CODING ACTIVITIES

DECENTRALIZED STATE-OF-THE-ART SOFTWARE DEVELOPMENT FOR HIGH QUALITY, OPEN-SOURCE CODESubtitle

21.03.2019

The three main developments in the project are:

- Metrological frameworks for the traceable calibration of smart sensors (with digital pre-processed output) and strategies to digitally enhance existing calibration facilities correspondingly.

- Guidelines and best-practices for the organisation and metrological treatment of networks of smart sensors, including timing issues, ambient conditions and data aggregation.

- Mathematical frameworks for the (close-to) real-time evaluation of uncertainties in IoT sensor networks in accordance with guidelines in metrology.

15 partner institutions spread all over Europe are working together with several international industry partners on these topics. In all three areas there is collaborative coding required to reach the project goals and produce the desired results to create industry impact. One underlying idea for putting a focus on collaborative open-source coding is to

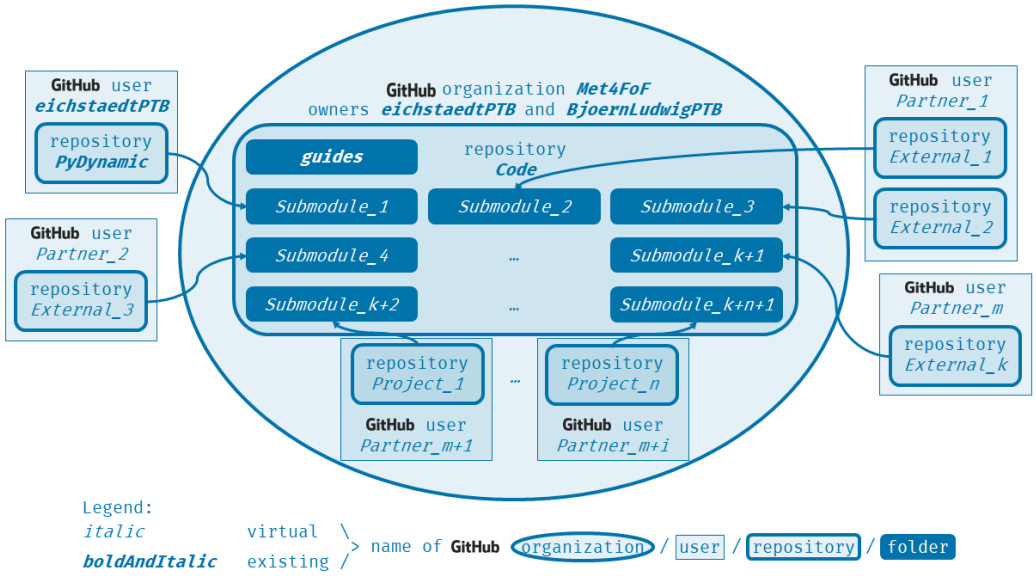
... ensure an easy uptake of the approaches by industrial users.

Several tasks and activities in the project address this aspect. Wherever feasible, the results, in particular but not restricted to code, are published early together with the steps taken to obtain them following the FAIR Principles. The main challenge in achieving that is to enable all partners using their institution-specific tools and workflows and common research practices while still integrating everything smoothly on-the-fly.

That is why during the kick-off meeting an early decision was made for publishing all project related code in a [collective Git repository](https://github.com/Met4FoF/Code) hosted on GitHub where during the whole course of the project current developments can be shared and accessed over one single point of contact. It was agreed on formulating coding conventions, circulating them and putting one PTB developer in charge of maintaining this central repository, to keep an overview of all software development activities and assuring the published code meets the project's conventions. To this end the code writing and publication in open-source software-repositories is perceived as an inseparable part of the scientific process.

The collective repository was quickly, initially set up with introductory material for Git like a short Git Glossary, links to external references about the Git installation process and guidance to perform introductory tasks, e.g. getting the collected code and staying up-to-date. Additionally a couple of styling conventions for the produced code were added. Later and ongoing the repository was augmented with step by step guides on basic Git tasks like initialization of repositories, committing, pushing and pulling, and more advanced coding conventions about code structure and design, documentation and source code management, describing what practices to adopt while using Git.

The chosen architecture of the collective repository is based on the Git submodules mechanism. This ensures all partners can use their standard development workflows including their usual Git habits and practices or it makes sure researchers being new to Git can work and learn in confinement without the fear of corrupting the projects code base. At the same time, it creates the desired single point of contact to the project's code. A submodule consists of a pointer to a specific commit in any repository which leads to a subfolder in the so called superproject containing the specified version of the submodules code. These pointers are inserted by the collective repository maintainer to minimize complexity for the involved developers.



A CI/CD pipeline was set up for the collective repository to use modern software development practices on the produced scientific code without putting to much load on every single developer and bundle up the efforts for the setup and maintenance. Before a submodules pointer is reset the pipeline is triggered by a Pull Request in the collective repository to check if tests have passed. The testing code and other parts of the pipeline are discussed with the developers for the initial set up of the according submodule and adapted while the project advances.