# SIP4C/C++ @ CERN – STATUS & LESSONS LEARNED

BE
Beams Department

S. Jensen, J-C Bau, A. Dworak, M. Gourber-Pace, F. Hoguin, J. Lauener, F. Locci, K. Sigerud, W. Sliwinski, CERN, Geneva, Switzerland

## Objectives

Establish best software quality practices and integrate chosen tools in the software development process

## Results

We show the uptake of tools in our development process, along with findings reported by 4 different software projects

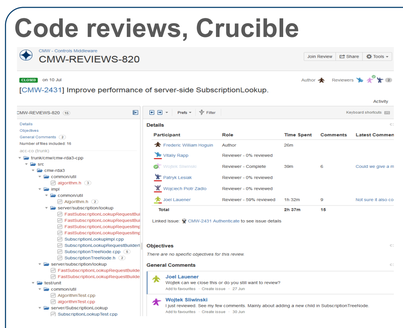
## Legend

4 participating projects

= Strong
= Medium
= Weak
= None

**Pros:** Values as reported by projects
**Cons:** Challenges as reported by projects

### Code reviews, Crucible

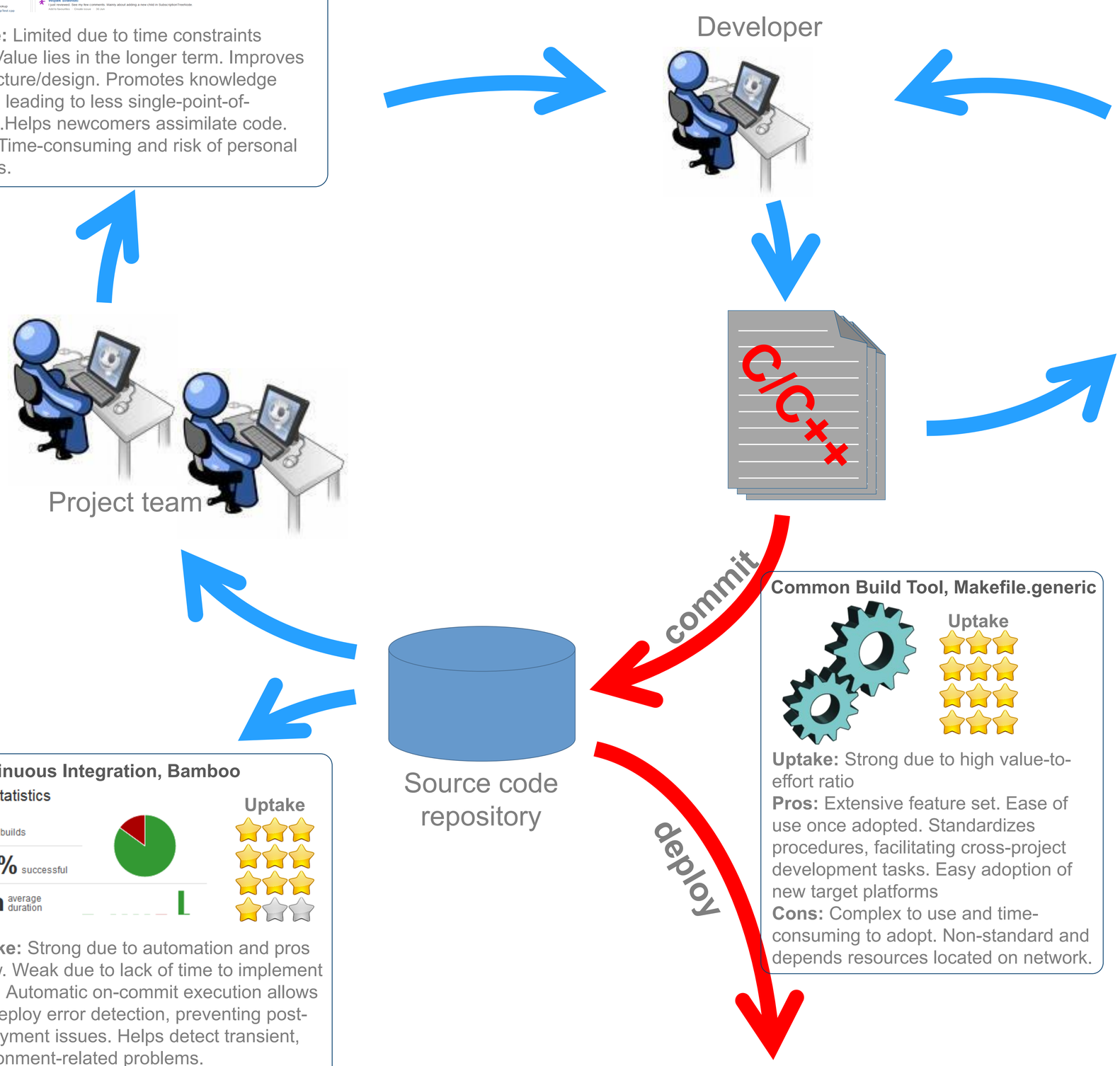



Uptake

**Uptake:** Limited due to time constraints
**Pros:** Value lies in the longer term. Improves architecture/design. Promotes knowledge sharing leading to less single-point-of-failures.Helps newcomers assimilate code.
**Cons:** Time-consuming and risk of personal conflicts.

## Conclusions

4 years of applying SIP4C/C++ at CERN shows increased code robustness and developer cross-project knowledge.




### Common Build Tool, Makefile.generic

Uptake

**Uptake:** Strong due to high value-to-effort ratio
**Pros:** Extensive feature set. Ease of use once adopted. Standardizes procedures, facilitating cross-project development tasks. Easy adoption of new target platforms
**Cons:** Complex to use and time-consuming to adopt. Non-standard and depends resources located on network.

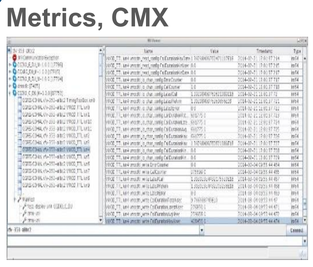
### Continuous Integration, Bamboo




Uptake

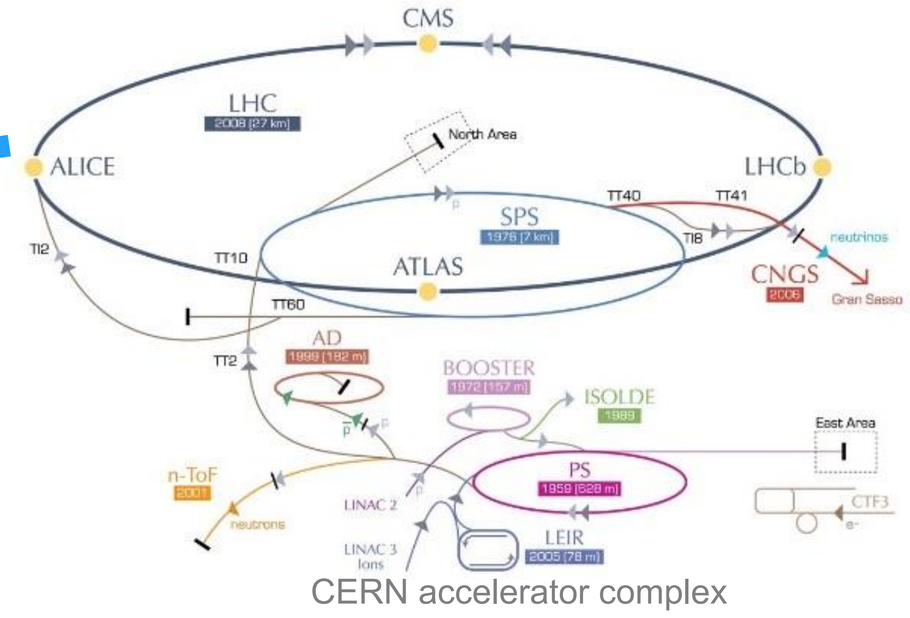
**Uptake:** Strong due to automation and pros below. Weak due to lack of time to implement
**Pros:** Automatic on-commit execution allows pre-deploy error detection, preventing post-deployment issues. Helps detect transient, environment-related problems.
**Cons:** Implementation and maintenance is time-consuming

### Metrics, CMX



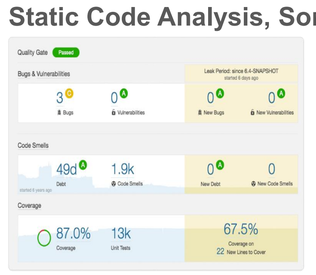
Uptake

**Uptake:** Limited due to cons below
**Pros:** Provides non-intrusive diagnostic on running process. Particularly valuable in libraries used by external users.
**Cons:** Lacks integration into control system stack to see metric trends over time. Lack command/reset functionality.




CERN accelerator complex

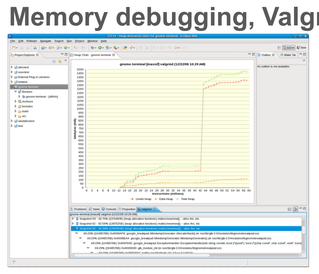
### Static Code Analysis, SonarQube




Uptake

**Uptake:** None due to cons below
**Pros:** Helps clean up existing code. Appealing reports
**Cons:** Not yet well integrated. Requires project-specific configuration. Produces many false-positives in existing code base.

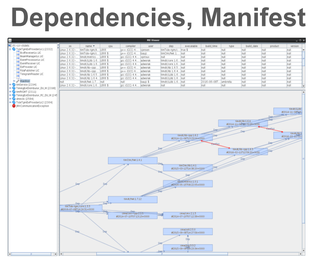
### Memory debugging, Valgrind



Uptake

**Uptake:** Limited due to cons below
**Pros:** Detects slower forms of memory leaks not found in unit tests.
**Cons:** Many false-positives related to shared-memory use and self-induced timeouts.

### Dependencies, Manifest file



Uptake

**Uptake:** Limited due to pre-existing solutions as well as cons below
**Pros:** Helps understand certain run-time problems.
**Cons:** Problems resolving symbolic links. Lack of notification-on-conflict. Non-standard.

### Unit tests, Google test

| | | | |
|---|---|---|---|
| timdt-core | #297 | 4 months ago | 810 passed |
| timdt-gmock-core | #51 | 4 months ago | No tests found |
| timdt-gmock-lib | #49 | 4 months ago | No tests found |
| timdt-gmock-LLlib | #38 | 4 months ago | No tests found |
| timdt-lib-c | #228 | 4 months ago | 186 passed |
| timdt-lib-cpp | #228 | 4 months ago | 3 passed |
| timdt-LLlib | #283 | 4 months ago | 528 passed |
| timdt-metrics | #10 | 4 months ago | No tests found |
| timdt-rtGMT | #301 | 2 months ago | 309 passed |
| timdt-service | #255 | 4 months ago | 378 passed |
| timdt-timloadGMT | #217 | 4 months ago | 483 passed |

Uptake

**Uptake:** Limited due to time constraints
**Pros:** It makes developers think in terms of testable cases. Tests often ramify, thereby solidifying code. Helps prevent regression and detect bugs. Appeals to younger developers.
**Cons:** Time-consuming to implement (~1:1 ratio LOC). Risk of over-confidence if tests pass.

## Future plans

Decrease barriers to adopting tools and procedures. Specifically, improve the build/release chain in terms of usability and feature set – in particular with respect to dependency management

**Author**
e-mail: steen.jensen@*cern.ch*
CERN BE-CO

ICALEPCS2017
Barcelona · Spain, October 8-13 · Palau de Congressos de Catalunya