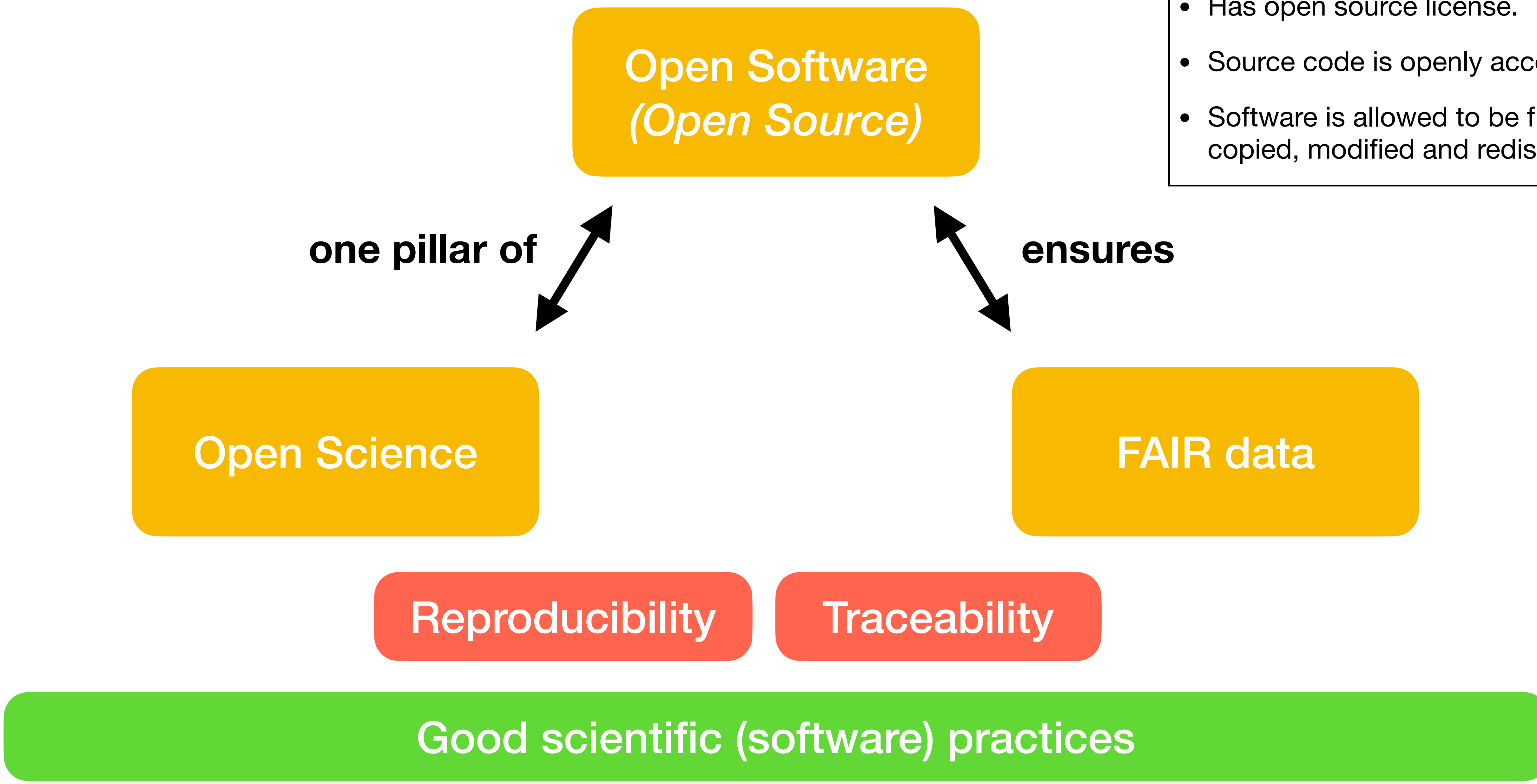
Open Software in theoretical nuclear physics

Jan Staudenmaier

Research Data Management at GSI/FAIR
05.07.22

An example of open software in theoretical physics

- Has open source license.
- Source code is openly accessible
- Software is allowed to be freely copied, modified and redistributed



3 types of theoretical research data

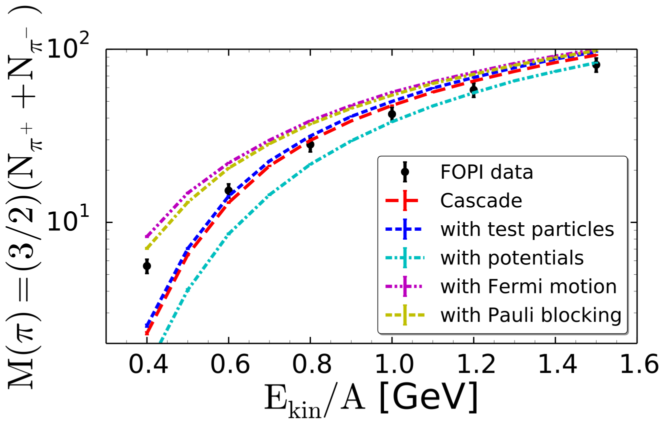
Software code

Raw output

Analysis output

```
template <typename Modus>
void Experiment<Modus>::run() {
  const auto &mainlog = logg[LMain];
  for (event_ = 0; !is_finished(); event_++) {
    mainlog.info() << "Event " << event_;

    // Sample initial particles, start clock, some printout
    initialize_new_event();

    run_time_evolution(end_time_, {});

    if (force_decays_) {
      do_final_decays();
    }

    // Output at event end
    final_output();
  }
```

```
#!OSCAR2013 particle_lists t x y z mass p0 px py pz pdg ID charge
# Units: fm fm fm fm GeV GeV GeV GeV GeV none none e
# SMASH-2.1.4-106-ga47efc7cc
# event 0 out 260
2 3.72191 0.657009 -2.00559 0.138 0.38715131 0.169799162 0.162103534 -0.275195976 -211 103 -1
2 3.63315 -2.14374 1.23356 0 0.362085072 0.177902243 0.0732821256 -0.306734611 22 106 0
2 -1.252 0.719679 -0.486824 0.938 1.51435403 -0.667634996 0.971713342 0.153169278 2112 108 0
2 3.63315 -2.14374 1.23356 0 0.108587222 0.100810028 0.0229643523 -0.0331837521 22 107 0
2 -0.358008 2.48197 1.07918 0.494 0.659367661 0.404886557 -0.125488216 -0.105115633 311 287 0
2 -0.647648 4.84535 -2.30154 0 0.357781696 -0.105884464 0.0848794705 -0.331046368 22 110 0
2 0.349074 -0.0767215 -3.72608 0 0.324353409 -0.269677097 -0.172201064 0.0531619308 22 285 0
2 -0.805182 4.6756 2.23076 0 0.455009882 -0.13295012 0.397190795 -0.177757504 22 283 0
2 2.9905 -0.596719 3.36116 0 0.252026403 0.086857639 0.0704115657 0.225865601 22 281 0
2 -5.1396 -2.06235 0.853681 0 0.301215903 -0.221223834 0.116393356 0.168058389 22 112 0
2 -2.44794 1.04686 3.03496 0 0.343892837 -0.0732091577 -0.144587005 0.303310567 22 289 0
2 -0.647648 4.84535 -2.30154 0 0.0582689648 -0.0124515847 0.0545570898 -0.0162405128 22 111 0
2 2.55353 -5.45727 -1.29306 0 0.750399042 0.516772234 -0.233334012 -0.491528656 22 291 0
2 -2.51007 -0.402294 0.36419 0 0.4877252 0.285629066 -0.255381027 -0.301782104 22 273 0
2 4.65611 0.153427 -0.933636 0 0.115504605 -0.0254671287 0.112125873 0.010978514 22 397 0
2 0.95196 -0.502446 1.00985 0 0.0565190066 -0.0114371818 0.0457897665 0.0310947948 22 293 0
2 -2.97025 1.73266 2.08891 0 0.222641575 0.172384075 0.137840793 0.0292047428 22 395 0
2 3.75688 -1.62538 0.750215 0 0.582619971 0.0507721506 -0.486080806 -0.317165051 22 262 0
2 -3.54865 3.24689 0.900235 0 0.217715625 -0.217500004 -0.00099329751 0.00963613634 22 393 0
2 0.294375 1.65573 -1.03035 0 0.0711606666 -0.0169726505 0.0297308157 -0.0623846792 22 391 0
2 0.938074 1.18135 -4.20574 0 0.19149254 0.0295819734 0.18487337 -0.0402012044 22 257 0
2 3.93456 2.76484 -0.59175 0 0.380113362 -0.0881711276 0.0402539286 0.367548148 22 295 0
2 -0.060849 1.81542 -3.41033 0 0.222550605 0.151966015 0.146550604 0.0704132278 22 253 0
2 1.58311 -2.74002 -1.84352 0 0.0769533739 0.0562872217 0.0410792658 0.0326506409 22 399 0
2 -0.0767581 -3.18546 3.7773 0 0.299750863 -0.0571561566 -0.173404343 -0.237728179 22 249 0
```



3 types of theoretical research data

Software code

Source in C++, Python, Fortran, …

Mathematica Notebooks

~100 MBs

Raw output

ASCII text

Binary files

~ TBs

Analysis output

Results

Plots and their x and y data

~ MBs

3 types of theoretical research data

Software code

Publicly Accessible (Open Source)

Documentation

Versions

Raw output

RDM varies with calculation time and data size

Challenging to store

Absent in analytic calculations

Analysis output

Easy to store and publish

Workflow documentation

3 types of theoretical research data

Software code

Publicly Accessible (Open Source)

Documentation

Versions

Practical solution in many cases

Analysis output

Easy to store and publish

Workflow documentation

if raw data is relatively easily reproducible

3 types of theoretical research data

Software code

Publicly Accessible (Open Source)

Documentation

Versions

Analysis output

Easy to store and publish

Workflow documentation

Focus of this talk

Next slide

Data Management Policy of CRC-TR 211

CRC-TR 211
Strong-interaction matter under extreme conditions

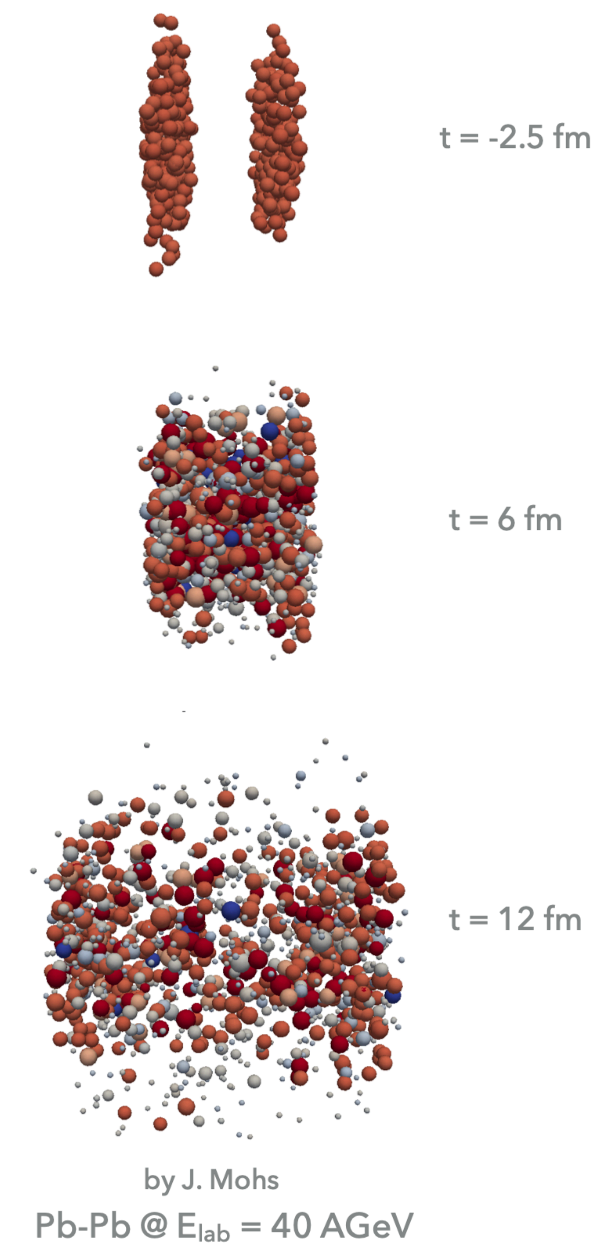
- DFG funded collaborative research center between Goethe University Frankfurt, TU Darmstadt, JLU Gießen and Bielefeld University
- Implemented and enforces **data policy**
- Common directory structure for publications
  - **`source`**: contains LaTeX source files and
    - **`figures`**: Separate pdf- or eps-files for each figure in the publication
    - **`anc`**: Separate text files with the data to reproduce the figure
  - **`workflow`**: Everything needed to reproduce results (used software, packages, scripts, handwritten notes, …)
- Source directory is to be uploaded on **arXiv. org**

HEPData
Repository for publication-related High-Energy Physics data

SMASH*

* **S**imulating **M**any **A**ccelerated **S**trongly-interacting **H**adrons

- Hadronic transport approach for low energy collisions and dilute non-equilibrium stages of high energy heavy-ion collisions
- Goal: standard reference for hadronic system with vacuum properties
- Predict and explain observables for future high-baryon density nucleus-nucleus collisions like at FAIR



J.Weil et al, Phys. Rev. C 94, 054905 (2016)

Example: SMASH

- SMASH is a ~30.000 lines C++ code with development since 2012
  → Publication in November 2018

- Large collaborative effort under the supervision of Hannah Elfner: 33 authors + 5 external contributors

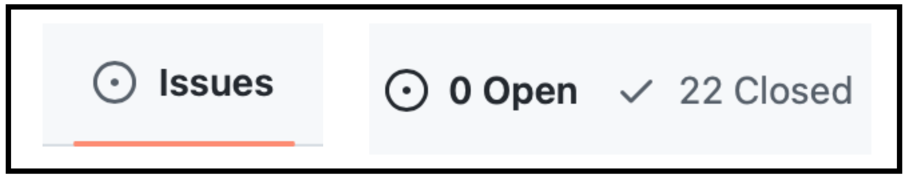
- Used by many theory groups and experimental collaborations around the world e.g. STAR, ALICE, HADES/CBM, NA61/SHINE and theory groups in McGill, BNL or Japan

- Good example: A large additional effort to follow good scientific (software) practices

- Idea from the beginning to use modern open source development tools and practices

SMASH on Github

- Made accessible on Github as the most popular platform for open-source software
- License from the start of development: GPL-3.0
- Use of project management tools like issues, pull requests and the wiki
- External users are welcomed to contribute to the code via opening public issues and pull requests



Version control

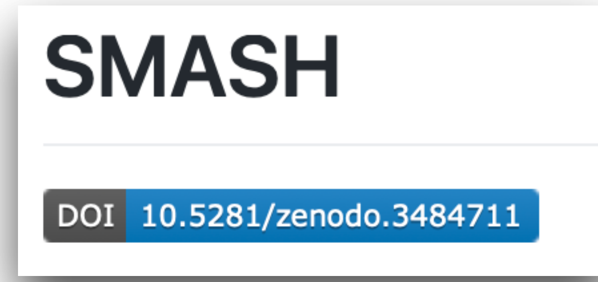
- Version control is the basis of reproducibility in theoretical software efforts
- Version control systems: **Git**, subversion, Mercurial, …
- Pinpoint precise state of source that lead to result
- Tagged versions or commit hash can be quoted in paper

  SMASH-2.2 **commit b08a1a2acd21210f6c3d4db555608c7c08e8c443**

- Also ensures data integrity

SMASH Versions

- Every 6 months a new release is finished
- Issues („to-dos“) are assigned to these milestones
- Explain in Changelog file in detail what is new or different
- Semantic versioning (MAJOR.MINOR.FIX)



- **Github release** can easily get a Digital Object Identifier (**DOI**) issued by Zenodo

[19] D. Oliinychenko, V. Steinberg, J. Weil, M. Kretz, H. Elfner (Petersen), J. Staudenmaier, S. Ryu, A. Schäfer, J. Rothermel, J. Mohs, F. Li, L. Pang, D. Mitrovic, A. Goldschmidt, L. Geiger, L. Prinz, J.-B. Rose, and J. Hammelmann, SMASH-1.6 (2019), doi: 10.5281/zenodo.3485108.

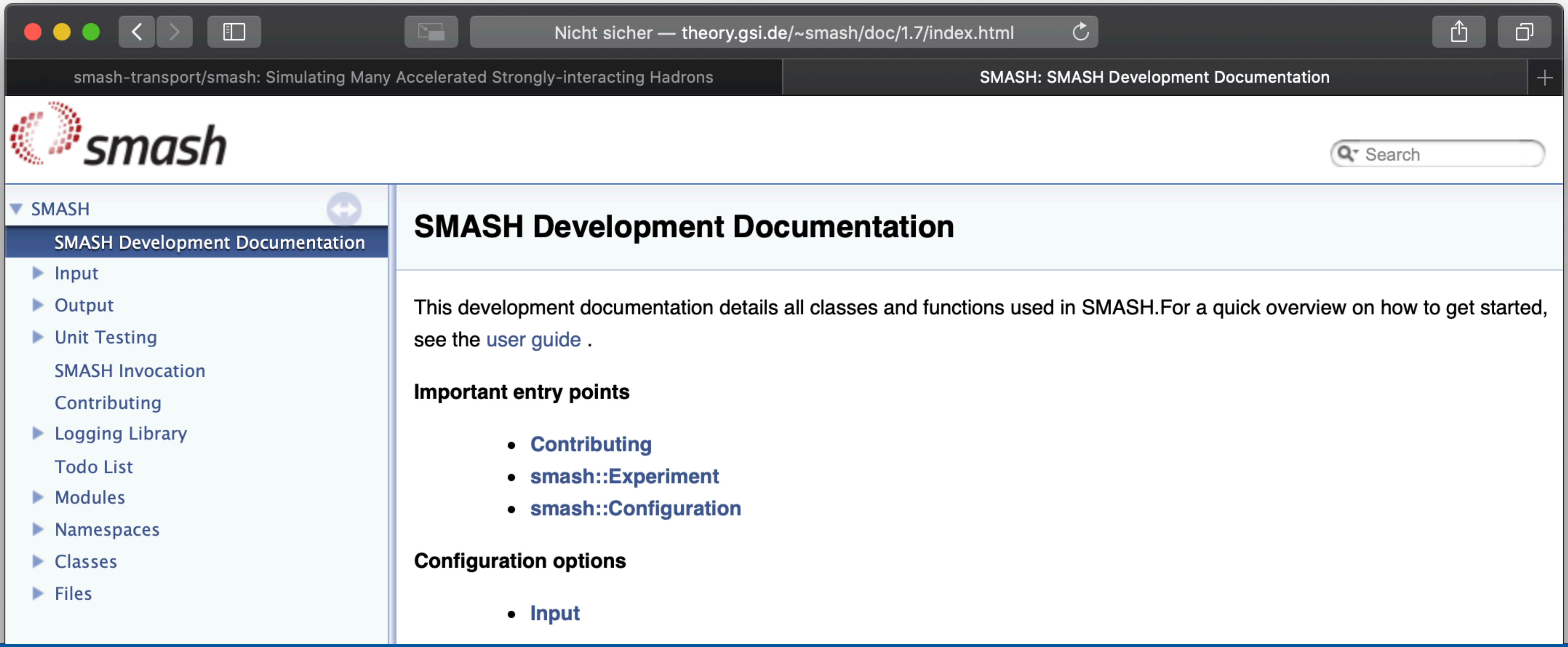
- Code versions are citable

- Version is archived

- Funded by CERN and the European Commission (Open AIRE)

Documentation and User Guide

- Also available online (for every version)
- Generated with `doxygen`: HTML documentation from special code comments



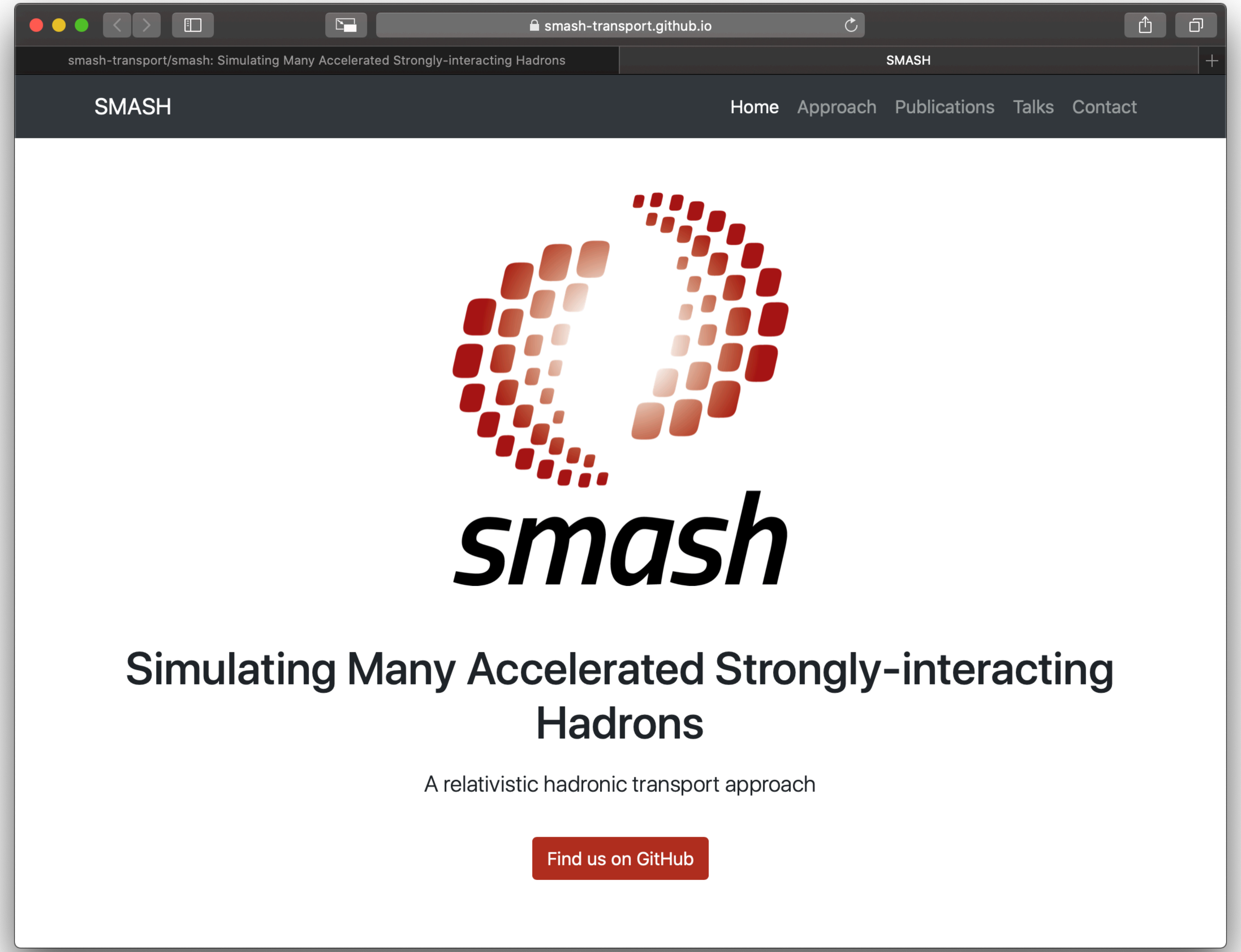
SMASH Configs

- Ensuring reproducibility on a practical level
- All used input files are merged into one file
  → automatically placed with output
- Includes the employed fixed random seed
- Re-run the same calculation
- Serves as metadata reference

config.yaml


```yaml
General:
    Modus:          Collider
    Time_Step_Mode: Fixed
    Delta_Time:     0.1
    End_Time:       200.0
    Randomseed:     -1
    Nevents:        100

Output:
    Output_Interval: 10.0
    Particles:
        Format:         ["Oscar2013"]

Modi:
    Collider:
        Projectile:
            Particles: {2212: 79, 2112: 118} #Gold197
        Target:
            Particles: {2212: 79, 2112: 118} #Gold197
```

Website



FAIR

Tests

- **Test suite** contains mostly unit and run tests - *nothing is broken*
  - Has to pass on every change before merged into main branch (+ code is reviewed by someone else)
  - Automatically ensured by CI Service (Github Actions)
- **Physics analysis suite** compares observables against previous versions and experimental data
  - Run for every new release
  - Results are public and analysis code is also open source

Rivet / HepMC

FAIR

- **Rivet**: particle-physics MC analysis toolkit
  https://rivet.hepforge.org
  - Provides experimental analyses useful for MC generator comparison to experimental data (open source)
  - Allows to add own analysis e.g. done by LHC experiments
  - Basically stores the workflow from raw output to extracted observable

MC generator = Monte-Carlo approaches like Pythia or SMASH

Rivet / HepMC

FAIR

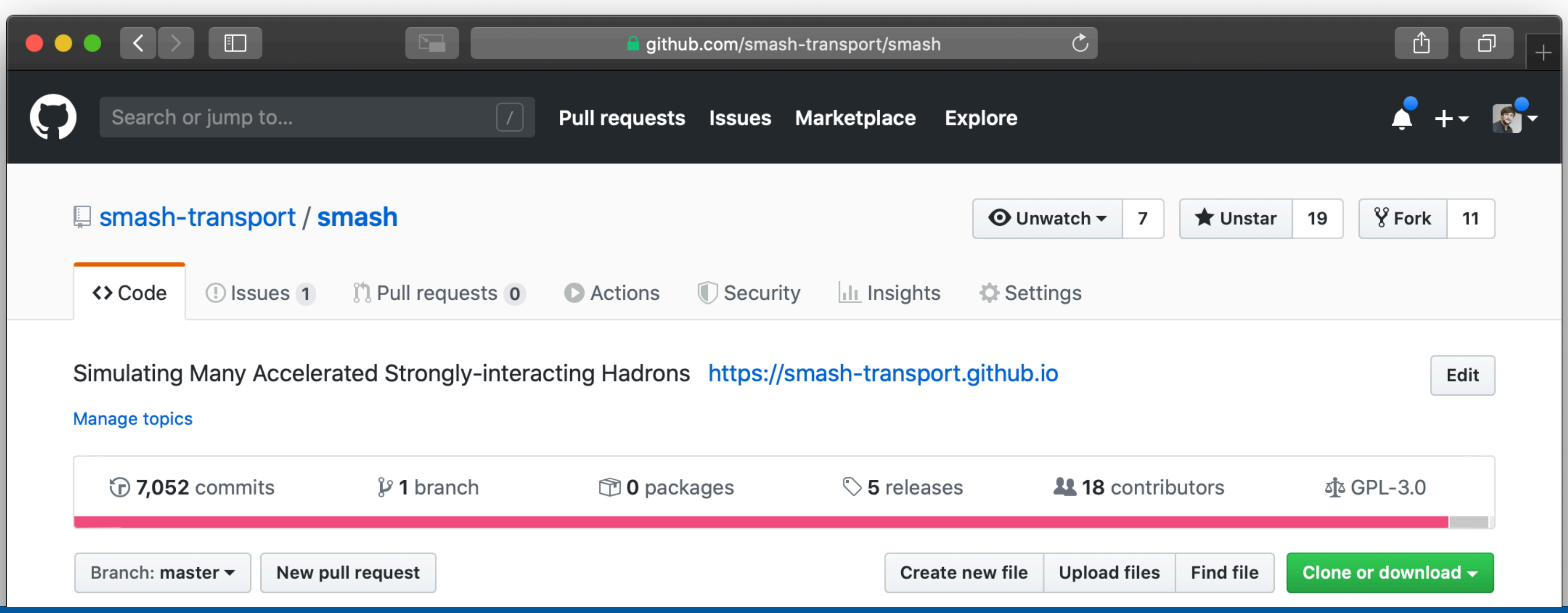
- **HepMC**: C++ event record for High Energy Physics Monte Carlo generators and simulation
  http://hepmc.web.cern.ch/hepmc/
  - Standardized output format e.g. used by Rivet
  - Stores all interaction vertices
  - Newly included in SMASH since `SMASH-2.2`
- SMASH also interfaces directly with Rivet with no output written to disk
  → external pull request contribution

Open Software and FAIR principle

- **Findable**: Website, Github repository, Documentation, Zenodo, …
- Publicly **Accessible**: Open Source, Github, arXiv anc directory, …
- **Interoperable**: Documentation, Output standards (HepMC/Rivet), …
- **Reusable**: Version control, Releases, User Guide, …

Summary

- Theoretical research data has three parts: software, raw data, results
- Open Software is one of the main requirements in theoretical work to ensure FAIR data and Open Science
- SMASH is one example, where we undertake a large additional effort to follow this practice and we have made good experiences with it



https://github.com/smash-transport/smash

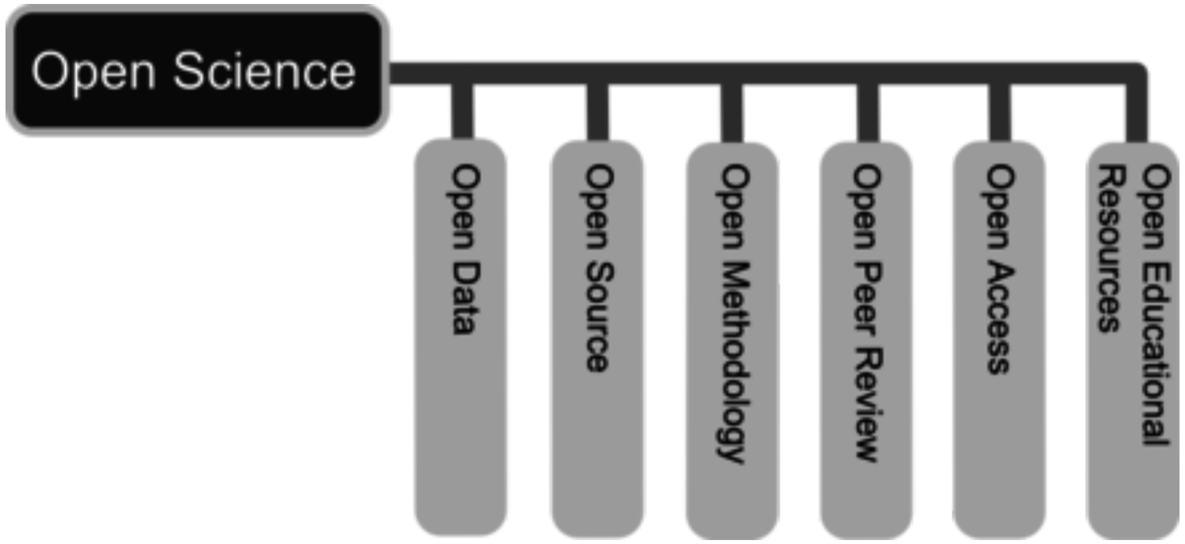
Backup

License

- GNU General Public License v3.0 (or short GPL-3.0)

Permissions	Limitations	Conditions
✓ Commercial use	✗ Liability	ⓘ License and copyright notice
✓ Modification	✗ Warranty	ⓘ State changes
✓ Distribution		ⓘ Disclose source
✓ Patent use		ⓘ Same license
✓ Private use		

Open Science



https://de.wikipedia.org/wiki/Offene_Wissenschaft#cite_note-1

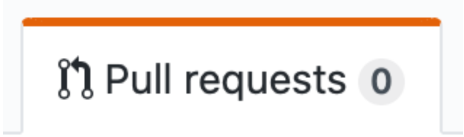
Discussion

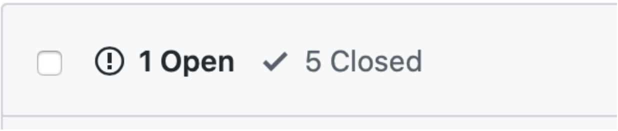
Our Experience

- Overall: **Positive** experience
- Community appreciates publication
- Ongoing initiatives to contribute to code
- Widely used by experimentalists
- Other approaches considering to go public, too
- Positive: People did not care too much e.g. no „support nightmare“ (no major changes concerning our work)
- Inspired other projects to be open-sourced

Our Experience

- Code base and documentation improved significantly
    - Some extra work is necessary (the later you think about going open source the more work)
    - Helped to be already a larger collaborative effort (infrastructure already set up at the beginning)
- Switch to Github was smooth and improved internal workflow
- Vital to have test and physics analysis suite
- Support-Stats: No pull requests, 6 issues (3 feature requests, 1 bug, 2 installation problems)
- …





Discussion

Why should you open source a code?

Why is your code (not) open source?

What are potential issues with open-sourcing a code?

Contra

- Overhead: create proper documentation, maintain infrastructure, support, …
    - „Hacking“ together results is faster in the short-term
- Someone else might write your paper first
- Uncontrolled and wrong usage of your code
    - You get confronted with/are blamed for wrong results
- Your code can get stolen without credit
- …

Pro

- **Good scientific practice**: Reproducibility, traceability and credibility of your results
- **Research funded by the public**, therefore you should make your research available to the public
- Funding agencies encourage (future: require?) open source
- Follows the general trend of Open Science (Open source is one pillar)
- Not everybody has to „reinvent the wheel“

Pro

- Other will use your approach more —> it gains more relevance in the community —> **more citations**
- More collaboration with other scientists —> **more publications**
- **Code is automatically improved**: Bugs are found, code is kept clean, documentation is done
- Developers get to know open source development (**new personal skill** for C.V.)
- Convenient, if you anyway want to share the approach (no e-mails with users, **easy to use** for them, explain and setup everything only once)

Old Talk

Platform

- Hosted on Github, the most popular place for open-source software
- Private development repository
- Public repository updated with new versions and allows to open issues and pull requests