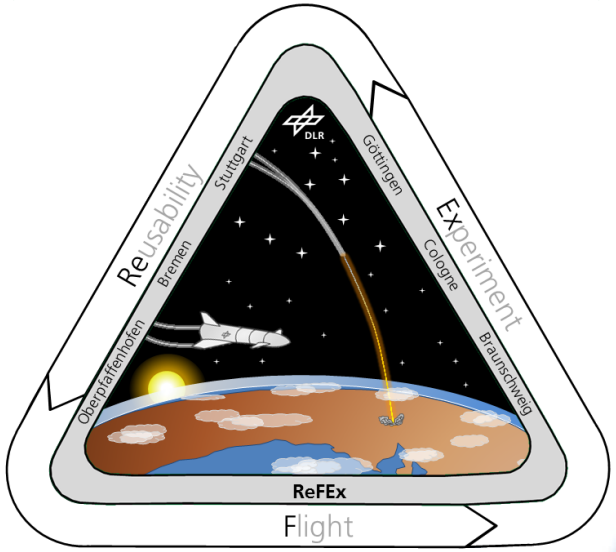
Commit, Release, Package:
Automation in the development process for the ReFEx GNC System

Jan Sommer
German Aerospace Center
Institute for Software Technology

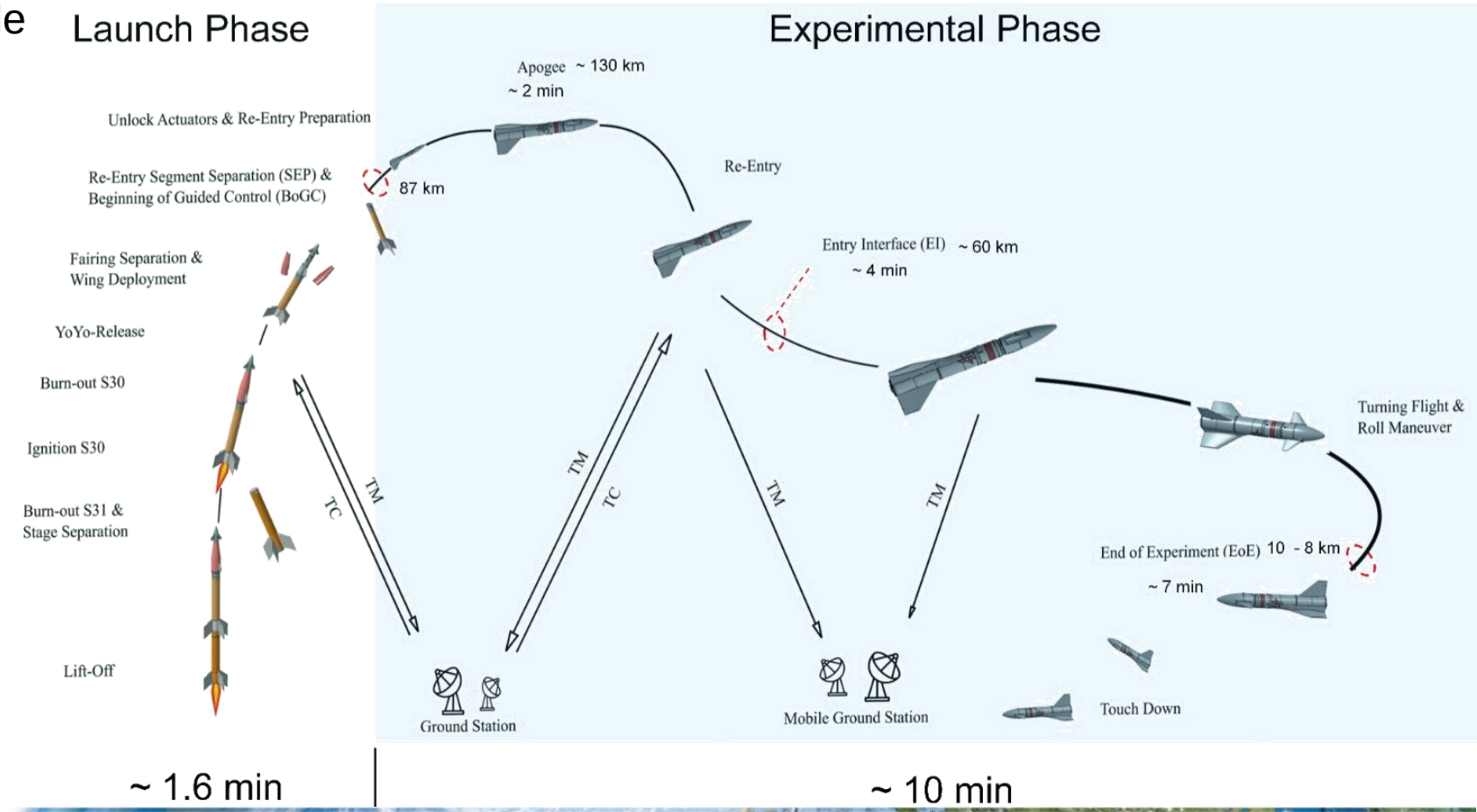


German Aerospace Center – Distributed Research Facility

- More than 10000 employees work in 54 institutes and facilities at 30 sites across Germany
- Only few institutes dedicated to software, but most engineering fields will require at least some domain specific software development (e.g. control algorithms)
- Large projects require collaboration between institutes
- Software easier to move than facilities like wind tunnel or clean rooms

Reusability Flight Experiment (ReFEx) – General Overview

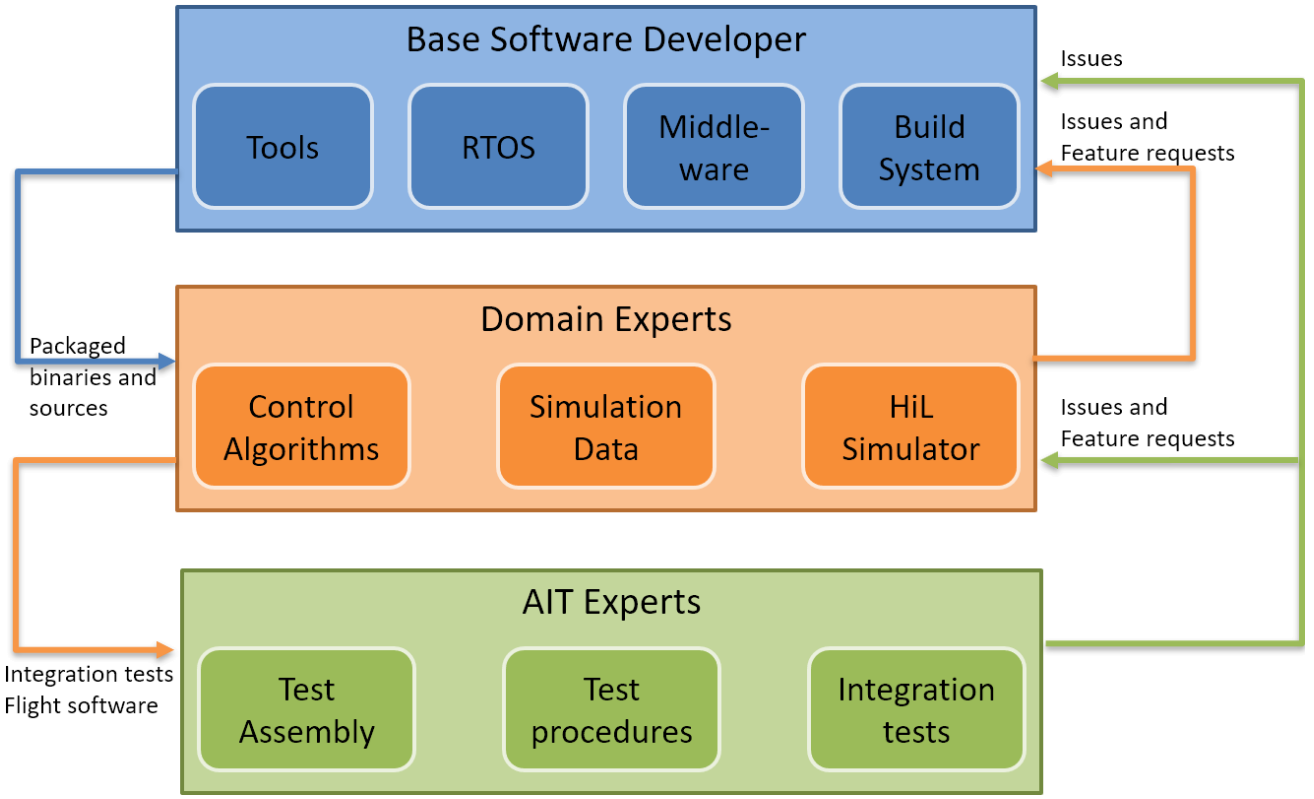
- Technology demonstrator for a fully aerodynamically controlled reusable launch vehicle
- Launched via a VSB-30 sounding rocket
- Autonomously controlled flight in hypersonic to subsonic velocities
- Launch planned for 2023 from the Koonibba Test Range (KTR) in South Australia



Challenges for the Software Development

The Situation:

- Small development teams distributed across DLR
- Many newly developed hardware and algorithms
- Tight development schedule
- Traditional V-model for development infeasible
- Need to develop different levels of software stack simultaneously
    - Development against moving target



Develop Flight Software using CI/CD

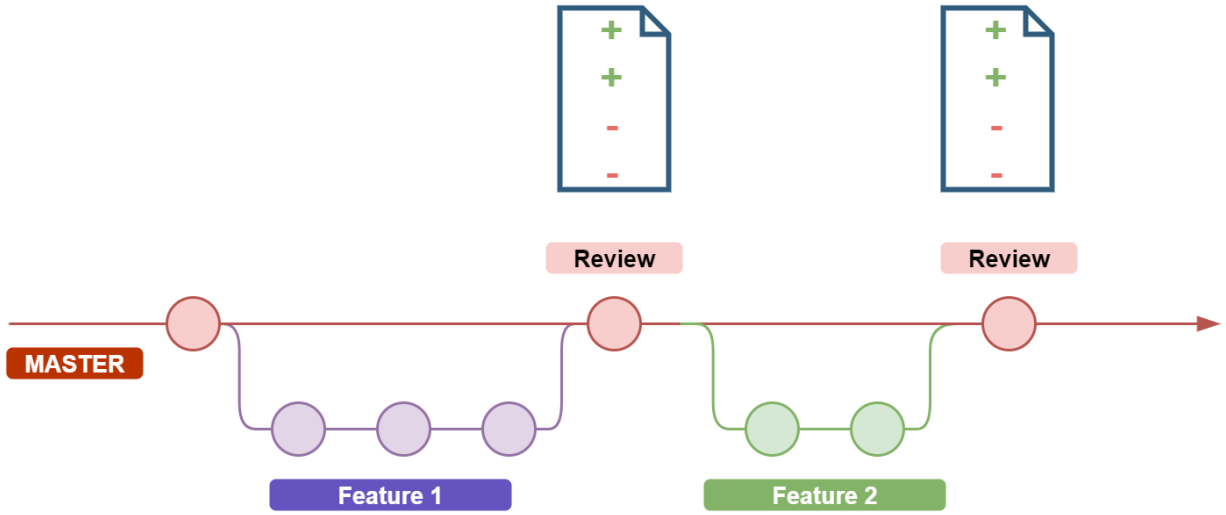
Is that really something new?

Technically not, but how common is:

- **Version Control System used**
- Regular build jobs
- Automatic build job before merging

- **Unit tests**
- Unit tests executed regularly
- Unit tests mandatory in review process
- Code coverage analysis in review process

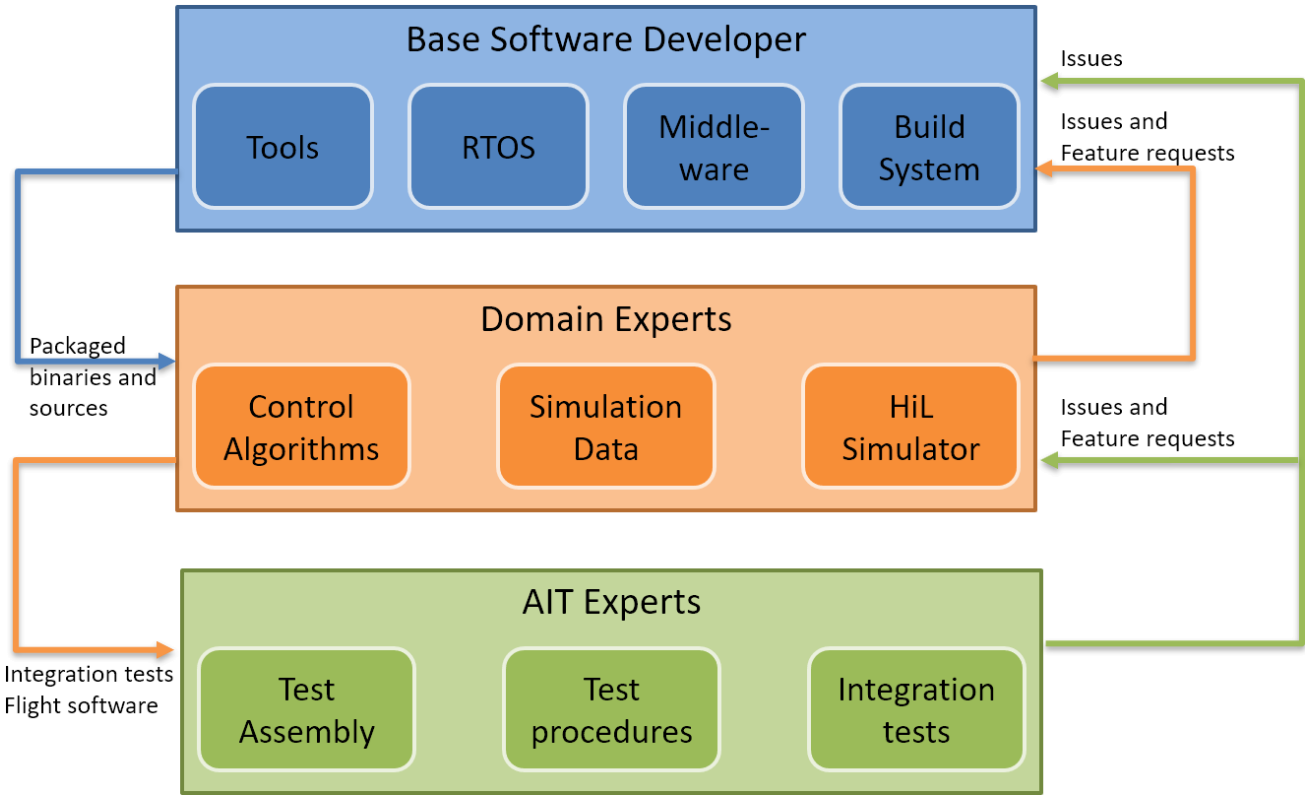
- Automatic static analysis
- Code climate report in review process



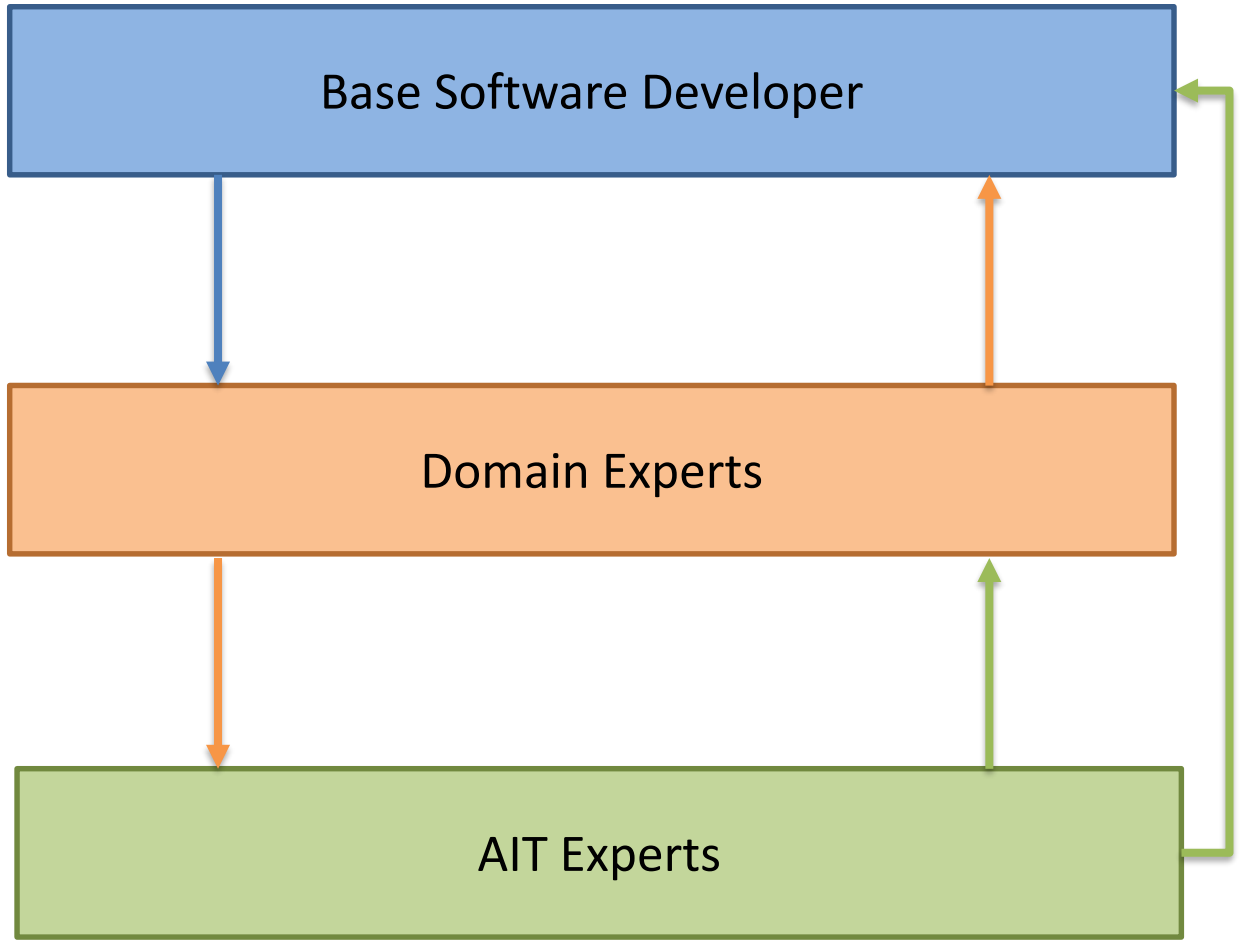
Develop flight software using CI/CD

Key: Communication and Automation

- Central tool for communication regarding S/W
  - Accessible for all developers
- CI/CD system to help with day-to-day tasks
- **DLR Gitlab:**
  - Maintained by IT department
  - Most members generally familiar
  - Provides many features to build upon



Requirements for the Software Development Infrastructure



- Manage access to repositories
- Provide central place for issue tracking
- Facilitate code reviews, milestone planning, developer documentation
- Continuous integration setup
- Release management

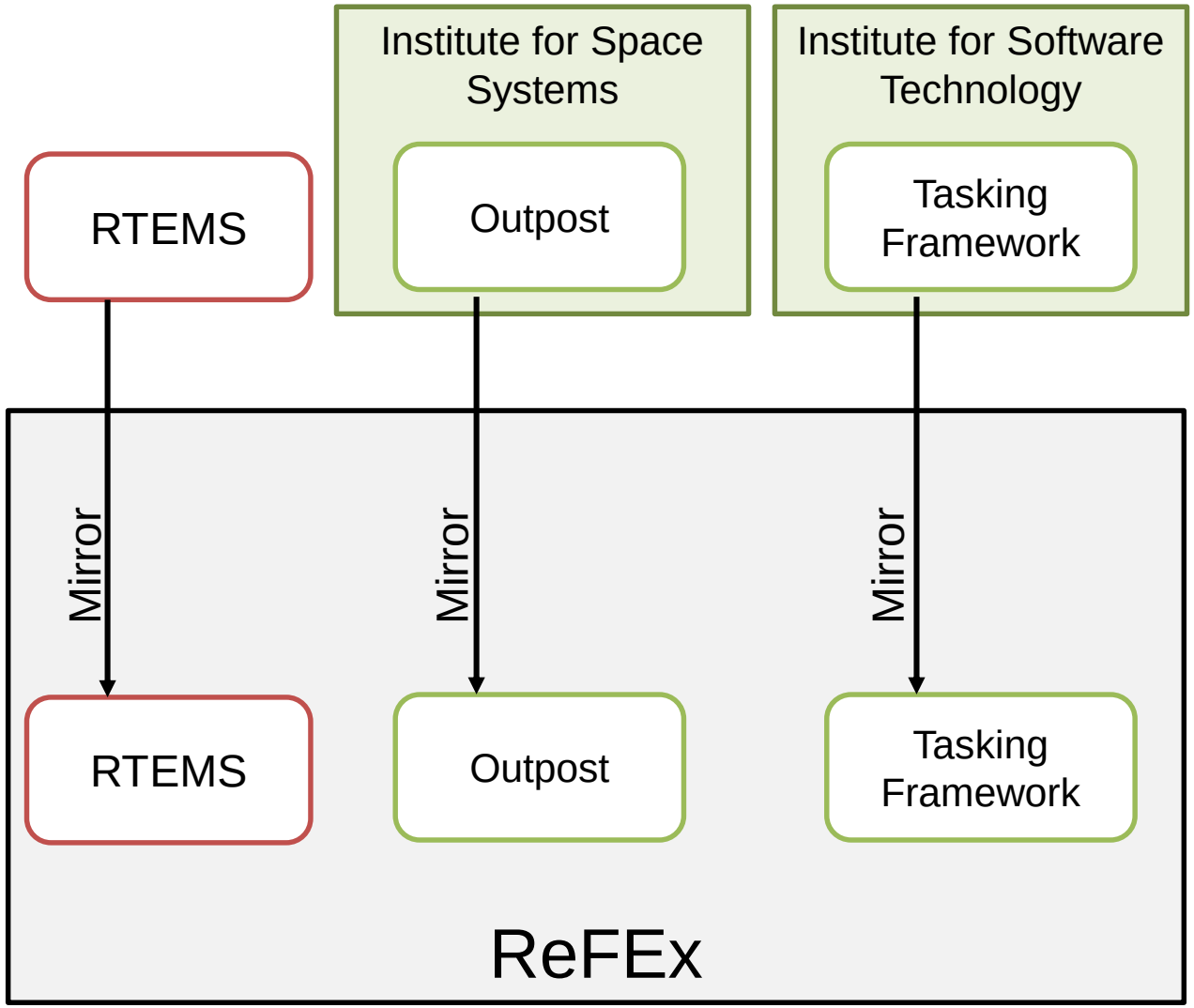
→ Use the DLR Gitlab instance

Repository Organization

- Mirrors of external and internal projects
  - Collect related repositories into groups
  - Monitor changes of upstream projects
  - Internal issue tracking and release management
  - Integration with internal CI subsystem
- Central package repository for ReFEx
- Central Wiki for software related information
- Repositories for ReFEx software development

Reuse of Existing Software

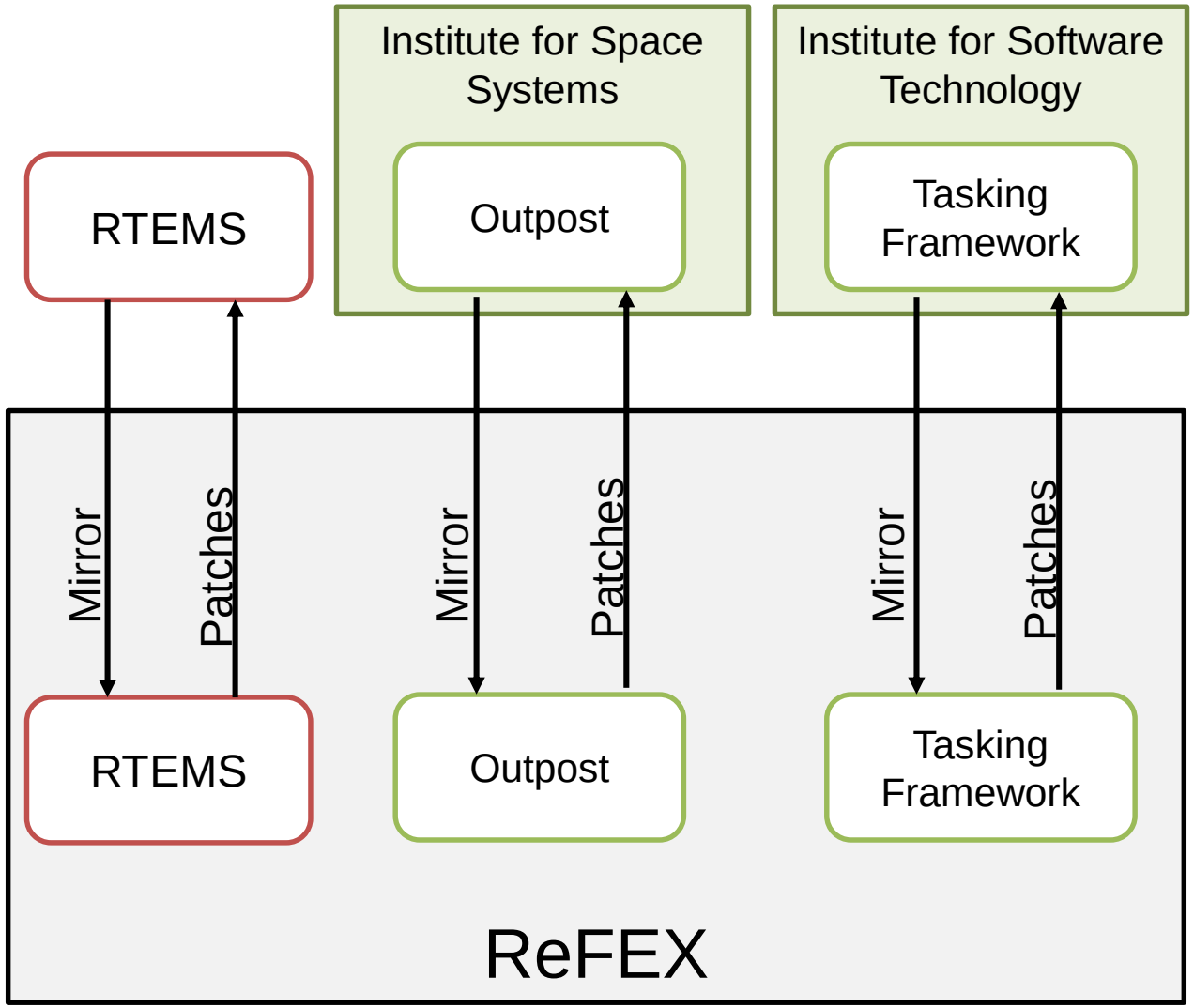
- Mirrors of external and internal projects
  - Have all software repositories in on place
  - Ensure that all project members have access
  - Branches for project specific modifications



Reuse of Existing Software

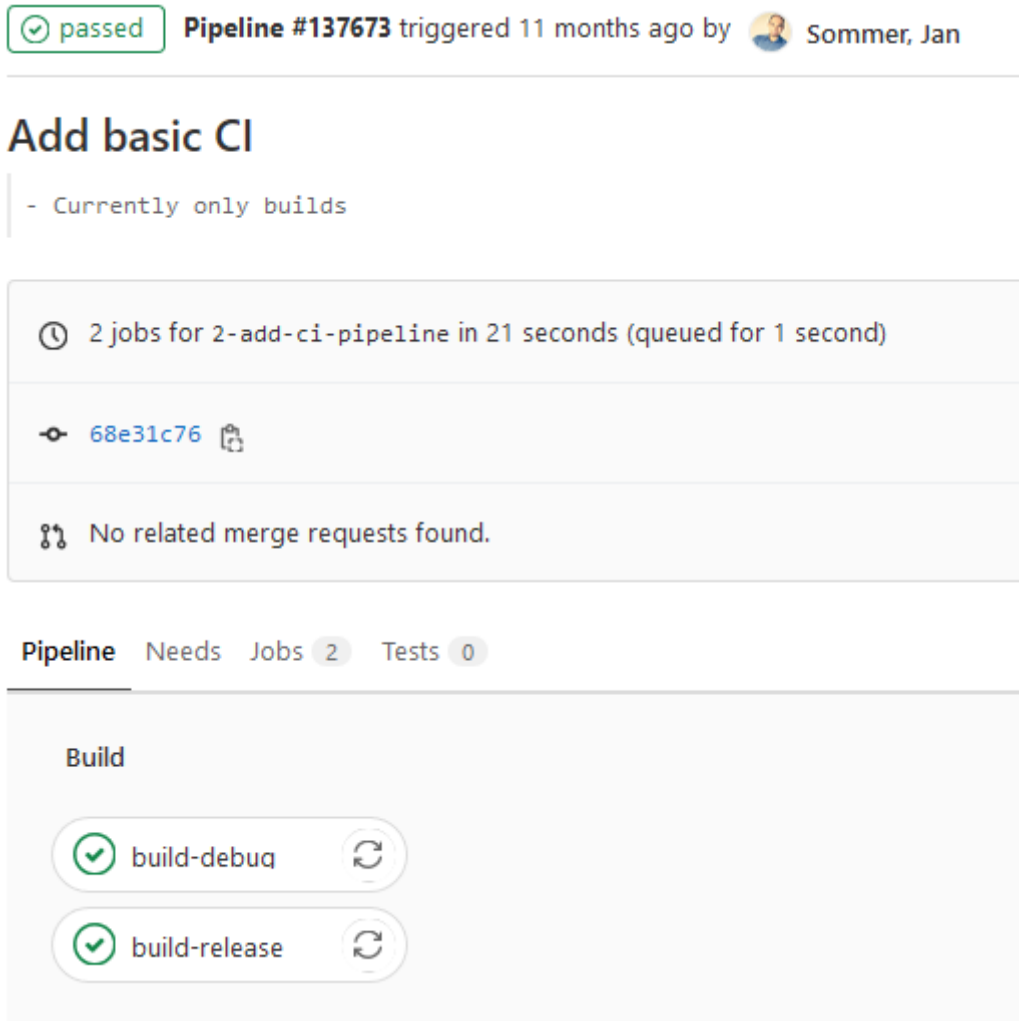
- Mirrors of external and internal projects
  - Have all software repositories in on place
  - Ensure that all project members have access
  - Branches for project specific modifications

- Contribute back relevant improvements to upstream
  - Assign a developer responsible for coordination for each mirrored repository
  - Go the extra-mile, polish changes and submit
  - Get a “free” review for submitted patches



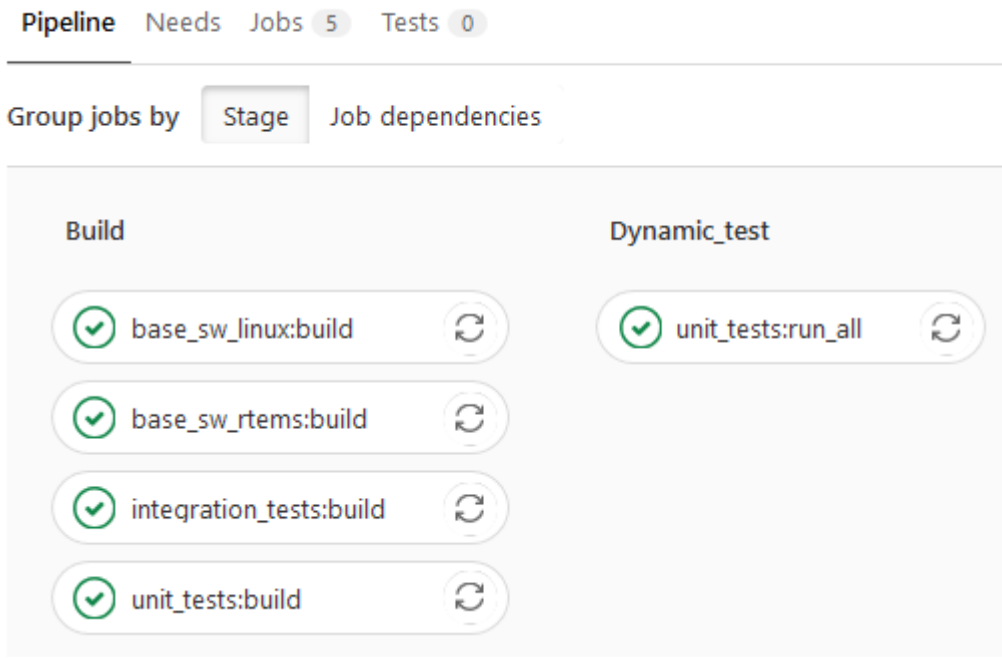
Continuous Integration – Start small and extend

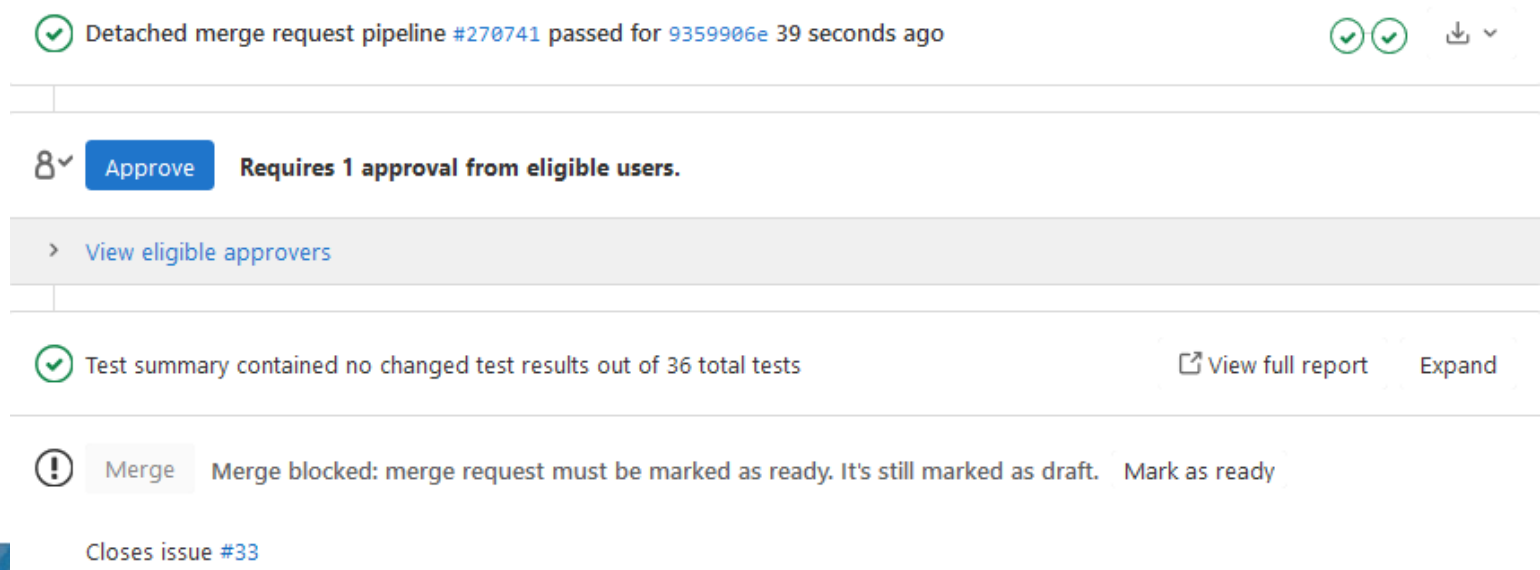
- Build job checks for compilation without issues
  - Provide pre-compiled binaries as artifacts
  - Avoids “works-on-my-machine” errors



Continuous Integration – Start small and extend

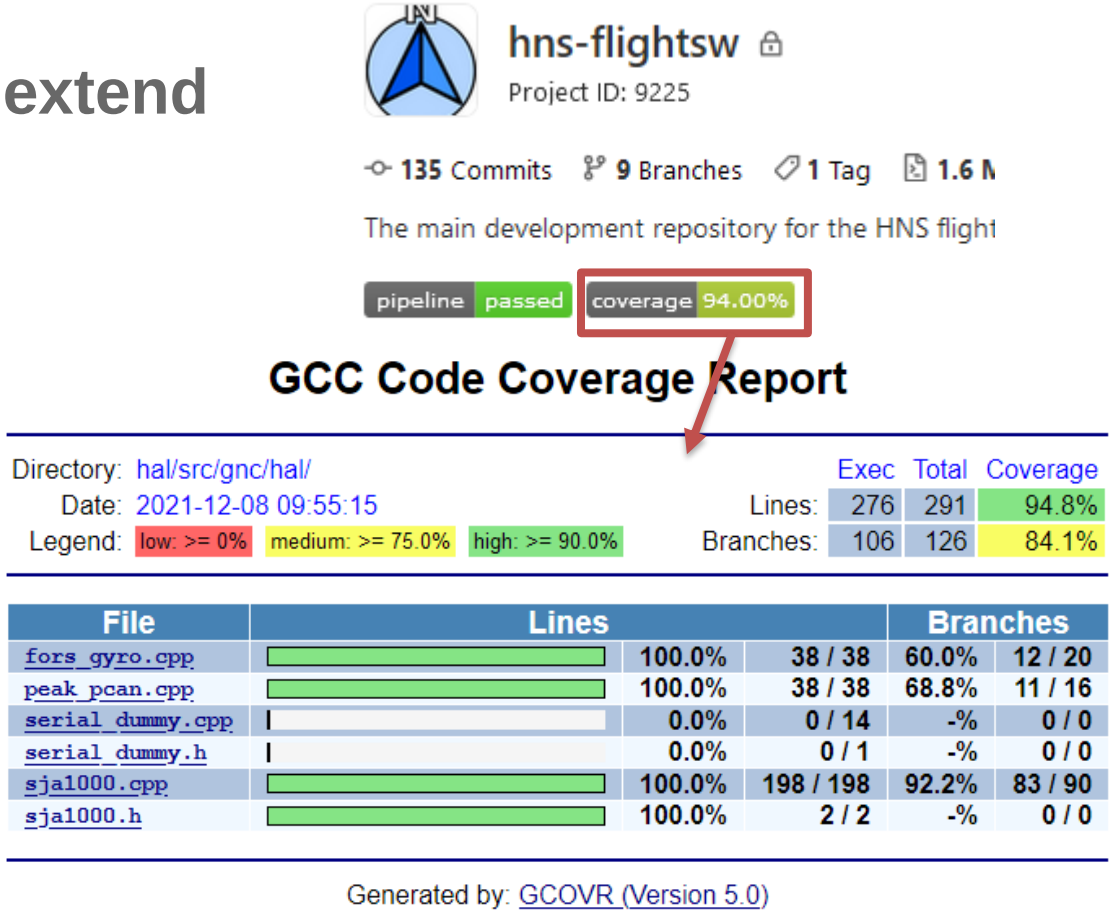
- Build job checking compilation without issues
  - Provide pre-compiled binaries as artifacts
- Execute unit tests and check test execution
  - Collect and display detailed test statistics





Continuous Integration – Start small and extend

- Build job checking compilation without issues
  - Provide pre-compiled binaries as artifacts
- Execute unit tests and check test execution
  - Collect and display detailed test statistics
- Compile and run unit tests with code-coverage
  - Publish generated html statistics as artifact



hns-flightsw

Project ID: 9225

135 Commits 9 Branches 1 Tag 1.6 M

The main development repository for the HNS flight

pipeline passed coverage 94.00%

GCC Code Coverage Report

Directory: hal/src/gnc/hal/
Date: 2021-12-08 09:55:15
Legend: low: >= 0% medium: >= 75.0% high: >= 90.0%

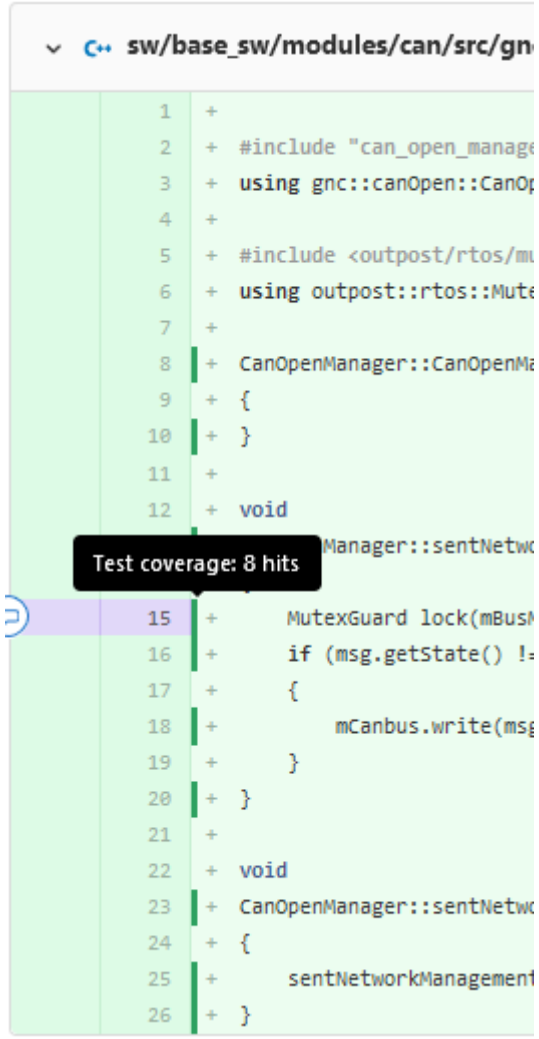
	Exec	Total	Coverage
Lines:	276	291	94.8%
Branches:	106	126	84.1%

File	Lines			Branches	
fors_gyro.cpp		100.0%	38 / 38	60.0%	12 / 20
peak_pcan.cpp		100.0%	38 / 38	68.8%	11 / 16
serial_dummy.cpp		0.0%	0 / 14	-%	0 / 0
serial_dummy.h		0.0%	0 / 1	-%	0 / 0
sja1000.cpp		100.0%	198 / 198	92.2%	83 / 90
sja1000.h		100.0%	2 / 2	-%	0 / 0

Generated by: GCOVR (Version 5.0)

Continuous Integration – Start small and extend

- Build job checking compilation without issues
  - Provide pre-compiled binaries as artifacts
- Execute unit tests and check test execution
  - Collect and display detailed test statistics
- Compile and run unit tests with code-coverage
  - Publish generate html statistics as artifact
  - Integrate the code-coverage into code review
- Compile documentation (if any)

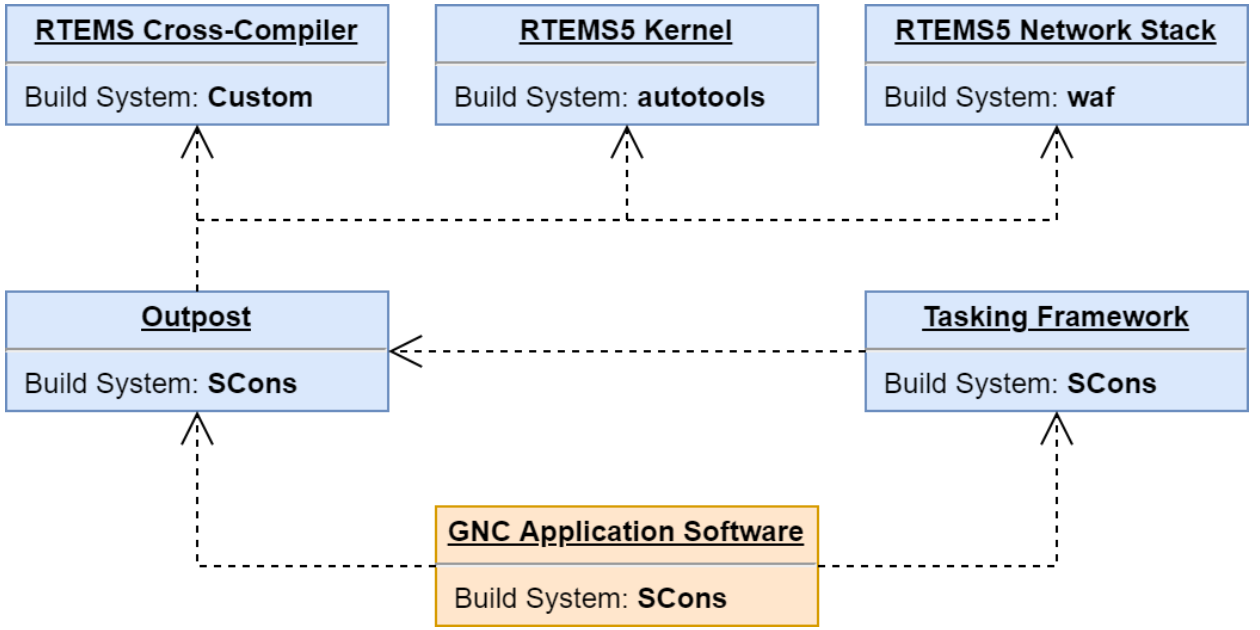


Continuous Integration – Extend even further

- Check for correct code formatting
- Run static analyzer
  - Provide results in code review
- Run memcheck
  - Provide results in code review
- Create and publish release
  - Prepare packages

Conan: Dependency Management and Package Delivery

- Works similar to a Linux package manager with recursive dependency resolution
- Integrated in Gitlab (some features missing)
  - Easy to upload from Gitlab-CI job
  - Same access management as for repositories
- Also allows easy delivery of cross-compiler tools
- Upload binaries for different HW platforms
- Supports different channels, e.g. *release*, *latest*
- Agnostic to build system for producer and consumer



Conan: Dependency Management in Practice

- *conanfile.txt* as part of the repository
  - Trace the required version for each dependency at every point in time
  - Switch to latest development version by changing the channel (e.g. *latest*)

conanfile.txt 135 Bytes

Edit | Web IDE | Replace | Delete

```
1  [requires]
2  gnc-i386-rtems5-gcc/3.0.1@GNC/release
3  gnc-rtems5/3.0.1@GNC/release
4  gnc-rtems5-libbsd/3.0.1@GNC/release
5
6  [generators]
7  scons
8
```

Conan: Dependency Management in Practice

- *conanfile.txt* as part of the repository
  - Trace the required version for each dependency at every point in time
  - Switch to latest development version by changing the channel (e.g. *latest*)

- Integrate into build system:
  - Check Conan dependencies, update if needed
  - Consume generated configuration file, e.g.:
    - SCons: *SConscript_conan*
    - CMake: *conanbuildinfo.cmake*

- No detailed knowledge about dependencies needed
  - Install cross-compiler toolchain automatically

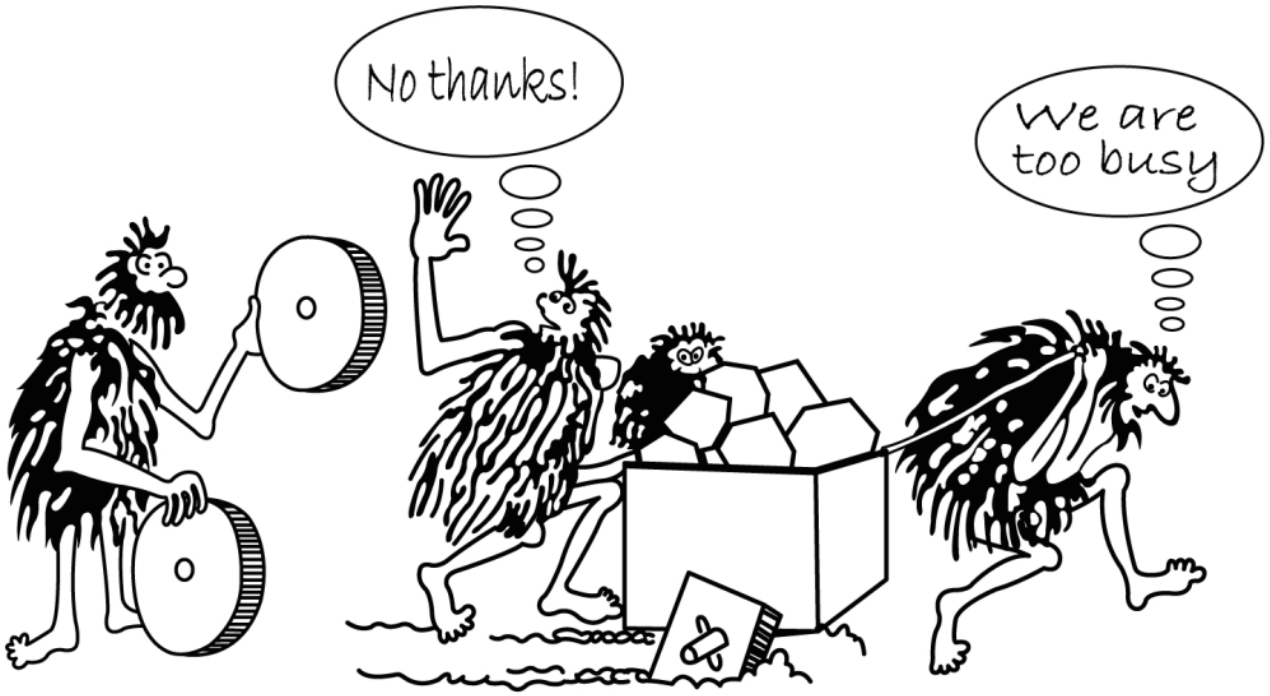
```
~) make
conan install . --remote gitlab
Configuration:
[settings]
arch=x86_64
arch_build=x86_64
build_type=Release
compiler=gcc
compiler.libcxx=libstdc++11
compiler.version=9
os=Linux
os_build=Linux
[options]
[build_requires]
[env]

gnc-rtems5-libbsd/3.0.1@GNC/release: Retrieving from server 'gitlab'
gnc-rtems5-libbsd/3.0.1@GNC/release: Trying with 'gitlab'...
Downloading conanmanifest.txt completed [0.06k]
Downloading conanfile.py completed [2.52k]
gnc-rtems5-libbsd/3.0.1@GNC/release: Downloaded recipe revision 0
conanfile.txt: Installing package
Requirements
    gnc-i386-rtems5-gcc/3.0.1@GNC/release from 'gitlab' - Cache
    gnc-rtems5/3.0.1@GNC/release from 'gitlab' - Cache
    gnc-rtems5-libbsd/3.0.1@GNC/release from 'gitlab' - Downloaded
Packages
    gnc-i386-rtems5-gcc/3.0.1@GNC/release:24647d9fe8ec489125dfbae4b3ebefaf7581674c - Cache
    gnc-rtems5/3.0.1@GNC/release:fbd406a4ab1ca324f54d11285bb4ceb859556476 - Cache
    gnc-rtems5-libbsd/3.0.1@GNC/release:24647d9fe8ec489125dfbae4b3ebefaf7581674c - Download

Installing (downloading, building) binaries...
gnc-rtems5-libbsd/3.0.1@GNC/release: Retrieving package 24647d9fe8ec489125dfbae4b3ebefaf7581674c from remote 'gitlab'
Downloading conanmanifest.txt completed [72.20k]
Downloading conaninfo.txt completed [0.25k]
Downloading conan_package.tgz completed [41316.92k]
Decompressing conan_package.tgz completed [0.00k]
gnc-rtems5-libbsd/3.0.1@GNC/release: Package installed 24647d9fe8ec489125dfbae4b3ebefaf7581674c
gnc-rtems5-libbsd/3.0.1@GNC/release: Downloaded package revision 0
gnc-i386-rtems5-gcc/3.0.1@GNC/release: Already installed!
gnc-rtems5/3.0.1@GNC/release: Already installed!
conanfile.txt: Generator scons created SConscript_conan
conanfile.txt: Generator txt created conanbuildinfo.txt
conanfile.txt: Aggregating env generators
conanfile.txt: Generated conaninfo.txt
conanfile.txt: Generated graphinfo

[PASS] Installed dependencies via conan!

make -C integration_tests
```

Lessons Learned

- Don’t expect getting it “right” the first time
  - Start with the low hanging fruits
    - Compile test
    - Code formatting check

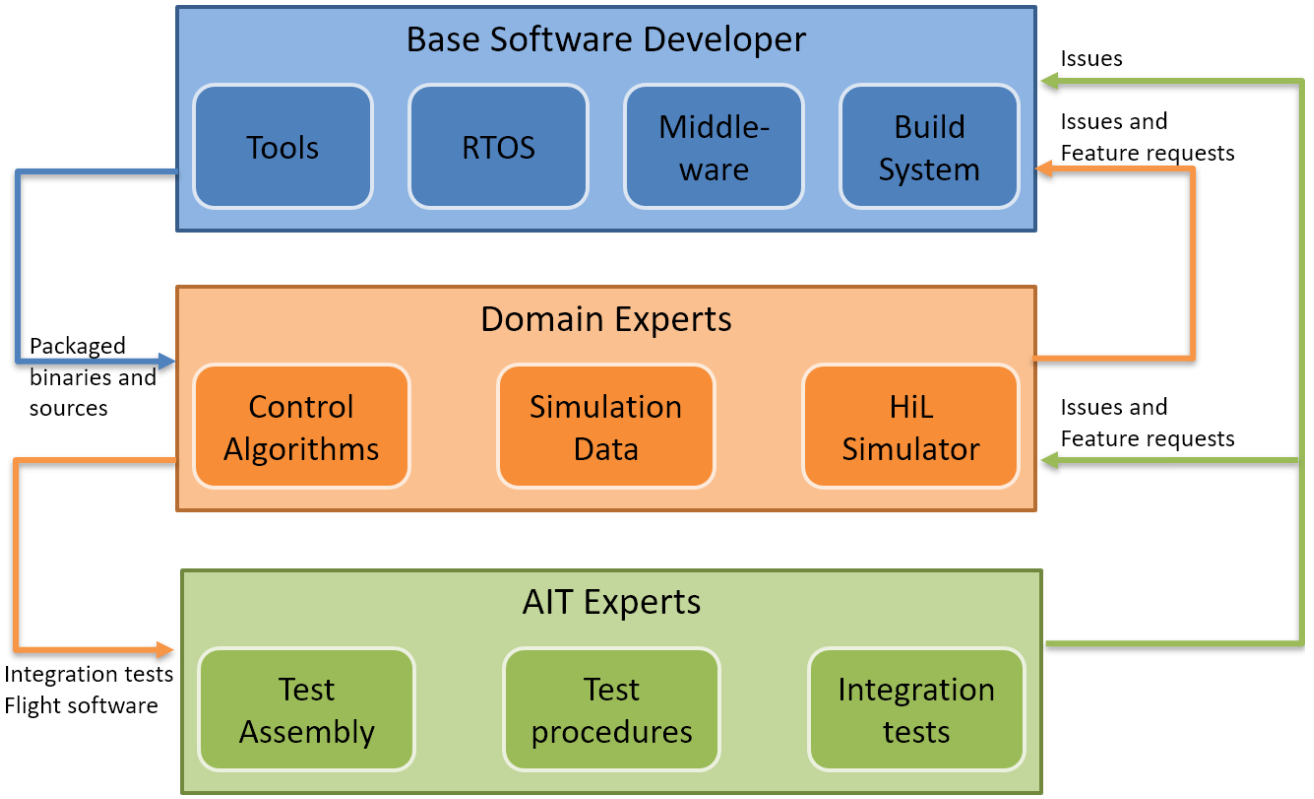
- Expect spending a significant amount of time at first
  - Setting up CI pipeline scripts
  - Integration into build system
  - Writing package recipes
  - Automatic release publication

- Continuous maintenance
  - Infrastructure changes (e.g. updates of Gitlab)
  - New tools emerge
  - Priorities change



Summary

- Need to develop flight software for ReFEx
  - Support newly developed hardware
  - Concurrent development in different S/W layers
  - Limited resources in time and developers
- Teams are distributed in institutes across Germany
  - Nearly 100% home office since March 2020
- Setup a central tool for S/W development
- CI for code checks and package delivery
- Next step: Containerization



References

- [1] R. Schwarz *et al.*, '*Overview of Flight Guidance, Navigation, and Control for the DLR Reusability Flight Experiment (ReFEx)*', presented at the 8TH EUROPEAN CONFERENCE FOR AERONAUTICS AND SPACE SCIENCES (EUCASS), Madrid, Spain, Jul. 2019.
- [2] RTEMS Project: https://www.rtems.org
- [3] Outpost Project: https://github.com/DLR-RY/outpost-core
- [4] Tasking Framework: https://github.com/DLR-SC/tasking-framework/