented uses.