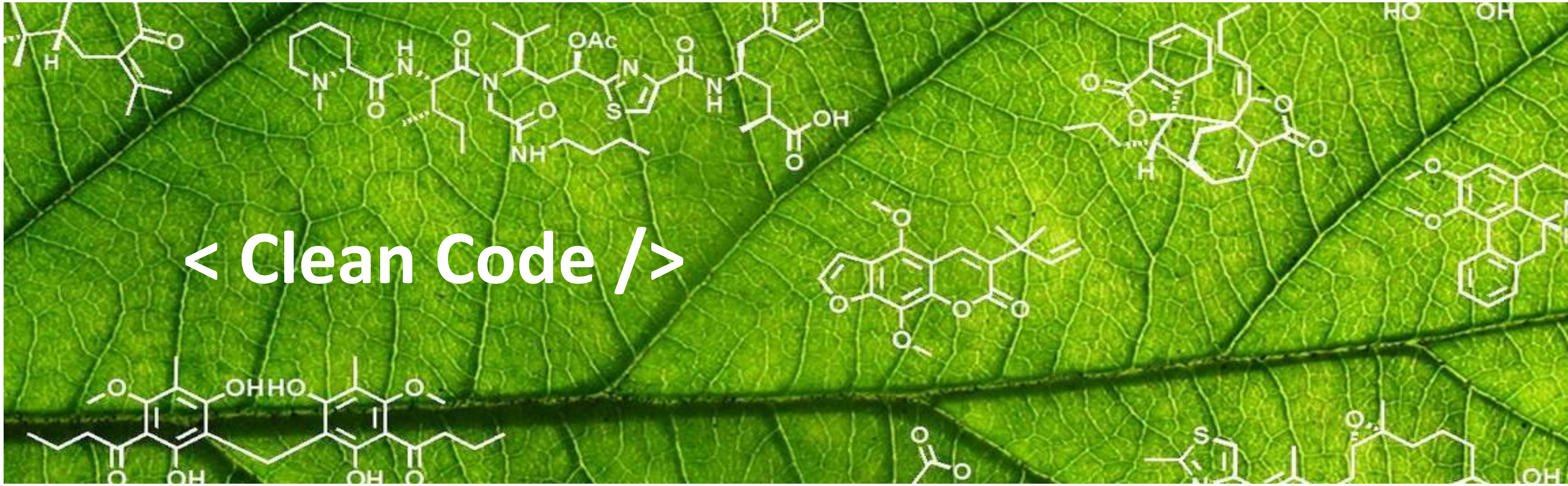
Leibniz-Institut für Pflanzenbiochemie



Sustainable software development in the context of NFDI4Chem

Fabian Mauz

WissenschaftsCampus Pflanzenbasierte Bioökonomie Halle

Short introduction

Brocken [1]

Wernigerode [2]

Stapelburg [3]

Inner german border [4]

- Born in Wenigerode
- 2004 moved to Halle and studied business informatics
- Joined the IPB 2019
  - worked in the AiA Project and implemented CRIMSy
  - since 2023 working in the Project „chemotion“ as a software developer

Agenda

1. Overview of NFDI4Chem
2. Sustainable software development
3. Summary and outlook

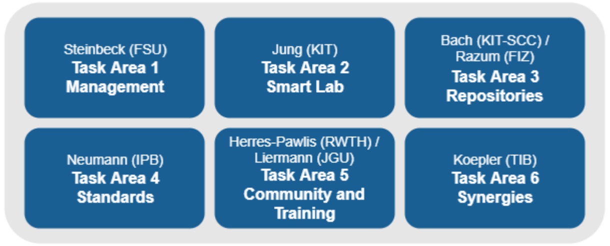
Overview of NFDI

nfdi Nationale Forschungsdaten Infrastruktur

- Systematically and sustainably improving access to research data
- Currently 26 different consortia + Base4NFDI ranging from social sciences to natural sciences

NFDI4Chem

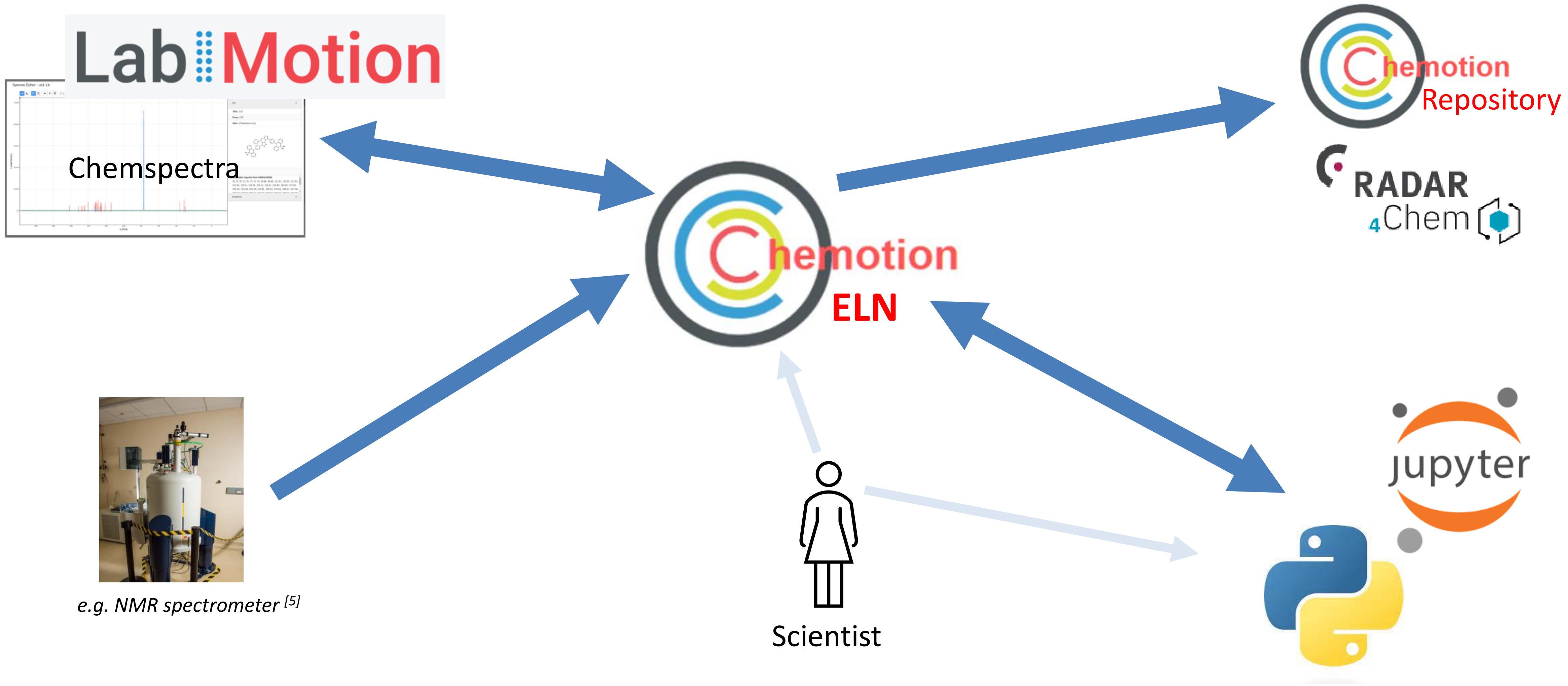
- All chemists publish FAIR data
- Create long living data infrastructure for the German research field of chemistry



Steinbeck (FSU) **Task Area 1 Management**	Jung (KIT) **Task Area 2 Smart Lab**	Bach (KIT-SCC) / Razum (FIZ) **Task Area 3 Repositories**
Neumann (IPB) **Task Area 4 Standards**	Herres-Pawlis (RWTH) / Liermann (JGU) **Task Area 5 Community and Training**	Koepler (TIB) **Task Area 6 Synergies**

- TA2 – Smart Lab : Implementation of IT components -> chemotion ELN

Chemotion ecosystem



Chemotion ELN

- **Open source** electronic lab notebook with a strong focus on **chemicals** and **reactions**
- Initiated by the working group of **Stefan Bräse**
  at the IOC/ IBCS of Karlsruhe Institute of Technology (KIT)
  in **2015**

- Contribute new features and workflows to support the biological and biochemical needs of the IPB
- E.g. Adding cell lines, extracts, ….

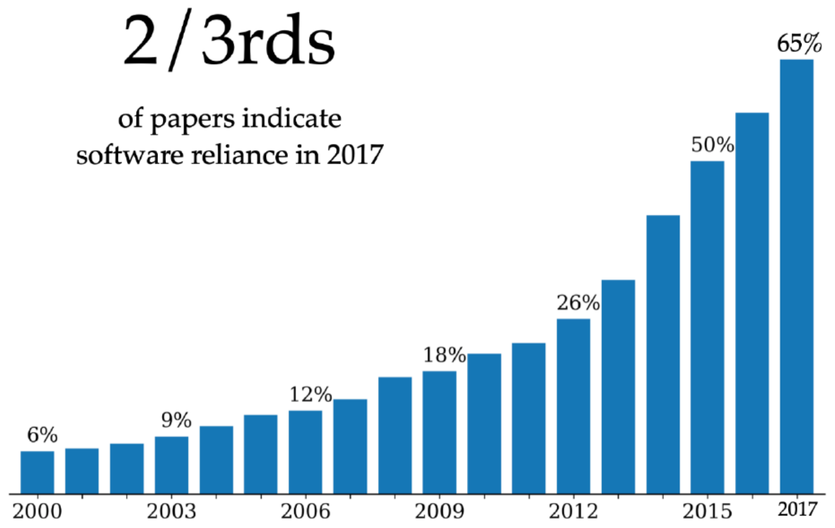
Motivation

Sustainable software

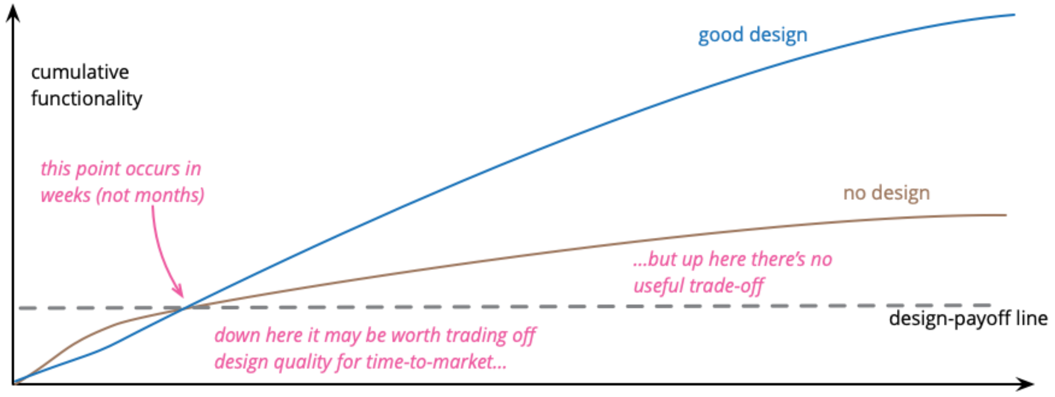
Longliving and maintainable

Easily expandable

Clean coded, tested and documented



https://bit.ly/37XEJ2u



M. Fowler, *Design Stamina Hypothesis*, **2007**

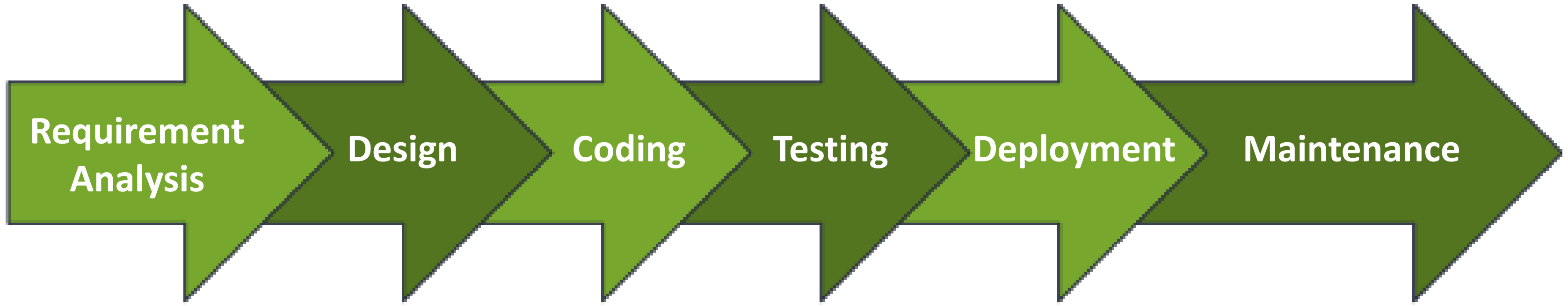
Scientists spend 50%
of the time finding bugs

P. Prabhu, *A Survey of the Practice of Computational Science*, **2011**

"Adding manpower to a late software project makes it later."

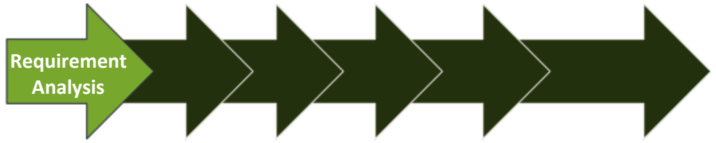
Brooks, *The Mythical Man-Month*, **1975**

Software development process



- **Each** step is just as important as any other
- Most projects focus only on the **“Coding”** step
- Many steps have a major influence on **subsequent** but also **previous** steps

Requirement analysis

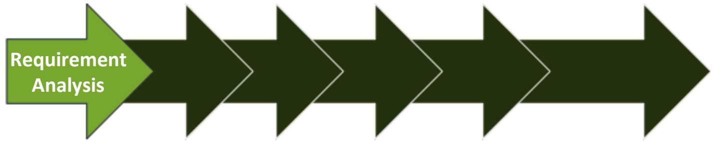


- Identify **each** stakeholder of the software, not only the obvious one

- Find a common domain language for each group of stakeholders

- There are different categories of requirements: domain, user and technical requirements, …

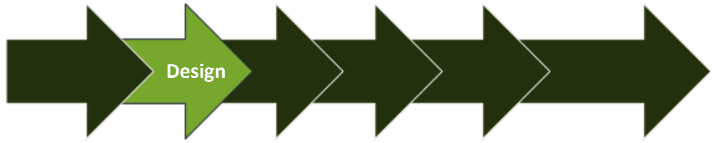
Requirement analysis

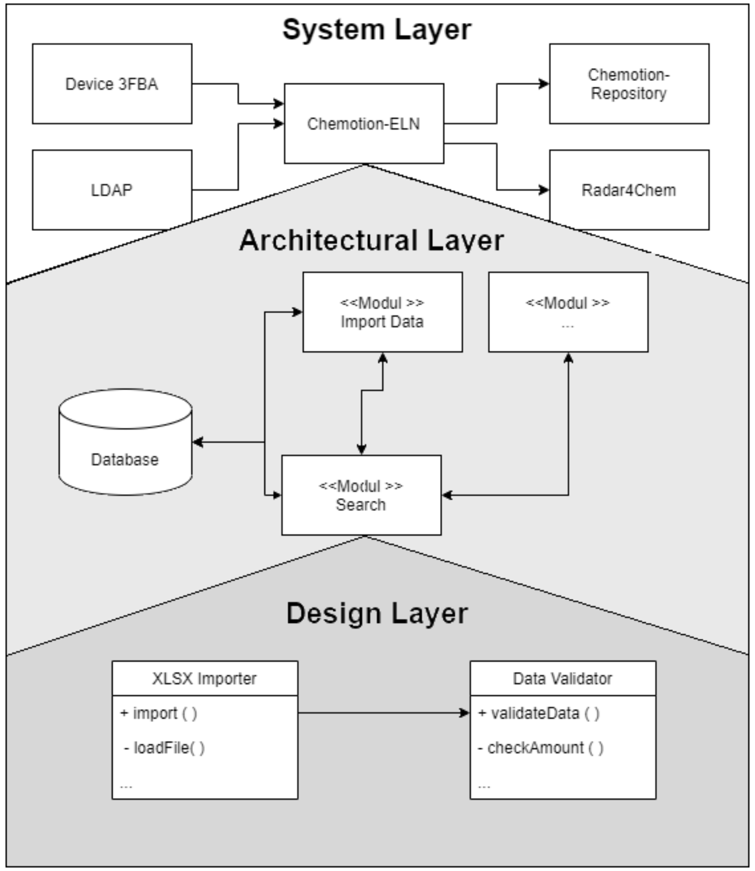


Example: chemotion eln – cell line element

- Identify **each** stakeholder of the software, not only the obvious one
  - **Enduser**: Scientists at KIT, TU Braunschweig and IPB
  - **Maintainer**: Ops team at KIT
  - **System / Software architect, Other Developers, Project Lead**
- Find a common domain language for each group of stakeholders
  - Very **technical** language with developers, architects
  - A **field specific** language for users and projectleads
    - I had to **learn** this language by studying the domain
- There are different categories of requirements: domain, user and technical requirements, …

Software architecture and design

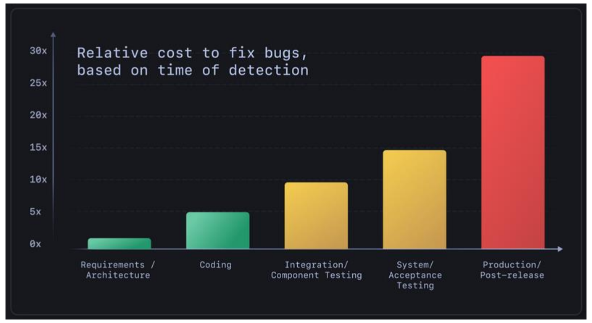




- How does the software **interact** with **other Software ?**
- “Where” is the software **located** ?
- How is the entire software **structured internally ?**
- What are the basic **technologies ?**
- How is a **modul** structured internally ?
- What are the concrete **technologies ?**

Coding and Testing – Costs of a bug

Coding & Testing



G. Tassay, *The Economic Impacts of Inadequate Infrastructure for Software Testing* , **2002**

- **Costs** of finding a bug grows **exponetially**

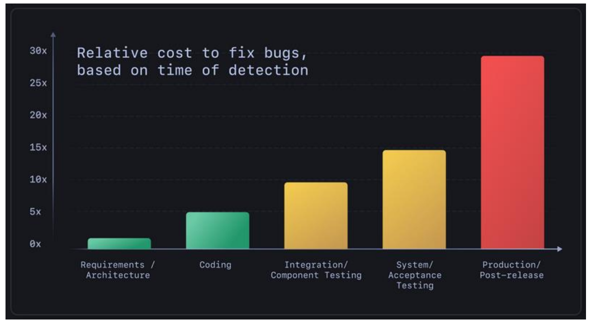
Facebook made big mistake in data it provided to researchers, undermining academic work

C Timberg, **2021,** Washington Post

Coding and Testing – Costs of a bug

Coding & Testing

- **Costs** of finding a bug grows **exponetially**



G. Tassay, *The Economic Impacts of Inadequate Infrastructure for Software Testing ,* **2002**

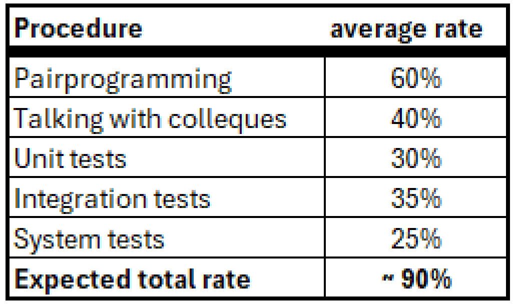
Facebook made big mistake in data it provided to researchers, undermining academic work

C Timberg, **2021,** Washington Post

Ariane 5 crash 1996 [6]

But there is hope

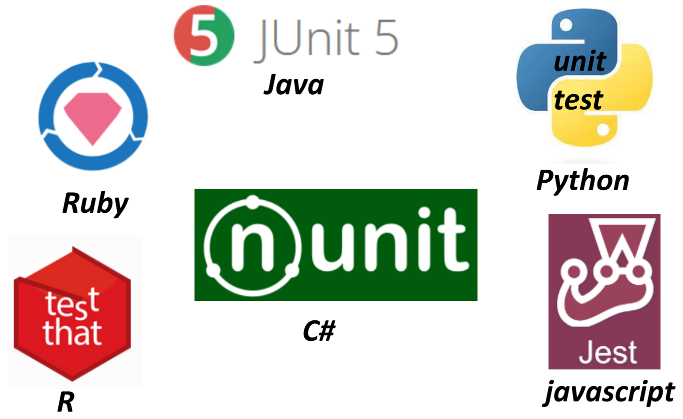
Probability of finding a bug

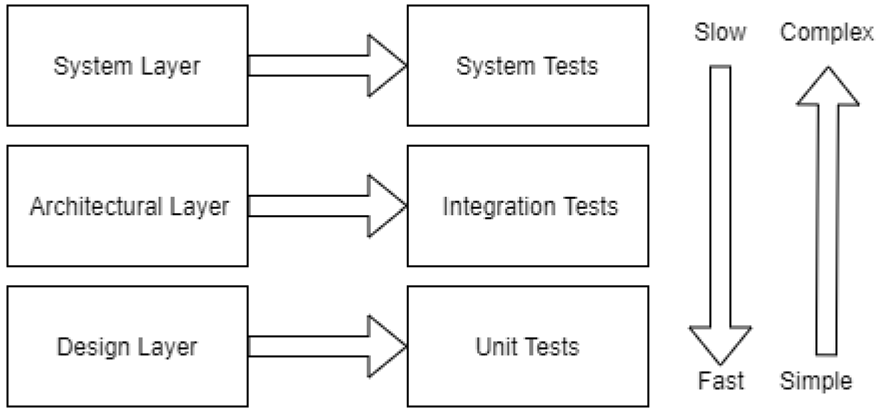


Procedure	**average rate**
Pairprogramming	60%
Talking with colleques	40%
Unit tests	30%
Integration tests	35%
System tests	25%
Expected total rate	**~ 90%**

S. McConnel, Code Complete, **2004,** 485

Coding and Testing – Tests



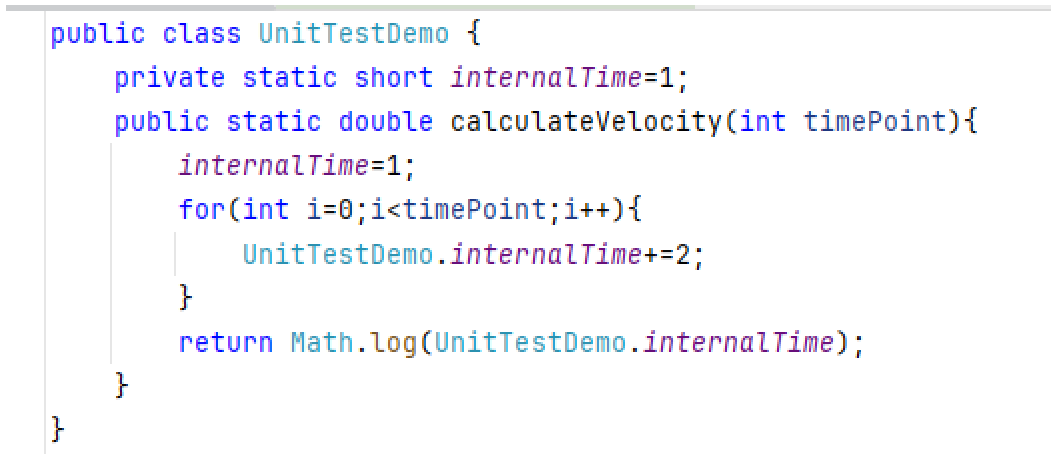


Unit Tests

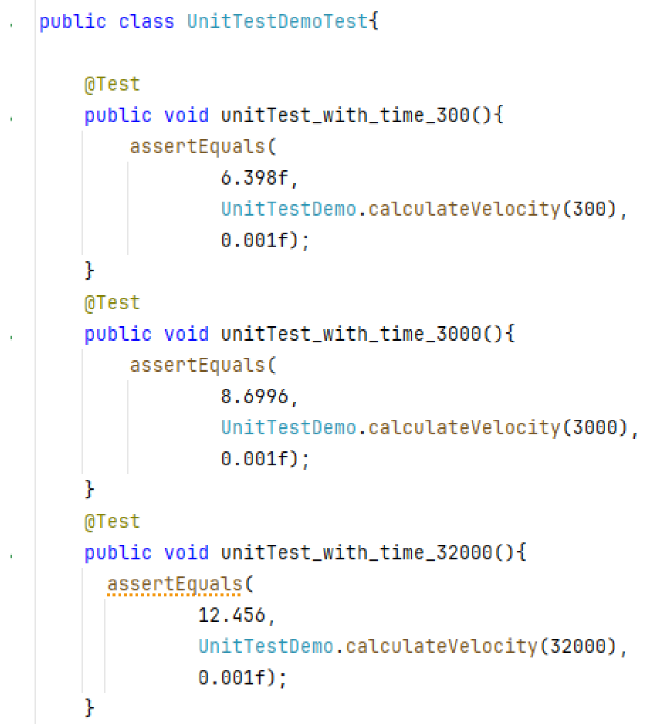
- Run **automatically** and **isolated**
- Tests the component under clearly **defined conditions**
- Written by the **developer**

Coding and Testing – Unit test example

Coding area

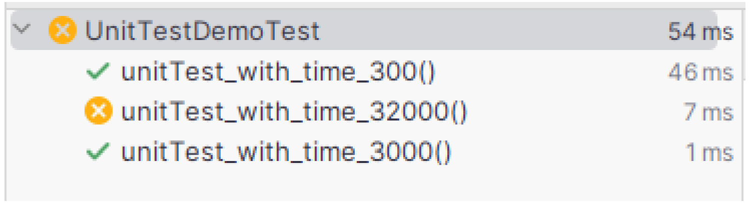


```
public class UnitTestDemo {
    private static short internalTime=1;
    public static double calculateVelocity(int timePoint){
        internalTime=1;
        for(int i=0;i<timePoint;i++){
            UnitTestDemo.internalTime+=2;
        }
        return Math.log(UnitTestDemo.internalTime);
    }
}
```



```
public class UnitTestDemoTest{

    @Test
    public void unitTest_with_time_300(){
        assertEquals(
                6.398f,
                UnitTestDemo.calculateVelocity(300),
                0.001f);
    }
    @Test
    public void unitTest_with_time_3000(){
        assertEquals(
                8.6996,
                UnitTestDemo.calculateVelocity(3000),
                0.001f);
    }
    @Test
    public void unitTest_with_time_32000(){
        assertEquals(
                12.456,
                UnitTestDemo.calculateVelocity(32000),
                0.001f);
    }
```



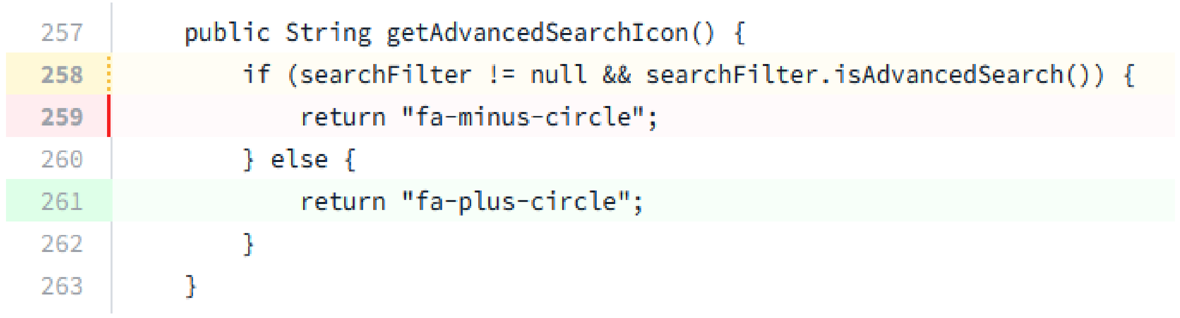
```
UnitTestDemoTest                    54 ms
   unitTest_with_time_300()         46 ms
   unitTest_with_time_32000()        7 ms
   unitTest_with_time_3000()         1 ms
```

- Checks well **defined situations**
- Can only prove **absence** of bugs not correctness
- **Asyncronous** code is hard to test

Coding and Testing – Code Coverage

Coding area

- Gives you information how much of your **production code** is **tested**
- **Idea**: the test programm rembers which line was executed in a test
- The result is a value between **0% - 100%**



```
    public String getAdvancedSearchIcon() {
        if (searchFilter != null && searchFilter.isAdvancedSearch()) {
            return "fa-minus-circle";
        } else {
            return "fa-plus-circle";
        }
    }
```

Software	Coverage
CRIMSy	70.10%
Chemotion_ELN	
- Backend	63.80%
- Frontend	25.96%
Rocket.chat	54.00%
Flutter	92.00%

!! But it can lead to false security !!

Coding and Testing – Clean code & S.O.L.I.D. principles

Clean Code

- Collection of guidelines and principles
- Good code is intiuative code

„Selfdocumeting code“

„Refactoring“

„Principle of least suprise“

„Don‘t repeat yourself“

„YAGNI“

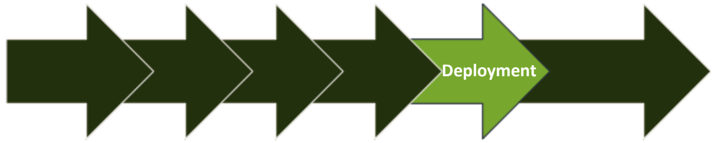
„KISS principle“

Robert C. Martin [7]

Single-responsibility principle
Open close principle
Liskov substitution principle
Interface segregation principle
Dependency inversion principle

R. Martin, Clean Code: A Handbook of Agile Software Craftsmanship, **2008**

Maintenance

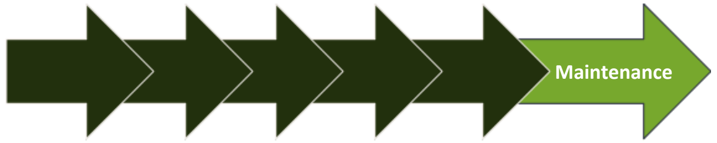


- A **version control system** is an absolute must for software development
- Not only source code versioning but a lot of **tools**

  -> **Github Actions**

- Allows **continous integration** & deployment
- Public IPB instance : https://github.com/ipb-halle

Deployment – Continuous Integration



Where to put the software code ?

https://github.com/ipb-halle

How can i make the code / software visible ?

https://wiki.ipb-halle.de

How is responsible for the software / data ?

That depends

Outlook: Artificial Intelligence in software development

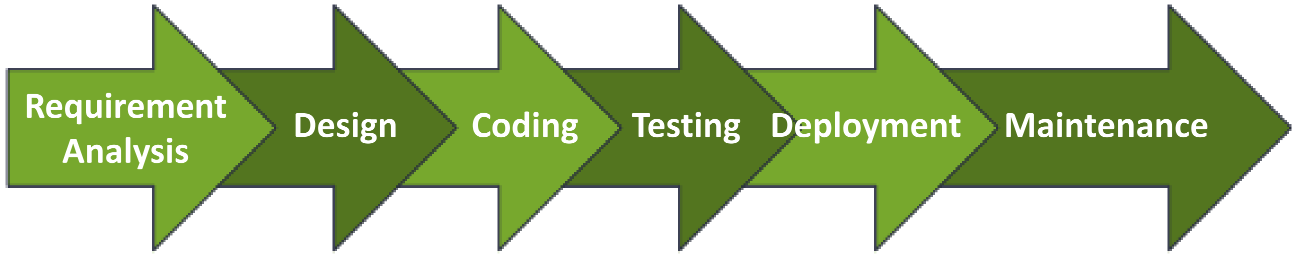
… it is vital that your children learn computer science. Everybody should learn how to program. And in fact, it‘s almost exactly the opposite.

Jensen Huang, CEO of Nvidia, *World Government Summit in Dubai*, **2024**

Outlook: Artificial Intelligence in software development

… it is vital that your children learn computer science. Everybody should learn how to program. And in fact, it‘s almost exactly the opposite.

Jensen Huang, CEO of Nvidia, *World Government Summit in Dubai*, **2024**



Contributed features to ELN

Features contributed to ELN by IPB

Functional

Embedded image editor

new core element : **Cell lines**

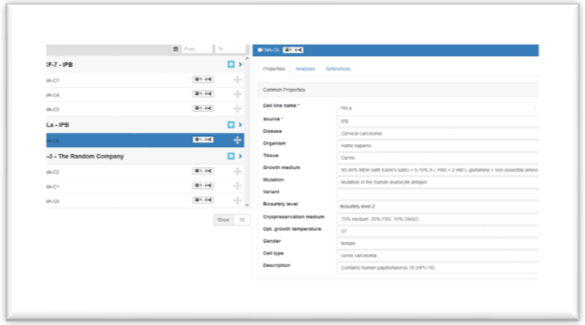
Flexibilisation of the well plates

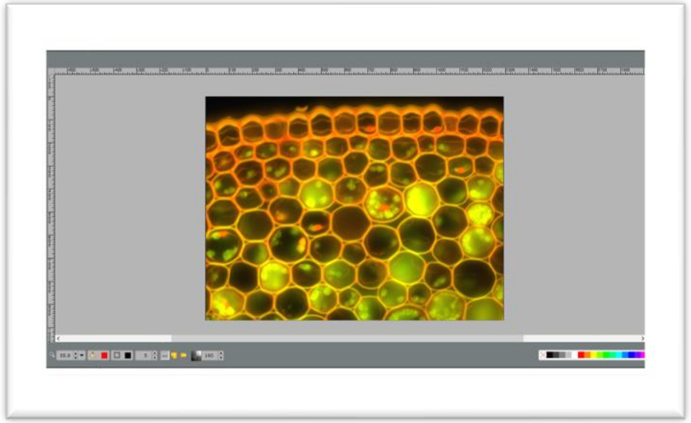
Non-functional

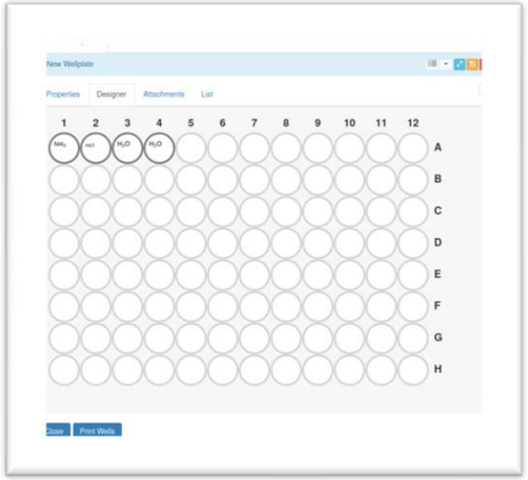
Refactoring of file handling system

Introduction of clean code principles

Creation and improving of a CI pipeline







Thank you

Suresoft
SUSTAINABLE RESEARCH SOFTWARE

https://suresoft.dev/

I would like to thank

Prof. Ludger Wessjohann

Dr. Frank Broda

The chemotion team from KIT

NWC department & Dr. Steffen Neumanns group

Chemotion developer team

DFG Deutsche Forschungsgemeinschaft

Image references

[1] - https://commons.wikimedia.org/wiki/File:Brocken_und_Hohneklippen_im_Harz_%28Sachsen-Anhalt%29_Nationalpark.jpg, Creative Commons Attribution-Share Alike 4.0 International

[2] - https://commons.wikimedia.org/wiki/File:Schloss_Wernigerode_2018.jpg, Creative Commons Attribution-Share Alike 4.0 International

[3] - https://commons.wikimedia.org/wiki/File:Stapelburg_001.jpg, Creative Commons Attribution-Share Alike 4.0 International

[4] - https://www.google.de/maps/place/38871+Stapelburg/@51.8983073,10.6461708,6001m/data=!32!1e3!4b1!4m6!3m5!1s0x47a56cdd8d07d2af:0x4236659f8074bc0!8m2!3d51.901267!4d10.6626119!16s%2Fm%2F03b_r56?entry=ttu

[5] - https://www.flickr.com/photos/pnnl/46765164934, CC BY-NC-SA 2.0 Deed

[6] - https://www.youtube.com/watch?v=PK_yguLapgA

[7] - https://itkonekt.com/2019/12/19/robert-c-martin-uncle-bob-2/