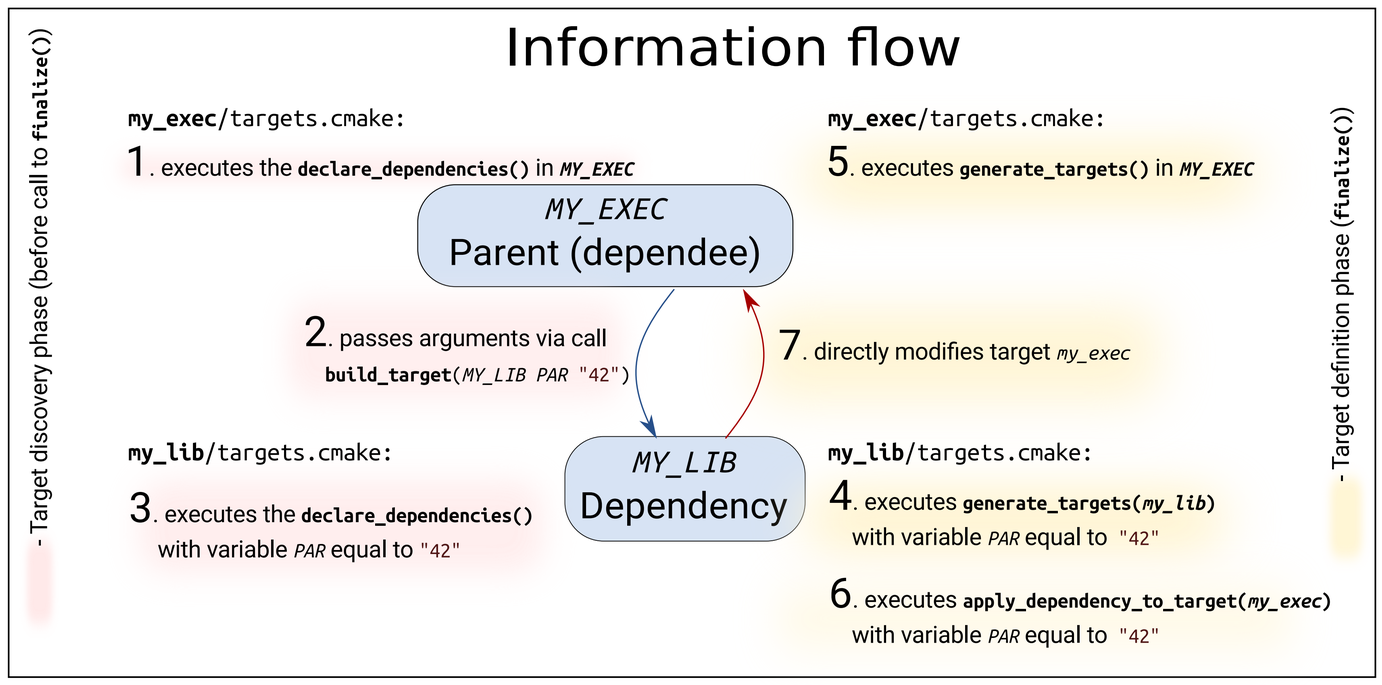
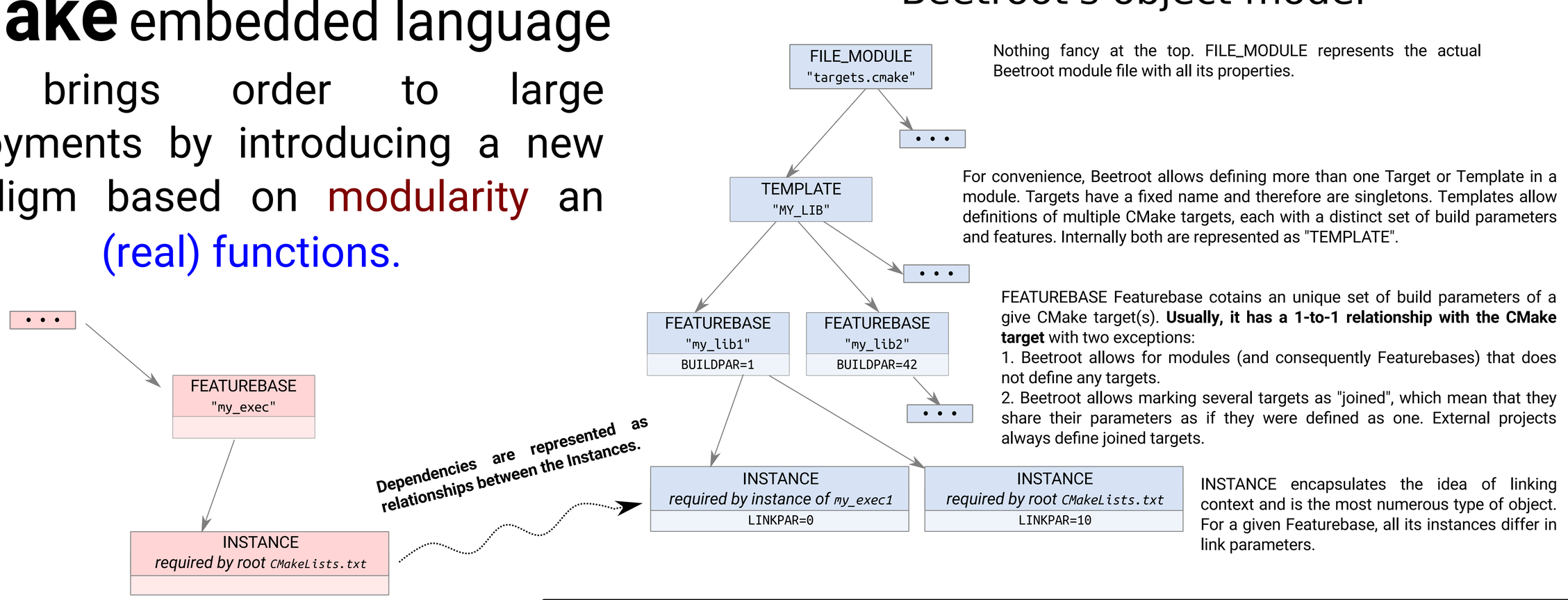
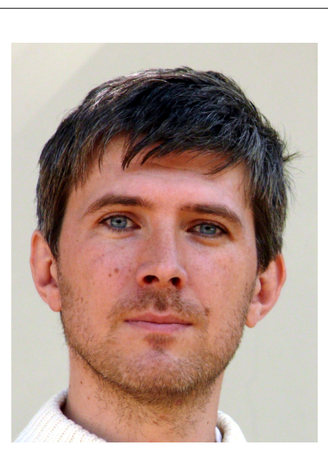
Meet **Beetroot**

CMake embedded language

that brings order to large deployments by introducing a new paradigm based on modularity an (real) functions.





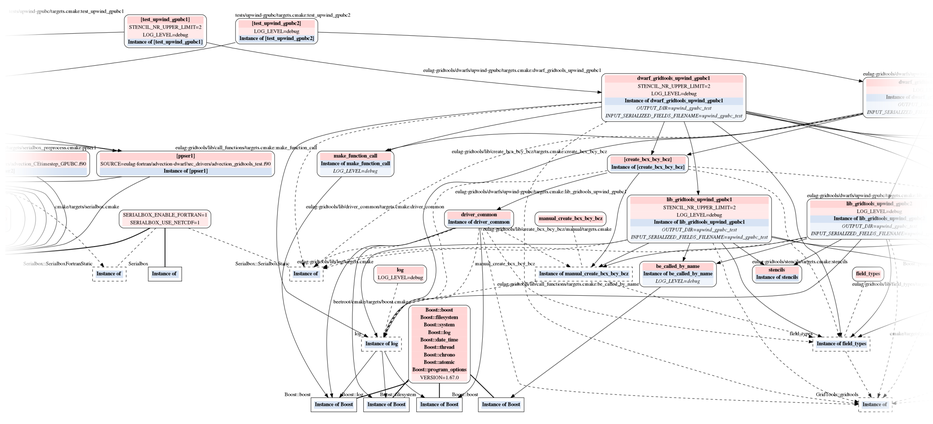


Hi, I'm **Adam Ryczkowski**, the designer of the Beetroot. This is how I look like. I am here to answer your questions, discuss the building process in general and CMake and Beetroot in particular.

At work, I design and supervise the build process of an effort to port existing large Fortran codebase implementing physical equations we use for weather forecast into C++. We use many external libraries, and our tests involve compiling pairs of Fortran and C++ executables with compatible test parameters.

"*Beetroot is all about **maintainability** and **flexibility** of the build process*"

Beetroot was born out of the need to deliver a complex build requirements. The idea was started in 2017, and was initially set to reduce usage of global variables and augment parameter handling.
After several iterations and several complete redesignings, the current shape of the Beetroot was formed.



In case you wonder, this is how our dependency graph looks like. The picture is cropped to show ca. 5% of the contents. **This is what drove the development of the Beetroot.**

*In case I am not around, and **you know any other CMake productivity library** that has a similar scope to the Beetroot, please take a pen and write it down here:*

CMake module allowing to create a build system on top of ExternalProjects `https://github.com/commontk/Artichoke`
"Just A Working Setup" `https://github.com/DevSolar/jaws`
`https://github.com/ecmwf/ecbuild`