1st Conference on Research Data Infrastructure

Poster presentations II

https://doi.org/10.52825/CoRDI.v1i.417

Published: 07 Sept. 2023

Conda, Container and Bots

How to build and maintain tool dependencies in workflows and training materials

Paul Zierep[1][https://orcid.org/0000-0003-2982-388X], Sanjay Kumar Srikakulam[1][https://orcid.org/0000-0002-1752-5060], Sebastian Schaaf[1,2][https://orcid.org/0000-0001-8193-0151], and Björn Grüning[1][https://orcid.org/0000-0002-3079-6586]

[1] Freiburg Galaxy Team, Bioinformatics Group, Department of Computer Science, University of Freiburg, Germany

[2] ELIXIR Officers Team, Institute for Bio- and Geosciences 5, Research Center Jülich, Germany

Abstract. The lifecycle of scientific tools comprises the creation of code releases, packages and containers which can be deployed into cloud platforms, such as the Galaxy Project, where they are run and integrated into workflows. The tools and workflows are further used to create training material that benefits a broad community. The need to organize and streamline this tool development lifecycle has led to a sophisticated development and deployment architecture.

Additionally, it is crucial to keep the tools and their dependencies in sync, as well as to update downstream workflows and training material upon new tool releases. Outdated workflows and training material are disadvantageous to the community, but the effort to keep the tools up-to-date is an immense burden considering the huge amount of tools available. For example, there are more than 3200 tools hosted on the European Galaxy server (https://usegalaxy.eu) [1] as of this writing.

This talk will explain how the Galaxy community is automatizing the updates for tools, packages, containers, workflows, and training materials to lower the maintenance burden of the community.

The Galaxy Project [2] has made thousands of open-source software utilities for scientific data analysis easily accessible to researchers by providing computational infrastructure and a user-friendly web interface. Although Galaxy itself does not require any significant computational skills to use, the development and maintenance of new tools and workflows benefit from sophisticated infrastructure with both human and automated components. The process of integrating software into Galaxy requires knowledge of both the command-line interface of the underlying tool and the schema used by Galaxy to define interfaces in order to be able to write a “Galaxy tool wrapper”. Such a wrapper maps dataset inputs, additional parameters, and expected outputs.

The deployment of software tools envisioned for use in Galaxy is orchestrated by Conda [3]. Conda is a package-, dependency- and environment-manager that enables the distribution of software packages via dedicated channels (e.g. Bioconda, conda-forge). Currently, conda-forge and Bioconda provide more than 21,000 and more than 9,900 tools, respectively.

It is a community-driven project that aims to make it easier to install and manage a wide range of software on different platforms. Conda has several advantages over traditional methods of installing scientific software. It simplifies software installation by automatically managing dependencies and by providing a consistent environment for software development and analysis. The ease of use of Conda packages benefits the scientific community far beyond the Galaxy use cases. The success of this deployment strategy has already led to adaptation outside the bioinformatic field, e.g. in material science [4].

Bioconda [5] extends the basic Conda package management functionality by creating additional Docker and Singularity containers that provide a containerized tool environment [6]. Containerization is further promoted by the automatic generation of so called mulled containers, that provide different Conda dependencies in one container, for the use of tool wrappers that depend on multiple packages. Currently, there are more than 87,000 Docker [7] and Singularity [8] containers available that can be used in cloud infrastructures, HPC-, as well as on local compute environments.

The semi-automated development of the Galaxy tool wrappers is facilitated by the Planemo toolkit [9], which offers a wide range of functionalities for developing, deploying, and executing Galaxy tools, workflows, and training material, with a simple and powerful command-line interface. It also facilitates automated deployment of tools and automatic updates of software dependencies, and offers testing and linting functionality that helps to ensure high-quality tool wrappers and workflows. Although these tasks can be performed individually without Planemo, it provides a convenient single tool that encourages best practices and is already extensively used in the Galaxy ecosystem.

To ensure that tools, wrappers, workflows and the Galaxy servers are up-to-date, a system of bots is maintained that handle automated updates of components to their latest dependencies versions. The bots are based on Planemo as well as Bioconda utilities that are controlled by continuous integration (CI) GitHub pipelines. The demonstrated deployment architecture combines automated update functionality with necessary code-review steps in order to ensure quality controlled tools and wrappers. Updates of Bioconda packages, for example, will trigger pull requests (PRs) in dependent Galaxy tool repositories, but these PRs will require review and approval by expert developers before they can be merged.

Ideally, a bioconda tool update would propagate all the way to the main Galaxy servers, and to community-maintained workflows and training material using these tools.

In this talk, the ecosystem will be explained on the basis of a use case that follows one of the latest tools that were integrated into Galaxy. It will cover the complete tool development, testing and deployment process until its installation on the European Galaxy server and its integration into the Galaxy Training Network (GTN) [10] training material. Furthermore, the update propagation workflows will be demonstrated for this tool.

Keywords: conda, containers, bots, automation, Galaxy

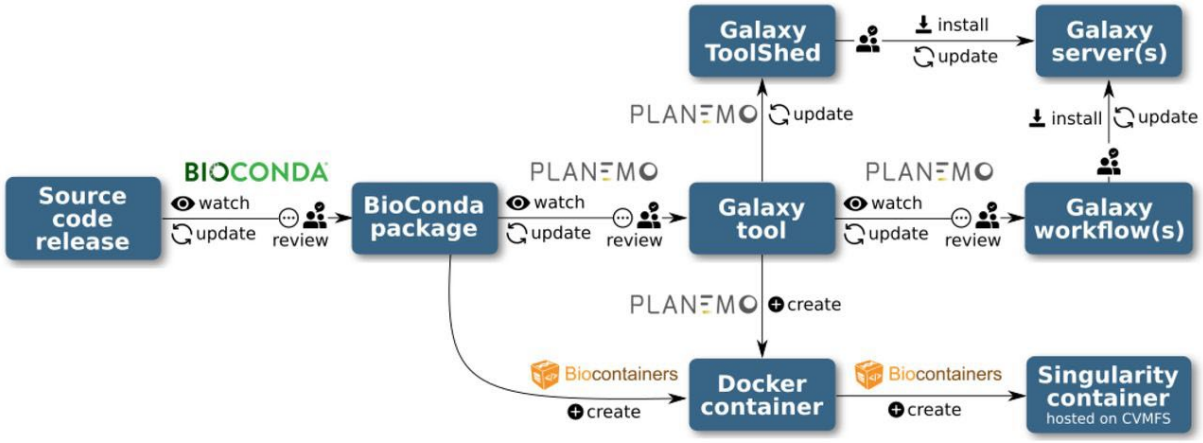

Figure 1. Overview flow diagram.

Competing interests

The authors declare that they have no competing interests.

References

1. "Galaxy Europe." https://usegalaxy.eu (accessed Apr. 24, 2023).
2. "Galaxy platform for accessible, reproducible and collaborative biomedical analyses: 2022 update | Nucleic Acids Research | Oxford Academic." doi: 10.1093/nar/gkac247
3. "Conda — conda documentation." https://docs.conda.io/en/latest/ (accessed Apr. 26, 2023).
4. J. Janssen et al., "pyiron: An integrated development environment for computational materials science," Computational Materials Science, vol. 163, pp. 24–36, Jun. 2019, doi: 10.1016/j.commatsci.2018.07.043.
5. B. Grüning et al., "Bioconda: sustainable and comprehensive software distribution for the life sciences," Nat Methods, vol. 15, no. 7, Art. no. 7, Jul. 2018, doi: 10.1038/s41592-018-0046-7.
6. F. da Veiga Leprevost et al., "BioContainers: an open-source and community-driven framework for software standardization," Bioinformatics, vol. 33, no. 16, pp. 2580–2582, Aug. 2017, doi: 10.1093/bioinformatics/btx192.
7. "Quay Container Registry · Quay." https://quay.io/organization/biocontainers (accessed Apr. 26, 2023).
8. "Index of /singularity/." https://depot.galaxyproject.org/singularity/ (accessed Apr. 26, 2023).
9. S. Bray et al., "The Planemo toolkit for developing, deploying, and executing scientific data analyses in Galaxy and beyond," Genome Res, vol. 33, no. 2, pp. 261–268, Feb. 2023, doi: 10.1101/gr.276963.122.
10. "Galaxy Training," Galaxy Training Network. https://training.galaxyproject.org/training-material/ (accessed Apr. 26, 2023).