Testing? Testing. Testing!

How RSEs can Assure Software Quality in Complex HPC Code Bases

July 11, 2023 | Ivo Kabadshow | PASC 2023 | Jülich Supercomputing Centre

Software Engineering

Why do we need it?

> *The code you write makes you a programmer.*
> *The code you delete makes you a good one.*
> *The code you don’t have to write makes you a great one.*

Mario Fusco (Principal Software Engineer)

Why Does Software Engineering Matter?

I/II

The usual way of extending code in HPC

- Algorithm

- MPI
- Threading/Tasking
- ILP/Unrolling
- Vectorization
- GPU-Offloading

Why Does Software Engineering Matter?

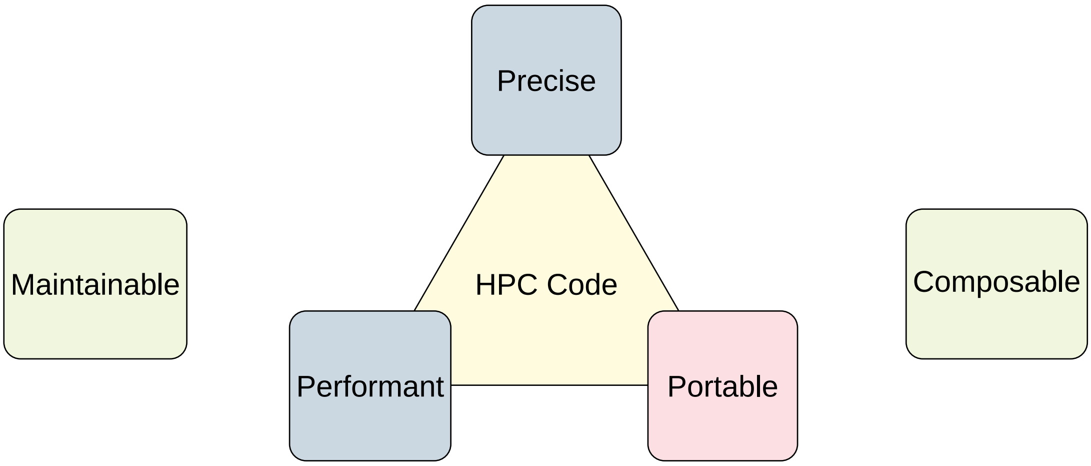
I/II

Parallelization is hidden in different layers (find the right abstractions)

- Algorithm

- MPI
- Threading/Tasking
- ILP/Unrolling
- Vectorization
- GPU-Offloading

HPC Requirements


Precise

Maintainable

HPC Code

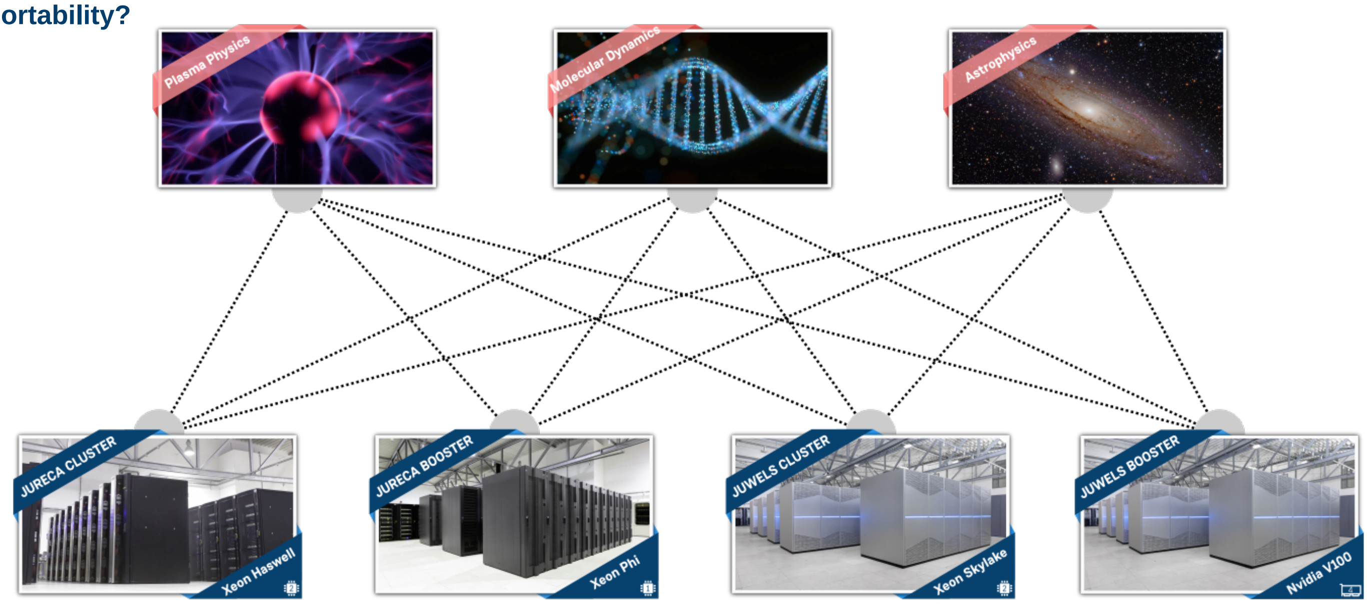
Composable

Performant

Portable

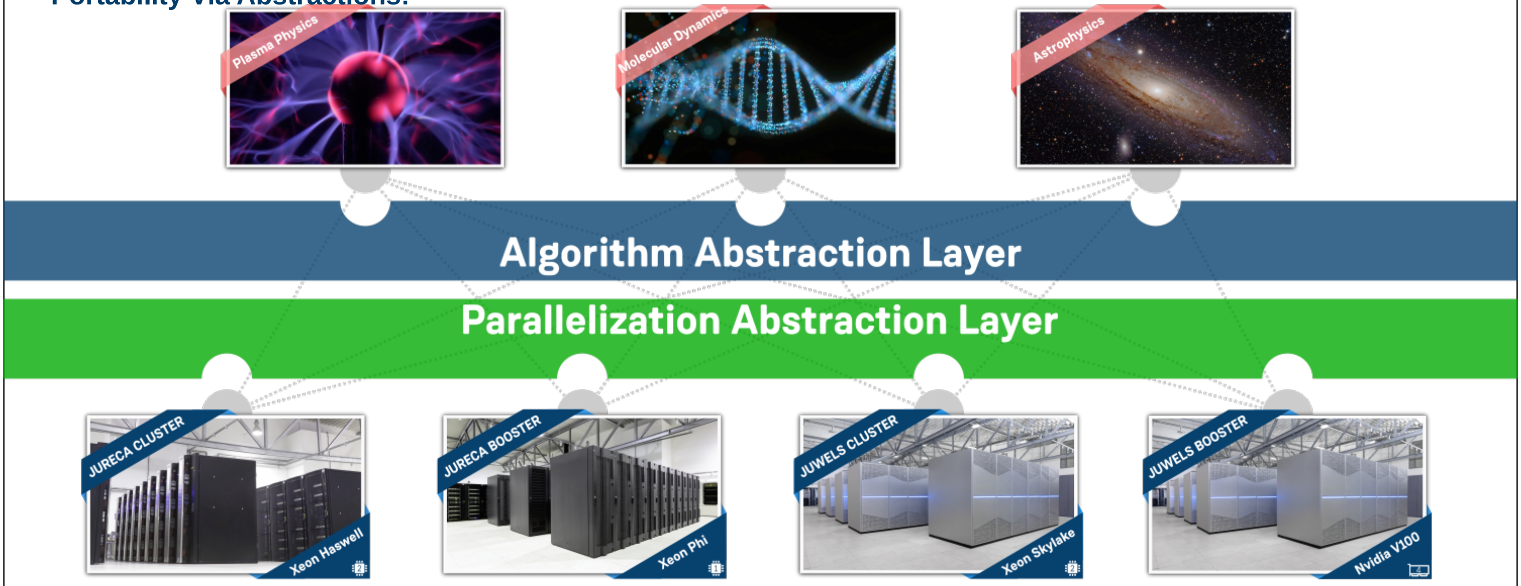
Why Does Software Engineering Matter? II/II

Portability?



Why Does Software Engineering Matter? II/II

Portability Via Abstractions!



Plasma Physics

Molecular Dynamics

Astrophysics

Algorithm Abstraction Layer

Parallelization Abstraction Layer

JURECA CLUSTER — Xeon Haswell

JURECA BOOSTER — Xeon Phi

JUWELS CLUSTER — Xeon Skylake

JUWELS BOOSTER — Nvidia V100

Challenges

Goal: Optimal time to solution on every platform

Flexibility

Hardware: CPU/GPU

- ILP, SIMD, OoOE
- Cache levels & sizes
- NUMA
- Threading & MPI

Configurability

Algorithm

- Different implementations
- Different critical paths

Customization

Application

- Physical model
- Accuracy range
- System size

Monolithic Softwarestack

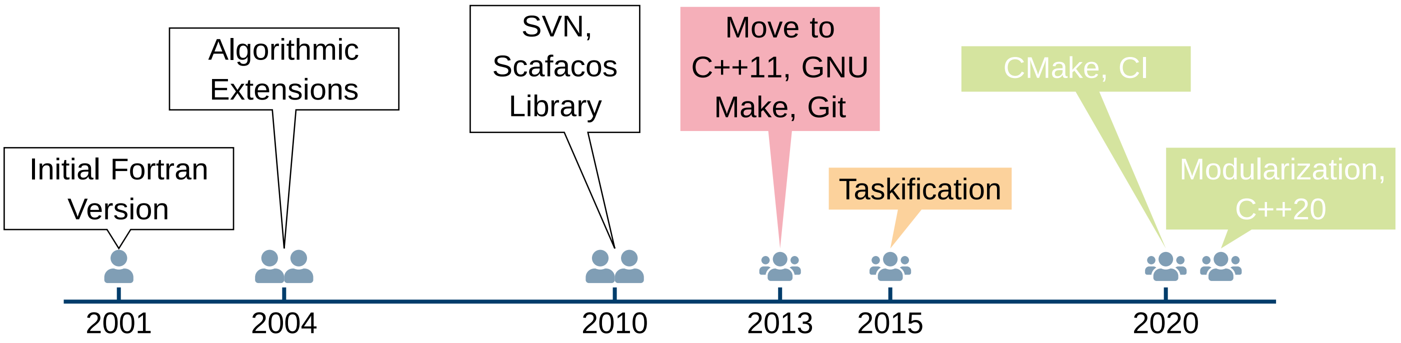
- m HW, n Applications $\longrightarrow m \times n$ dependencies

Modular Softwarestack

- m HW, n Applications $\longrightarrow m + n$ dependencies

Code Development Usecase: FMSolvr

Library: Fast Multipole Method for MD



Initial Fortran Version

Algorithmic Extensions

SVN, Scafacos Library

Move to C++11, GNU Make, Git

Taskification

CMake, CI

Modularization, C++20

2001

2004

2010

2013

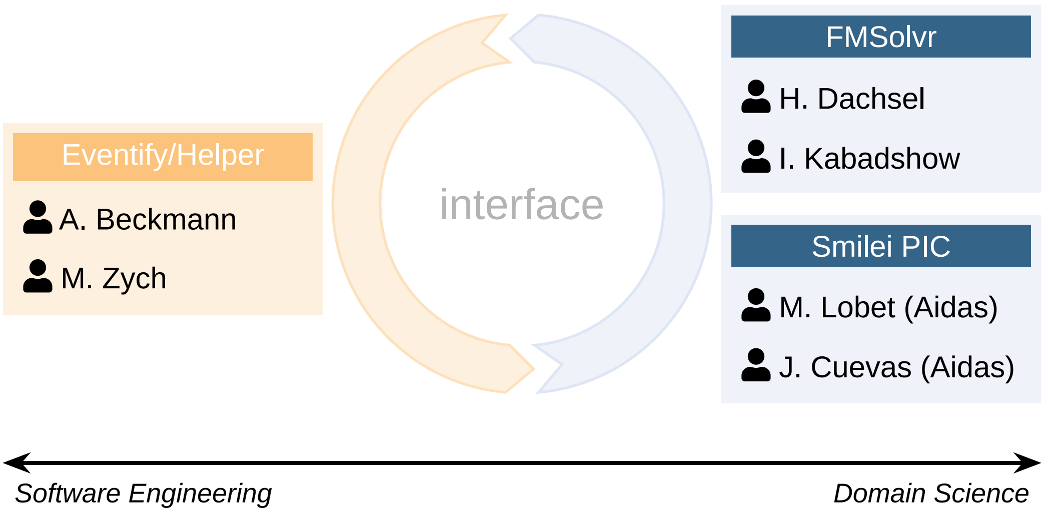
2015

2020

- Unittests of components
- Split into multiple separate/independent libraries (C++ template library, Eventify, FMSolvr)

Developer Roles

Extract reusable components from FMSolvr for Smilei PIC code



Eventify/Helper

A. Beckmann

M. Zych

interface

FMSolvr

H. Dachsel

I. Kabadshow

Smilei PIC

M. Lobet (Aidas)

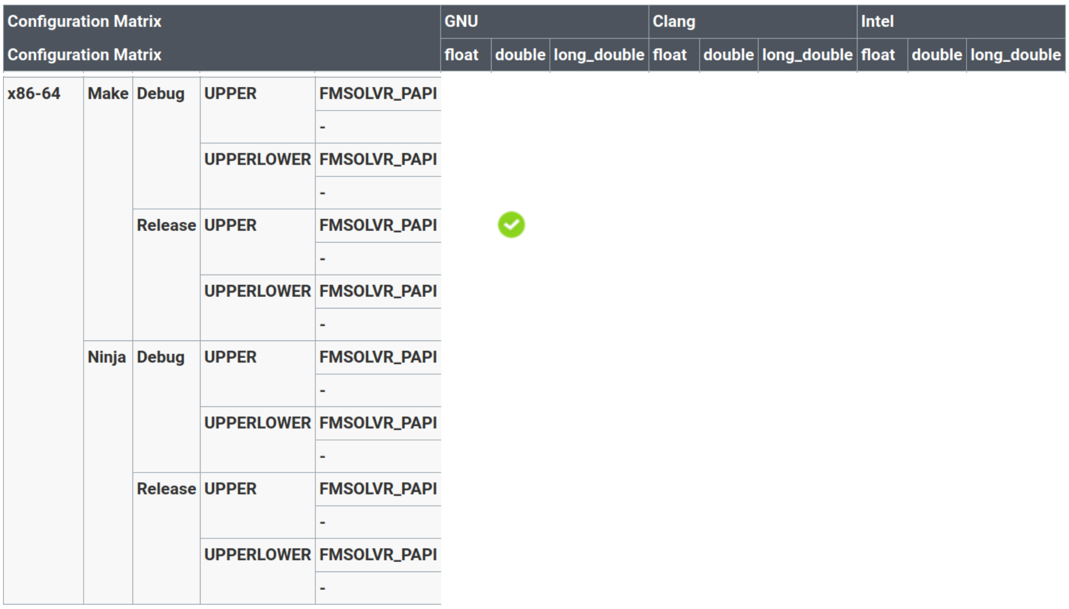
J. Cuevas (Aidas)

Software Engineering

Domain Science

Everything will be fine ...

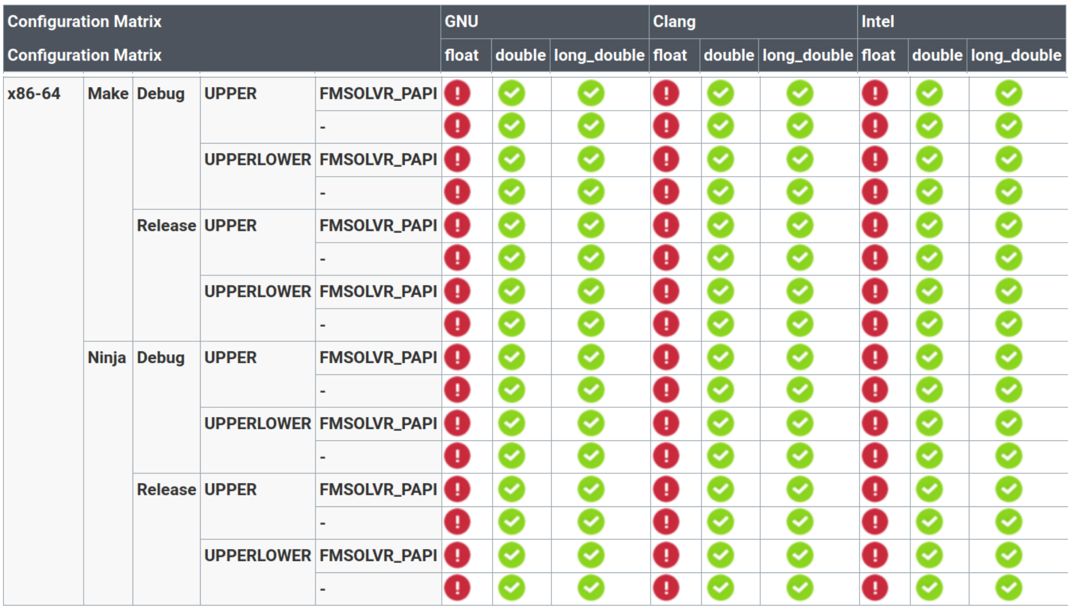
Single manual test on x86-64 with default compiler/settings



Configuration Matrix					GNU			Clang			Intel		
Configuration Matrix					float	double	long_double	float	double	long_double	float	double	long_double
x86-64	Make	Debug	UPPER	FMSOLVR_PAPI									
				-									
			UPPERLOWER	FMSOLVR_PAPI									
				-									
		Release	UPPER	FMSOLVR_PAPI		✓							
				-									
			UPPERLOWER	FMSOLVR_PAPI									
				-									
	Ninja	Debug	UPPER	FMSOLVR_PAPI									
				-									
			UPPERLOWER	FMSOLVR_PAPI									
				-									
		Release	UPPER	FMSOLVR_PAPI									
				-									
			UPPERLOWER	FMSOLVR_PAPI									
				-									

... or not

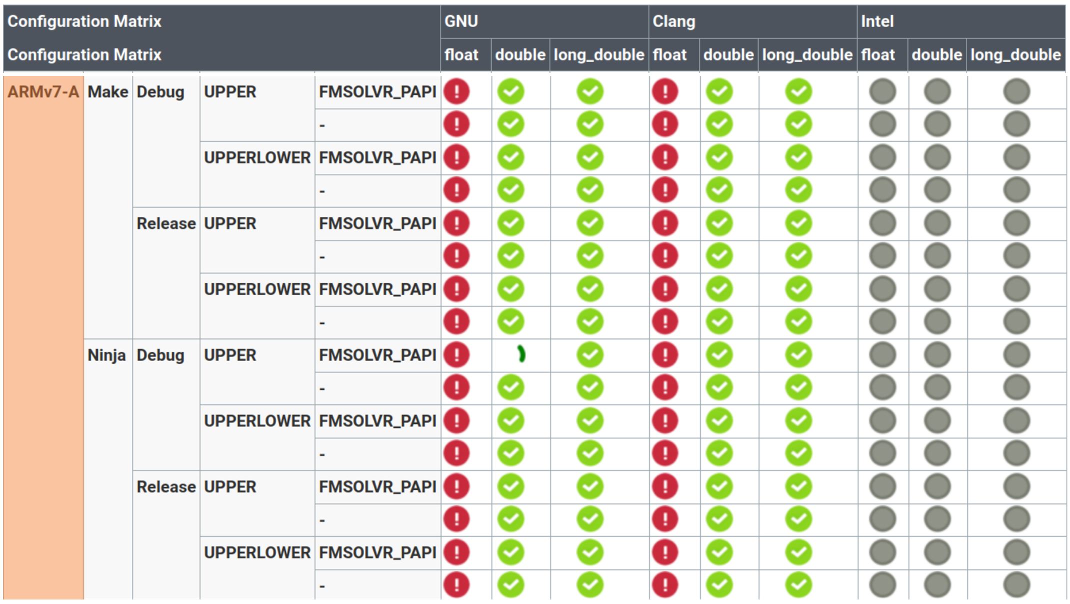
Extensive tests on x86-64



Configuration Matrix					GNU			Clang			Intel		
Configuration Matrix					float	double	long_double	float	double	long_double	float	double	long_double
x86-64	Make	Debug	UPPER	FMSOLVR_PAPI	!	✓	✓	!	✓	✓	!	✓	✓
				-	!	✓	✓	!	✓	✓	!	✓	✓
			UPPERLOWER	FMSOLVR_PAPI	!	✓	✓	!	✓	✓	!	✓	✓
				-	!	✓	✓	!	✓	✓	!	✓	✓
		Release	UPPER	FMSOLVR_PAPI	!	✓	✓	!	✓	✓	!	✓	✓
				-	!	✓	✓	!	✓	✓	!	✓	✓
			UPPERLOWER	FMSOLVR_PAPI	!	✓	✓	!	✓	✓	!	✓	✓
				-	!	✓	✓	!	✓	✓	!	✓	✓
	Ninja	Debug	UPPER	FMSOLVR_PAPI	!	✓	✓	!	✓	✓	!	✓	✓
				-	!	✓	✓	!	✓	✓	!	✓	✓
			UPPERLOWER	FMSOLVR_PAPI	!	✓	✓	!	✓	✓	!	✓	✓
				-	!	✓	✓	!	✓	✓	!	✓	✓
		Release	UPPER	FMSOLVR_PAPI	!	✓	✓	!	✓	✓	!	✓	✓
				-	!	✓	✓	!	✓	✓	!	✓	✓
			UPPERLOWER	FMSOLVR_PAPI	!	✓	✓	!	✓	✓	!	✓	✓
				-	!	✓	✓	!	✓	✓	!	✓	✓

... or not

Extensive tests on ARM



Configuration Matrix					GNU			Clang			Intel		
Configuration Matrix					float	double	long_double	float	double	long_double	float	double	long_double
ARMv7-A	Make	Debug	UPPER	FMSOLVR_PAPI	!	✓	✓	!	✓	✓	●	●	●
				-	!	✓	✓	!	✓	✓	●	●	●
			UPPERLOWER	FMSOLVR_PAPI	!	✓	✓	!	✓	✓	●	●	●
				-	!	✓	✓	!	✓	✓	●	●	●
		Release	UPPER	FMSOLVR_PAPI	!	✓	✓	!	✓	✓	●	●	●
				-	!	✓	✓	!	✓	✓	●	●	●
			UPPERLOWER	FMSOLVR_PAPI	!	✓	✓	!	✓	✓	●	●	●
				-	!	✓	✓	!	✓	✓	●	●	●
	Ninja	Debug	UPPER	FMSOLVR_PAPI	!	◔	✓	!	✓	✓	●	●	●
				-	!	✓	✓	!	✓	✓	●	●	●
			UPPERLOWER	FMSOLVR_PAPI	!	✓	✓	!	✓	✓	●	●	●
				-	!	✓	✓	!	✓	✓	●	●	●
		Release	UPPER	FMSOLVR_PAPI	!	✓	✓	!	✓	✓	●	●	●
				-	!	✓	✓	!	✓	✓	●	●	●
			UPPERLOWER	FMSOLVR_PAPI	!	✓	✓	!	✓	✓	●	●	●
				-	!	✓	✓	!	✓	✓	●	●	●

Combinatorial Explosion Of Possible Tests

Tests can easily reach into thousands → We cannot test everything

Hardware Features

- Floating Point Precision (float, double, long double, float128) ×4
- CPU Architecture (i686, x86-64, ARM, ARM64, RISC-V) [GPU/FPGA?] ×5
- Microarchitecture (SSE, AVX, AVX2, AVX512, Neon) ×5

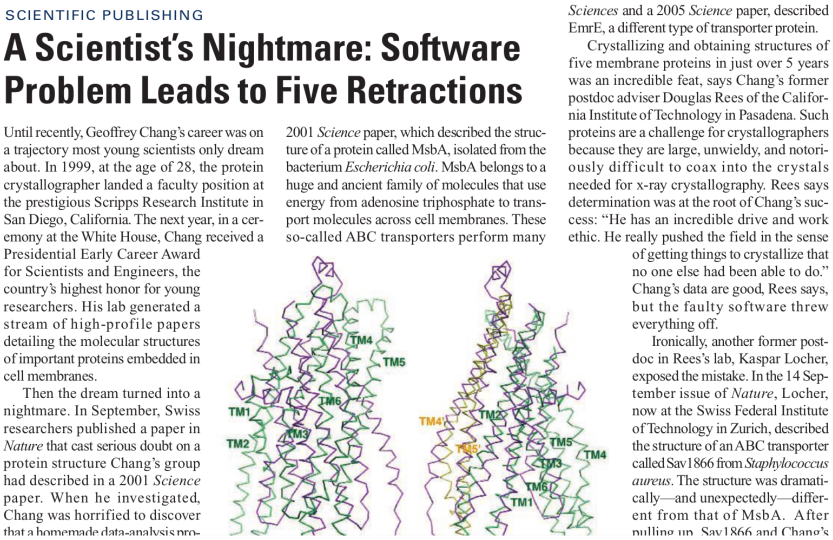
Build Environment

- Compiler (GNU, Clang, Intel) ×3
- Mode (Debug, Release) ×2
- Build system (GNU Make, Ninja) ×2
- Compiler Version

Why Tests?

1/2

DOI: 10.1126/science.314.5807.1856



SCIENTIFIC PUBLISHING

A Scientist's Nightmare: Software Problem Leads to Five Retractions

Until recently, Geoffrey Chang's career was on a trajectory most young scientists only dream about. In 1999, at the age of 28, the protein crystallographer landed a faculty position at the prestigious Scripps Research Institute in San Diego, California. The next year, in a ceremony at the White House, Chang received a Presidential Early Career Award for Scientists and Engineers, the country's highest honor for young researchers. His lab generated a stream of high-profile papers detailing the molecular structures of important proteins embedded in cell membranes.

Then the dream turned into a nightmare. In September, Swiss researchers published a paper in *Nature* that cast serious doubt on a protein structure Chang's group had described in a 2001 *Science* paper. When he investigated, Chang was horrified to discover that a homemade data-analysis pro-

2001 *Science* paper, which described the structure of a protein called MsbA, isolated from the bacterium *Escherichia coli*. MsbA belongs to a huge and ancient family of molecules that use energy from adenosine triphosphate to transport molecules across cell membranes. These so-called ABC transporters perform many

Sciences and a 2005 *Science* paper, described EmrE, a different type of transporter protein.

Crystallizing and obtaining structures of five membrane proteins in just over 5 years was an incredible feat, says Chang's former postdoc adviser Douglas Rees of the California Institute of Technology in Pasadena. Such proteins are a challenge for crystallographers because they are large, unwieldy, and notoriously difficult to coax into the crystals needed for x-ray crystallography. Rees says determination was at the root of Chang's success: "He has an incredible drive and work ethic. He really pushed the field in the sense of getting things to crystallize that no one else had been able to do." Chang's data are good, Rees says, but the faulty software threw everything off.

Ironically, another former postdoc in Rees's lab, Kaspar Locher, exposed the mistake. In the 14 September issue of *Nature*, Locher, now at the Swiss Federal Institute of Technology in Zurich, described the structure of an ABC transporter called Sav1866 from *Staphylococcus aureus*. The structure was dramatically—and unexpectedly—different from that of MsbA. After pulling up Sav1866 and Chang's

Retractions

- Simulation data was correct
- Analysis SW flipped two columns

Why Tests?

2/2

Fluctuating developer team

How persistent is your developer team?

- Mostly one core developer (staff)
- Master and PhD students (1-3 years)
- Guests (3-12 months)

Do you trust your developers unconditionally?

- Is the code correct in all required cases?
- Who do you ask, if a developer has left?
- Is the provided code in a reusable/extendable state?

Setting Up The Ecosystem

What do we need to make this work?

Automated Build Process and Dependency Management

- CMake, GNU Make, Ninja

Change Management + Continuous Integration Tools

- Version control (git, svn) + Ticket system (bug tracker)
- Test framework (Googletest, Catch 2)
- CI (Jenkins, Teamcity, gitlab)

Package Manager for C++ (optional)

- Conan
- vcpkg

What To Tests

Tests and Their Coverage

What to test?	Is	Should
Acceptance tests check if customer requirements are met on target environment	✓✓✓	✓
System tests check specified requirements on target environment	✓✓	✓
Integration tests/interface tests check interaction between certain modules and components	✓	✓✓
Module tests (unit/component tests) check specific unit (restrictions, constraints)	✓	✓✓✓

What about performance?

- Performance tests
  - check if performance requirements are met on target hardware

How to measure code quality?

Code Test Coverage

What is the quality of the software?

- Untested/uncovered code should be expected to be wrong
- If test cases are too complex, split the code further, introduce internal interfaces
- Test the smallest possible unit (e.g. functions)
- If every line is covered, bugs are likely to be found easily

ℹ Sensible tests are often better than outdated documentation

Test Automation

Continuous Integration

Git + Tests + CI

- Tests should run automatically for each commit
- Tests should be short and explain the correct usage of a piece of code
- Test matrix should be sensible

Code Review

- At least one branch (master) should be passing all tests
- Merges into the master branch should not be possible if tests fail

Open Issues

Sequential Tests

- Who is responsible to setup/maintain the infrastructure (RSEs?)

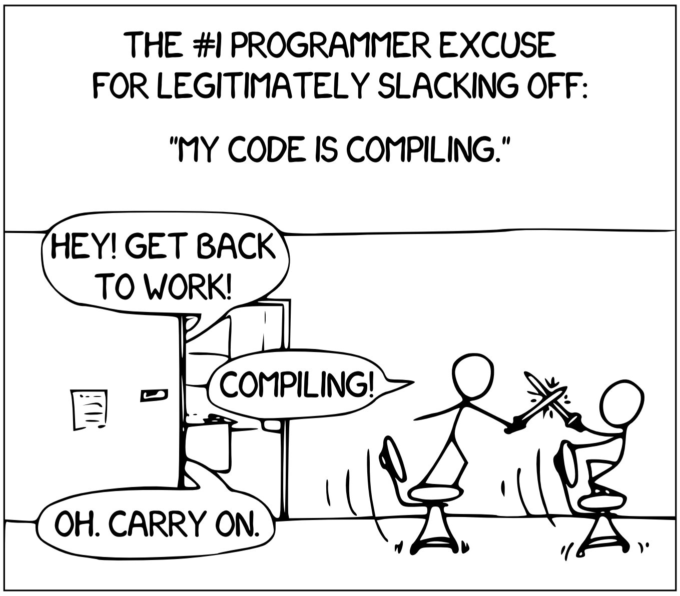
HPC & CI

- No CI access to cluster (personal ssh key, no dedicated resources)
- Hard to implement properly for every use case (compiler, os, tools, libraries)

What about users quota for tests on HPC resources?

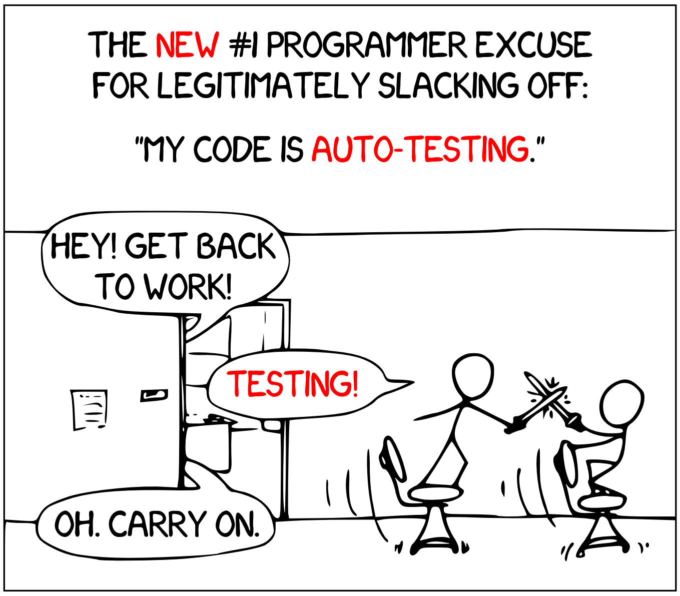
The Past

Based on https://xkcd.com/303/



The Future

Based on https://xkcd.com/303/


THE NEW #1 PROGRAMMER EXCUSE FOR LEGITIMATELY SLACKING OFF:
"MY CODE IS AUTO-TESTING."
HEY! GET BACK TO WORK!
TESTING!
OH. CARRY ON.

Testing? Testing. Testing!

How RSEs can Assure Software Quality in Complex HPC Code Bases

July 11, 2023 | Ivo Kabadshow | PASC 2023 | Jülich Supercomputing Centre