Thoughts on the organization of CORSIKA 8 development

A. Augusto Alves Jr

Presented at CORSIKA 8 Air-Shower Simulation and Development Workshop,
Heidelberg, 13 July 2022

Karlsruhe Institute of Technology

CORSIKA 8: development guidelines

CORSIKA 8 aims to be a C++ framework. Currently, we also deal with code written on other languages like C, Python, FORTRAN, CMAKE. From a maintainer/developer point of view:

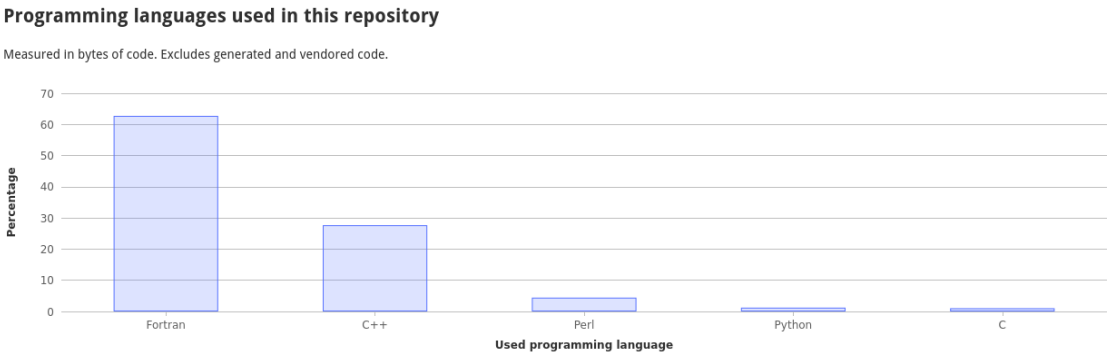
- The project is hosted on Gitlab at KIT infrastructure.
- Some tests (pipelines) run external infrastructure
- Code is distributed under GPLv3
- A preliminary, somewhat incomplete, version manual is available on ReadTheDocs. Part of the code is also documented using Doxygen.
- 58 members are listed in the repository. Some are blocked, others are double counted. Not all members contributed with code.

Repository

- The repository can be accessed at `https://gitlab.iap.kit.edu/AirShowerPhysics/corsika`.
- Currently it has 3,215 commits, 82 branches, 7 forks, 8 tags and 4 releases.
- Also: 115 open issues and 15 pending open requests.
- The Wiki of the project can be accessed at (`https://gitlab.iap.kit.edu/AirShowerPhysics/corsika/-/wikis/home`) and provides additional information on other aspects: coding conventions, TODO lists, talks on conferences etc.

In summary, the development has been fairly active. Many milestones have been met.

Repository: Languages



Repository: Homepage

IAP GITLAB

Menu | Search GitLab | 1 | 16

C corsika

- Project information
- Repository
- Issues 115
- Merge requests 15
- CI/CD
- Security & Compliance
- Deployments
- Monitor
- Infrastructure
- Packages & Registries
- Analytics
- Wiki
- Snippets
- Settings

Air Shower Physics › corsika

C **corsika**
Project ID: 4

Unstar 17 | Fork 7

3,215 Commits | **82** Branches | **8** Tags | **4.6 MB** Files | **4.2 GB** Storage | **4** Releases

Project badge Project badge docs passing License GPLv3

master | corsika / | History | Find file | Web IDE | Clone

Merge branch '506-qgsjetii-04-interface-broken' into 'master'
Maximilian Reininghaus authored 6 days ago
e3f13a19

README | Other | CONTRIBUTING | CI/CD configuration | Add Kubernetes cluster | Configure Integrations

Name	Last commit	Last update
.gitlab/merge_request_templates	IKP -> IAP	6 months ago
cmake	clang-format	1 year ago

Repository: Wiki

IAP GITLAB

Menu | Search GitLab | 1 | 16

Air Shower Physics > corsika > Wiki > **Home**

Last edited by **Ralf Ulrich** 6 months ago

Page history | New page

Home

CORSIKA 8 Framework for Particle Cascades in Astroparticle Physics**

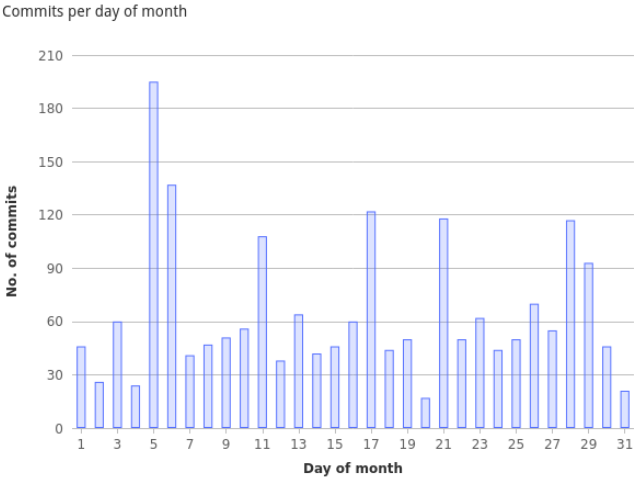
Documentation and reference guide for the CORSIKA 8 software framework for air shower simulations. We aim that CORSIKA remains the most comprehensive framework for simulating particle cascades with stochastic and continuous processes. The purpose of CORSIKA is to simulate any particle cascades in astroparticle physics or astrophysical context. A lot of emphasis is put on modularity, completeness, validation and correctness. To boost computational efficiency different techniques are provided, like thinning or cascade equations.

The software makes extensive use of static design patterns and compiler optimization. Thus, the most fundamental configuration decision of the user must be performed at compile time. At run time only specific model parameters can still be changed.

Clone repository | Edit sidebar

CORSIKA 8 Wiki

Repository: Coding frequency



Commit statistics for master May 10 - Jul 06

Current workflow for developers

People willing to get involved and contribute with code should:

1. Make an account at CORSIKA 8 Gitlab repository
2. Get in contact with the core developers team (CORSIKA 8 bi-weekly general call) to communicate intentions: pick-up a pending task, implement some new feature.
3. Branch the `master` and starting coding.
4. Eventually open `issues` if features requiring attention are found.
5. Open a pull request.
6. Code review, documentation, eventual corrections and finally the pull request is accepted.
7. The user branch is deleted.

Comments

This workflow works better when the number of active developers is small and the number of features under development is limited or the lifetime of the development branches is very short (days). Otherwise

- Promotes proliferation of branches. CORSIKA 8 repository has currently 82 branches, despite no formal release has been done yet.
- Not every work gets to a conclusion. This produces zombie (stale) branches.
- Makes cumbersome to coordinate working on related features. Users now will need to figure out how to synchronize the related branches. This impact clarity in the repository history (frequently requires rebase).

Comments

- It is difficult to handle large and deeply impact changes. Examples:
  1. It would be quite confusing to perform the last code refactory (2020/21) under this scheme. We just forked away from the repository and merged afterwards.
  2. Radio branch (see Nikos talk, yesterday) is now hosting a huge amount of contributions, related to other branches and issues, which are connected to other features.
- Makes difficult for newcomers to figure out what is going on. Example: Which branch should I start from? And if it is merged before I finish my work?
- Obfuscate tag-releases.
- Puts maintenance pressure on the core developer team.

About Git and forking

That "anybody can maintain their own version" worried some people about the GPLv2, but I really think it's a strength, not a weakness. Somewhat unintuitively, I think it's actually what has caused Linux to avoid fragmenting: everybody can make their own fork of the project, and that's OK. In fact, that was one of the core design principles of "Git" - every clone of the repository is its own little fork, and people (and companies) forking off their own version is how all development really gets done.

Linus Torvalds: Interview on Linux and Git

Alternative workflow

One popular choice among large projects

- CORSIKA 8 repository would have only `master` branch and the tag-releases.
- Developers fork the repository, implement the features and submit a pull request when they are done.
- Code on `master` should be always compilable and updated often.

This would ease coordination for working on large sets of features, keep the repository clean.

Another alternative workflow

One popular choice among large projects

- CORSIKA 8 repository would have only `master` and `devel` branches and the tag-releases.
- Developers fork the repository, implement the features and submit a pull request when they are done.
- `devel` would be updated often and always contain the latest compilable code.
- Code on `master` should more stable and maintained against bugs.

This would ease coordination for working on large sets of features, keep the repository clean.

Conclusions

Looking in advance for starting the relase cycle of CORSIKA 8 and the growing of our community of users and contributors, some reorganization of the CORSIKA 8 repository is probably needed in order to deploy a more scalable workflow for developers.