Scientific Software Center

IWR Interdisciplinary Center for Scientific Computing

UNIVERSITÄT HEIDELBERG ZUKUNFT SEIT 1386

Effective workflows for community engagement of users and developers in open-source software development

Speaker: **Edwin Carreño**, *Scientific Software Center*

Co-authors: Sebastian Lobentanzer, Inga Ulusoy

ssc.iwr.uni-heidelberg.de

Outline

- Motivation: Research Software as research infrastructure
- An effective workflow (methodology)
  - Onboarding and knowledge transfer
  - Structured contribution
  - Ongoing community engagement
- Summary and key points

Scientific Software Center

Motivation: Research Software as research infrastructure

*"My research software is a **research infrastructure** and needs to be maintained and adapted. There are no resources for this in science.*

Can I engage the community to support my project?"

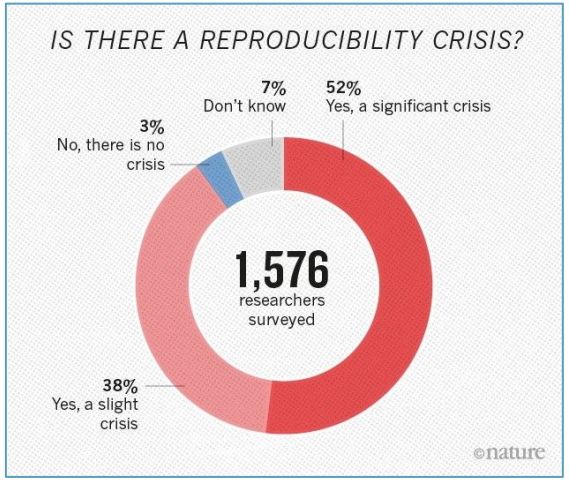


Image from Baker, M. (2016). *1,500 scientists lift the lid on reproducibility*. *Nature* 533, 452–454 (2016). Retrieved from: https://doi.org/10.1038/533452a

It implies:

- Building something functional
- Sharing with others
- Letting others contribute
- Keeping this project alive!

Scientific Software Center

Project Management

Software Engineering

Documentation

People

Tools

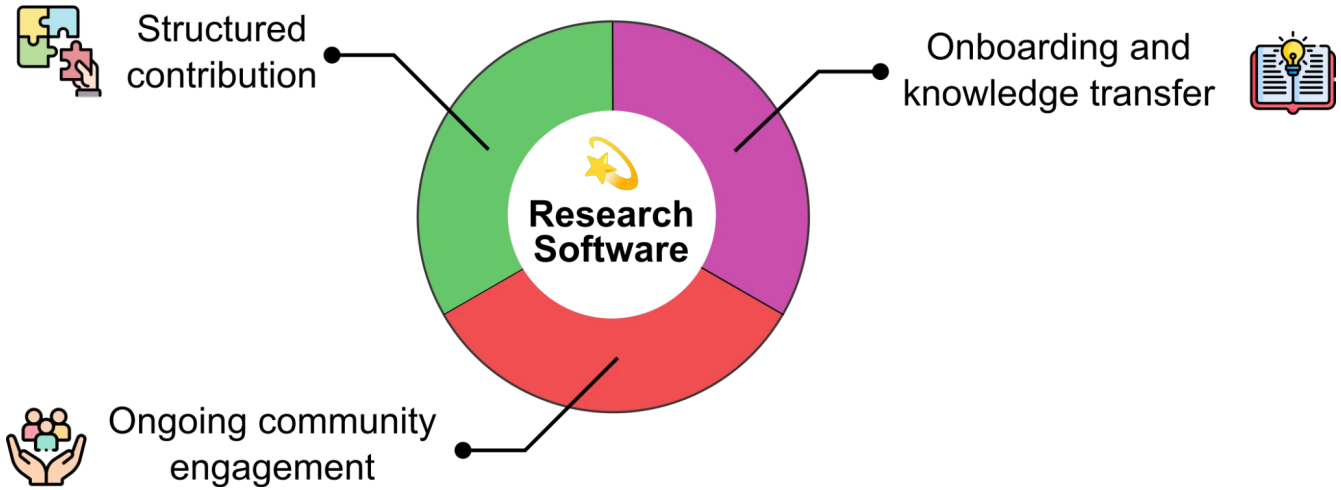
Communication

Software maintenance and usability **is not just writing code**; it's a process that involves orchestrating many aspects– including people!

An Effective Workflow

An Effective Workflow

- It not only takes software development into account but also aligns and coordinates the aspects surrounding the project!



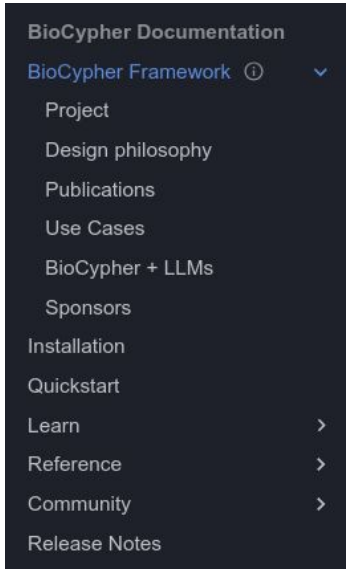
Onboarding and knowledge transfer

Activities:

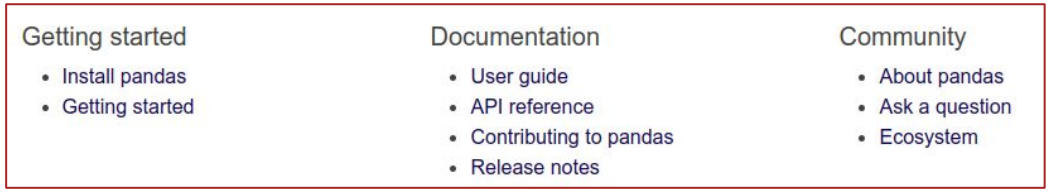
- Communicating the **project's goal**.
- Familiarization with coding and documentation **standards**.
- Familiarize new contributors with the existing **project ecosystem**.
- **Define templates** for: tutorials, documentation, issues and discussions.
- Providing **good first issues for engagement** and a workflow to translate user requests into specific requirements.

- Create a structured documentation space for your project.
  - **Tools:** Readthedocs, MkDocs, Sphinx, etc.
  - Structure the documentation with harmonized formatting to reduce cognitive load.

BioCypher docs structure



Pandas documentation structure



TensorFlow documentation structure

- Divide the learning process by using the Diataxis documentation framework [1].

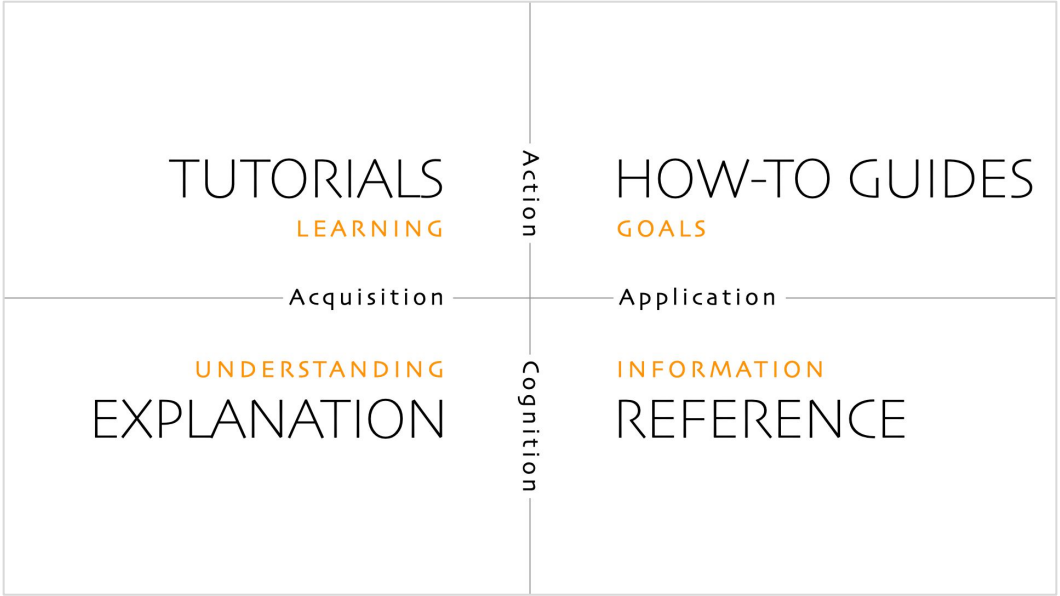
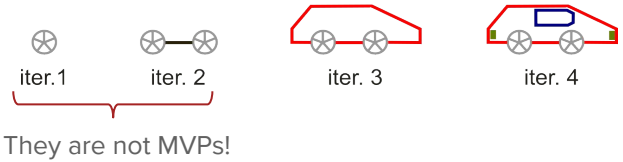


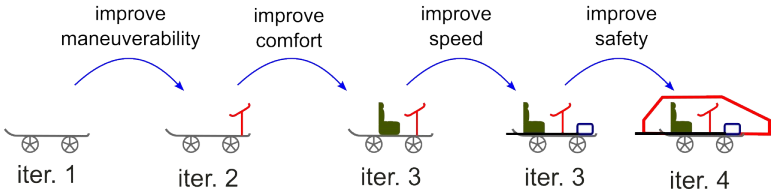
Image from Proclida, D. "*Diataxis documentation framework*". 2021. Retrieved from: diataxis.fr

- Start by defining steps for realizing user requests, a good strategy is to define a Minimum Viable Product (MVP).
  - *Fail fast, deliver faster* – Lean principle

Bad example of MVP



Good example of MVP



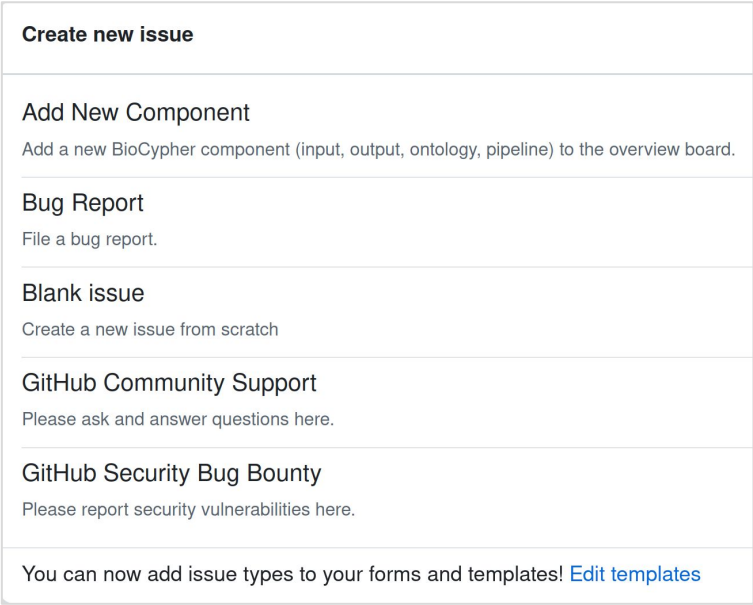
- Define a roadmap with milestones, each milestone should add functionalities to the MVP.
  - **Tools:**
    - GitHub Issues milestones
    - GitHub Projects (Kanban and Roadmaps views)

Structured contribution

Activities:

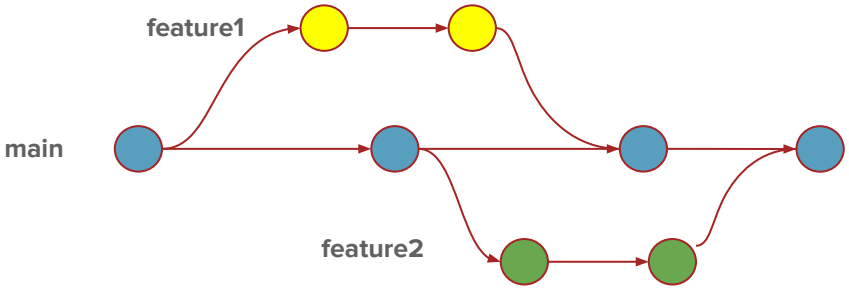
- Managing **the development process** with GitHub-specific workflows: branching strategies, commit messaging conventions, and pull request.
- **Automation** for code quality and documentation using GitHub Actions.

- Create templates for establishing a common language and a protocol.
  - **Tools:** GitHub Issues, GitHub Pull Request
  - About issue and Pull Request templates [2] offers a good guide



Templates designed for specific purposes

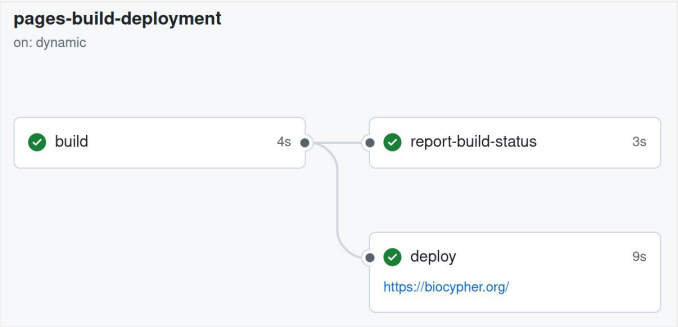
- Consult different branching strategies and choose one.
  - Branching strategies: GitHub Flow [3] (recommended), GitFlow, GitLab Flow, TBD, etc.
  - Explain this in your documentation too!



- Cultivate a culture of writing good Commit messages.
  - Resources: Conventional Commits [4], Write Better Commits, Build Better Projects [5]
  - **Document the strategy** new contributors should follow

- Write descriptive Pull Requests, balance between detail and practicality.
  - **Tools:** GitHub Pull Request templates
  - **D**o not **R**einvent **Y**ourself, learn, adapt and apply
  - Needing inspiration? Making a Pull Request [6]

- Automate **tests**, **deployments**, checks before releasing. This allows early delivery which is crucial in the community engagement.
  - **Tools:** GitHub Actions
  - Resources: GitHub Actions quickstart [7]



Ongoing Community engagement

Activities:

- Prioritize feedback-driven development
- Create effective communication channels
- Recognize and showcase contributors

Image from Fishel, D. (2023). *What is Sustainability? How sustainabilities work, benefits, and example.* Retrieved from: https://www.investopedia.com/terms/s/sustainability.asp

- Establish communication channels to gather ideas, opinions, complaints
  - **Tools:** Zulip, Slack, GitHub Issues, GitHub Discussions, etc.
  - Create categories to discuss, i.e. #development, #newfeatures, #documentation, etc.

- Inclusivity!
  - **Consider different audiences and adjust:** such as users, programmers, sponsors, science communicators, and others
  - Resources: Elevator speech, One concept in 5 levels of complexity!

- Code + Connect + Learn = Hackathons 🚀
  - Show your project, but most importantly, let people try it, use it and **criticize it**
  - **Feedback is the real measure for defining changes**

Summary and Key Points

- Build a community for long-term software maintenance, especially with limited resources
- Use a structured methodology to boost engagement and efficiency
- Leverage feedback to improve development and community involvement
- Prioritize clear communication with diverse audiences
- Adapt our three-phase methodology to fit your needs
    - That is why we talk about **workflows** and not **The Workflow!**

Thank you!

Scientific Software Center

Q&A

References

[1] Proclida, D. “*Diataxis documentation framework”*, 2021. [Online]. Available: diataxis.fr

[2] GitHub, “About issue and pull request templates”, *GitHub,* 2025. [Online]. Available: https://docs.github.com/en/communities/using-templates-to-encourage-useful-issues-and-pull-requests/about-issue-and-pull-request-templates

[3] GitHub, “GitHub flow”, *GitHub*, 2025. [Online]. Available: https://docs.github.com/en/get-started/using-github/github-flow

[4] Conventional Commits, "Conventional Commits Specification v1.0.0," *Conventional Commits*, 2024. [Online]. Available: https://www.conventionalcommits.org/en/v1.0.0/

[5] Dye, Victoria. “Write better commits, build better projects”, *GitHub Blog*, 2022. [Online]. Available: https://github.blog/developer-skills/github/write-better-commits-build-better-projects/

[6] Pandas Development Team, "Making a Pull Request," *Pandas Documentation*, 2024. [Online]. Available: https://pandas.pydata.org/docs/development/contributing.html#making-a-pull-request

[7] GitHub, “Quickstart for GitHub Actions”, *GitHub*, 2025. [Online]. Available: https://docs.github.com/en/actions/writing-workflows/quickstart

Credits

Icons retrieved from Flaticon.com

- Knowledge icons created by Freepik - Flaticon
- Help icons created by Freepik - Flaticon
- Contribute icons created by pojok d - Flaticon

Icons retrieved from Creazill.com

- https://creazilla.com/media/silhouette/14906877/conductor-maestro