Development Workflow

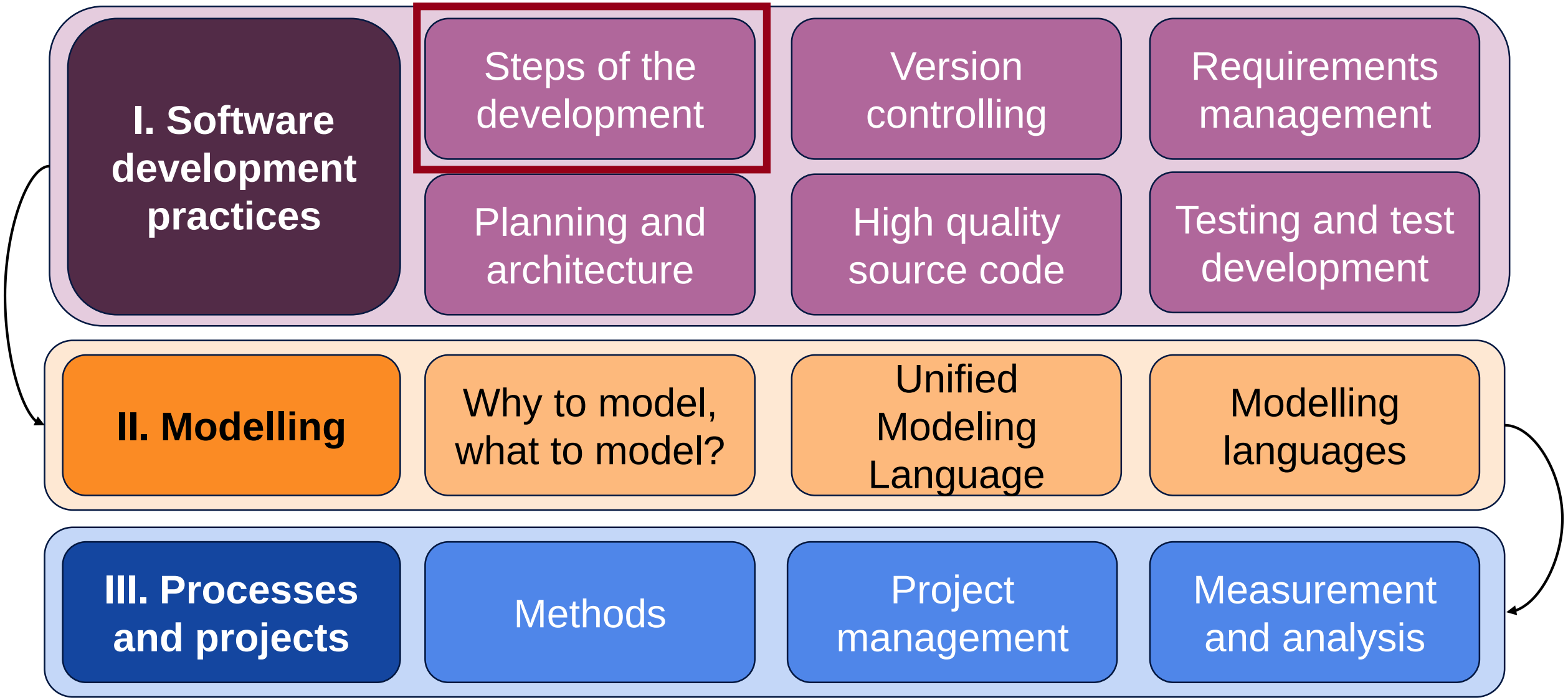
HUSZERL Gábor
huszerl@mit.bme.hu

MŰEGYETEM 1782
BME VILLAMOSMÉRNÖKI ÉS INFORMATIKAI KAR
Méréstechnika és Információs Rendszerek Tanszék
ftsrg Critical Systems Research Group

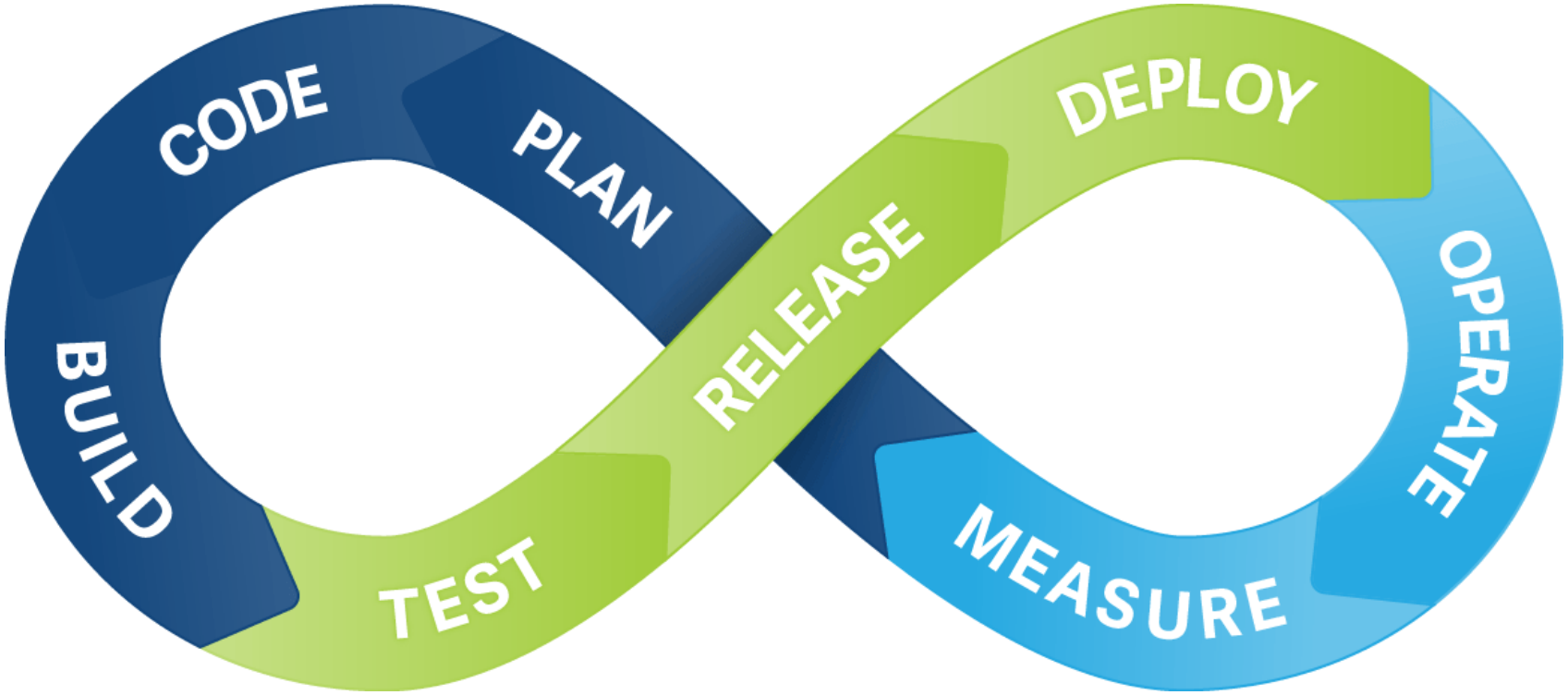
Learning Outcomes

- At the end of the lecture the students are expected to be able to
- (K2) summarize the typical steps of software development,
- (K1) list the good practices that support the individual steps,
- (K3) use build tools to create software products.

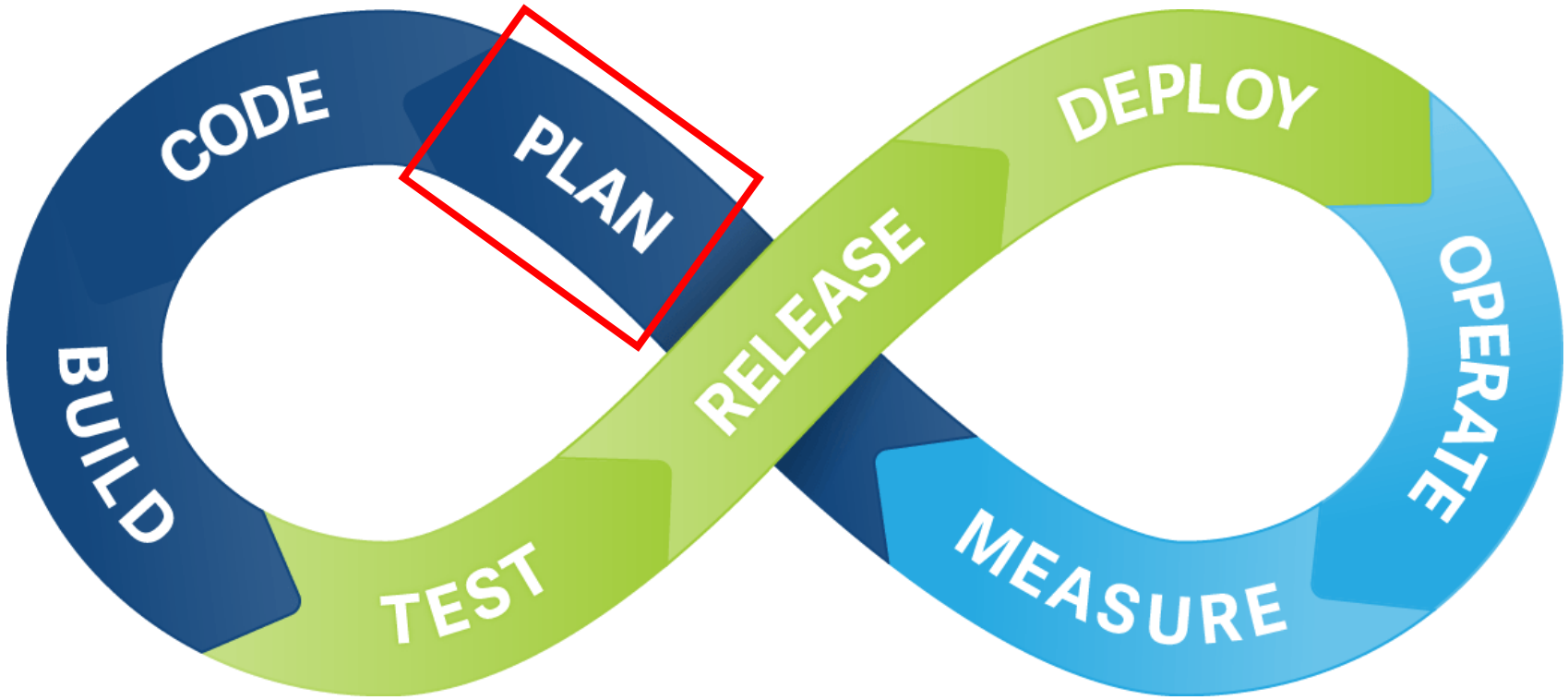
Further Topics of the Subject



DevOps = Development + Operations



The Tasks of Software Development



What Are the Tasks of a Junior Developer?

New feature

Repair error

Refactor

...

Developing New Features (1): Input

- The goal of the feature, and its detailed description
- (The process does not start here, see the *Requirements* lectures)

node #46727

Indicate when a test is started in test_runner #46727

Closed connor4312 opened on Feb 19

connor4312 on Feb 19

What is the problem this feature will solve?

When running tests in an editor (like VS Code), users want to know what test is currently being executed. This is useful, so if a test is hanging for instance, it's easy to see what it is.

What is the feature you are proposing to solve the problem?

Currently, TAP output for a test like...

```
describe("math", () => {
  it("addition",async () => {
    strictEqual(1 + 1, 2);
  });
});
```

looks like this:

```
# Subtest: math
    # Subtest: addition
    ok 1 - addition
      ---
      duration_ms: 1.369826
      ...
```

Assignees: No Assignees

Labels: good first issue, feature request, test_runner

Milestone: No Milestone

Status: Pending Triage

Repository: node

Linked pull requests: #47797, #48428

Open in new tab

Copy link

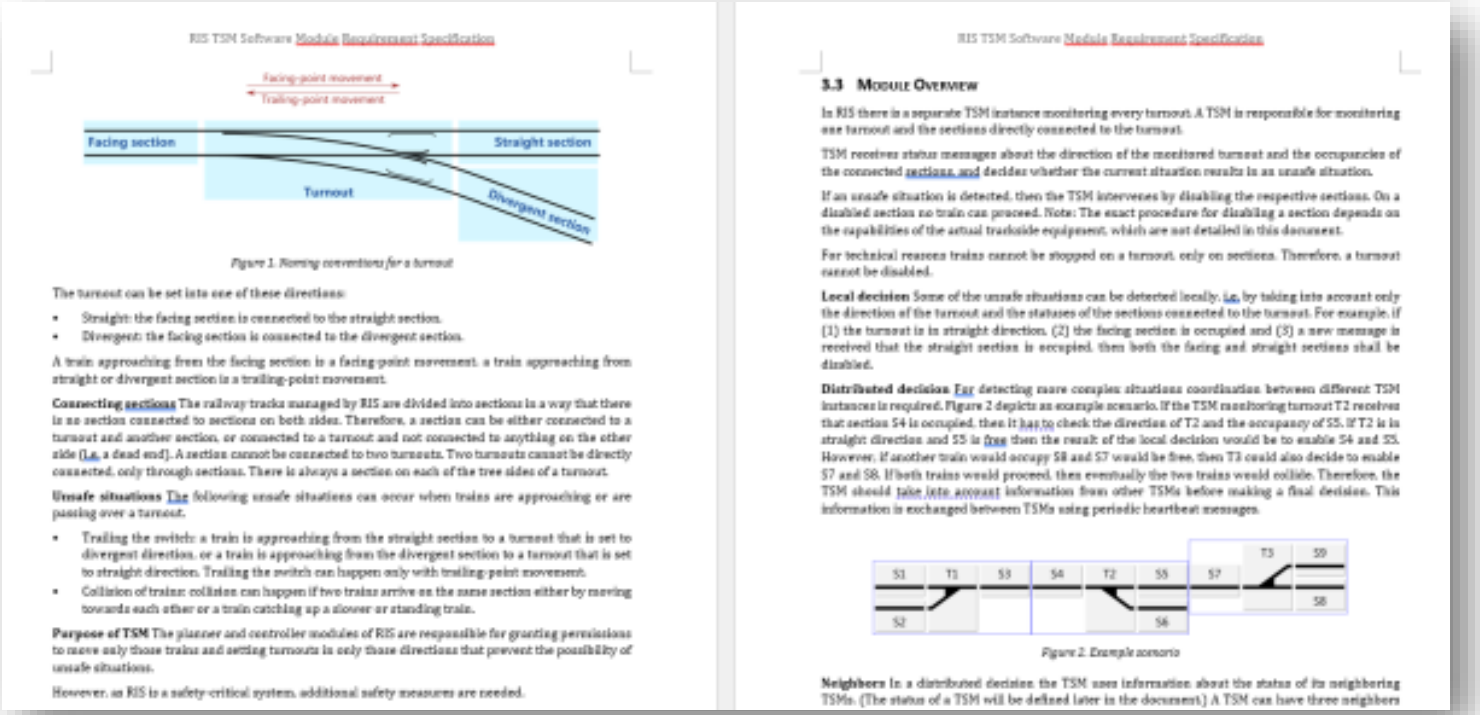
Copy link in project

In an open source / agile environment:

issue / story / ticket

Developing New Features (1): Input

- The goal of the feature, and its detailed description
- (The process does not start here, see the *Requirements* lectures)



RIS TSM Software Module Requirement Specification

Figure 1. Naming conventions for a turnout

The turnout can be set into one of these directions:

- Straight: the facing section is connected to the straight section.
- Divergent: the facing section is connected to the divergent section.

A train approaching from the facing section is a facing-point movement, a train approaching from straight or divergent section is a trailing-point movement.

Connecting sections The railway tracks managed by RIS are divided into sections in a way that there is no section connected to sections on both sides. Therefore, a section can be either connected to a turnout and another section, or connected to a turnout and not connected to anything on the other side (i.e. a dead end). A section cannot be connected to two turnouts. Two turnouts cannot be directly connected, only through sections. There is always a section on each of the two sides of a turnout.

Unsafe situations The following unsafe situations can occur when trains are approaching or are passing over a turnout.

- Trailing the switch: a train is approaching from the straight section to a turnout that is set to divergent direction, or a train is approaching from the divergent section to a turnout that is set to straight direction. Trailing the switch can happen only with trailing-point movement.
- Collision of trains: collision can happen if two trains arrive on the same section either by moving towards each other or a train catching up a slower or standing train.

Purpose of TSM The planner and controller modules of RIS are responsible for granting permissions to move only those trains and setting turnouts in only those directions that prevent the possibility of unsafe situations.

However, as RIS is a safety-critical system, additional safety measures are needed.

RIS TSM Software Module Requirement Specification

3.3 Module Overview

In RIS there is a separate TSM instance monitoring every turnout. A TSM is responsible for monitoring one turnout and the sections directly connected to the turnout.

TSM receives status messages about the direction of the monitored turnout and the occupancies of the connected sections, and decides whether the current situation results in an unsafe situation.

If an unsafe situation is detected, then the TSM intervenes by disabling the respective sections. On a disabled section no train can proceed. Note: The exact procedure for disabling a section depends on the capabilities of the actual trackside equipment, which are not detailed in this document.

For technical reasons trains cannot be stopped on a turnout, only on sections. Therefore, a turnout cannot be disabled.

Local decision Some of the unsafe situations can be detected locally, i.e. by taking into account only the direction of the turnout and the statuses of the sections connected to the turnout. For example, if (1) the turnout is in straight direction, (2) the facing section is occupied and (3) a new message is received that the straight section is occupied, then both the facing and straight sections shall be disabled.

Distributed decision For detecting more complex situations coordination between different TSM instances is required. Figure 2 depicts an example scenario. If the TSM monitoring turnout T2 receives that section S4 is occupied, then it has to check the direction of T2 and the occupancy of S5. If T2 is in straight direction and S5 is free then the result of the local decision would be to enable S4 and S5. However, if another train would occupy S8 and S7 would be free, then T3 could also decide to enable S7 and S8. If both trains would proceed, then eventually the two trains would collide. Therefore, the TSM should take into account information from other TSMs before making a final decision. This information is exchanged between TSMs using periodic heartbeat messages.

Figure 2. Example scenario

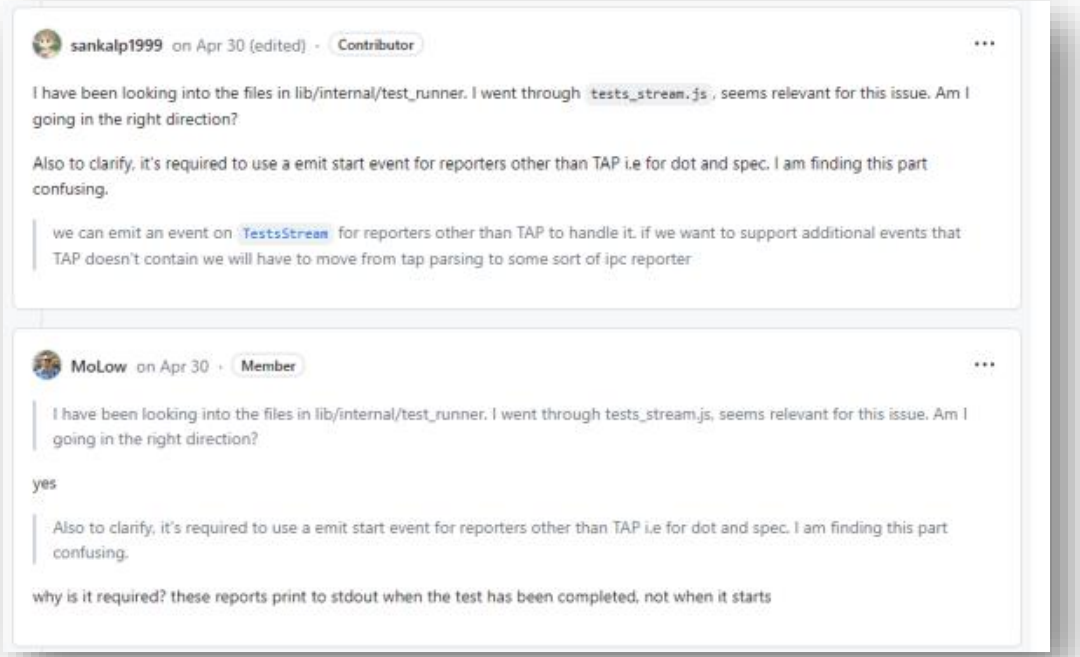
Neighbors In a distributed decision the TSM uses information about the status of its neighboring TSMs. (The status of a TSM will be defined later in the document.) A TSM can have three neighbors

In a (safety) critical environment:

modul specification

Developing New Features (2): Review

- Before you start to develop anything: **review**
- Realizable/practicable? (In)consistent? (In)complete? Verifiable?



sankalp1999 on Apr 30 (edited) · Contributor

I have been looking into the files in lib/internal/test_runner. I went through `tests_stream.js`, seems relevant for this issue. Am I going in the right direction?

Also to clarify, it's required to use a emit start event for reporters other than TAP i.e for dot and spec. I am finding this part confusing.

> we can emit an event on `TestsStream` for reporters other than TAP to handle it. if we want to support additional events that TAP doesn't contain we will have to move from tap parsing to some sort of ipc reporter

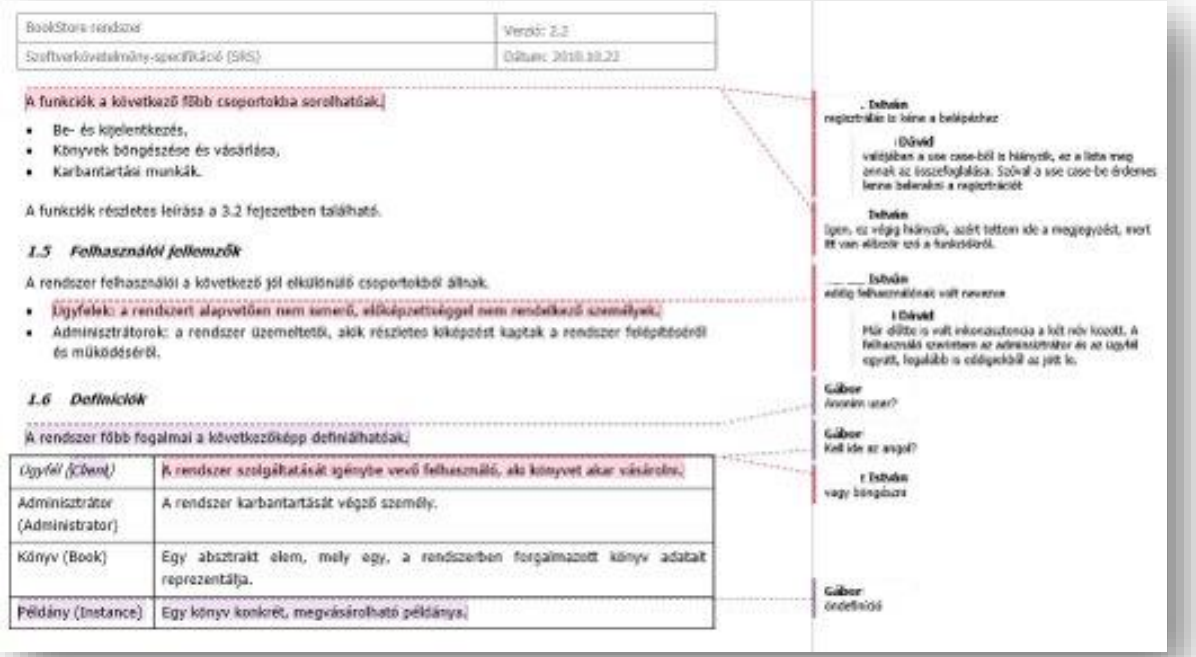
MoLow on Apr 30 · Member

> I have been looking into the files in lib/internal/test_runner. I went through tests_stream.js, seems relevant for this issue. Am I going in the right direction?

yes

> Also to clarify, it's required to use a emit start event for reporters other than TAP i.e for dot and spec. I am finding this part confusing.

why is it required? these reports print to stdout when the test has been completed, not when it starts



BookStore rendszer	Verzió: 2.2
Szoftverkövetelmény-specifikáció (SRS)	Dátum: 2010.10.22

A funkciók a következő főbb csoportokba sorolhatóak:

- Be- és kijelentkezés,
- Könyvek böngészése és vásárlása,
- Karbantartási munkák.

A funkciók részletes leírása a 3.2 fejezetben található.

1.5 Felhasználói jellemzők

A rendszer felhasználói a következő jól elkülönülő csoportokból állnak.

- Ügyfelek: a rendszert alapvetően nem ismerő, előképzettséggel nem rendelkező személyek.
- Adminisztrátorok: a rendszer üzemeltetői, akik részletes kiképzést kaptak a rendszer felépítéséről és működéséről.

1.6 Definíciók

A rendszer főbb fogalmai a következőképp definiálhatóak:

Ügyfél (Client)	A rendszer szolgáltatását igénybe vevő felhasználó, aki könyvet akar vásárolni.
Adminisztrátor (Administrator)	A rendszer karbantartását végző személy.
Könyv (Book)	Egy absztrakt elem, mely egy, a rendszerben forgalmazott könyv adatait reprezentálja.
Példány (Instance)	Egy könyv konkrét, megvásárolható példánya.

István: regisztrálni is kéne a belépéshez

Dávid: valójában a use case-ből is hiányzik, ez a lista meg ennek az összefoglalása. Szóval a use case-be érdemes lenne belenézni a regisztrációt

István: Igen, ez végig hiányzik, azért tettem ide a megjegyzést, mert itt van eltérés szó a funkcióról.

István: eddig felhasználónak volt nevezve

Dávid: Már dőlte is volt elkerülhetetlen a két név kapott. A felhasználó szintben az adminisztrátor és az ügyfél egyaránt, legalább is célügyeiről az jut le.

Gábor: Anonim user?

Gábor: Kell ide az angol?

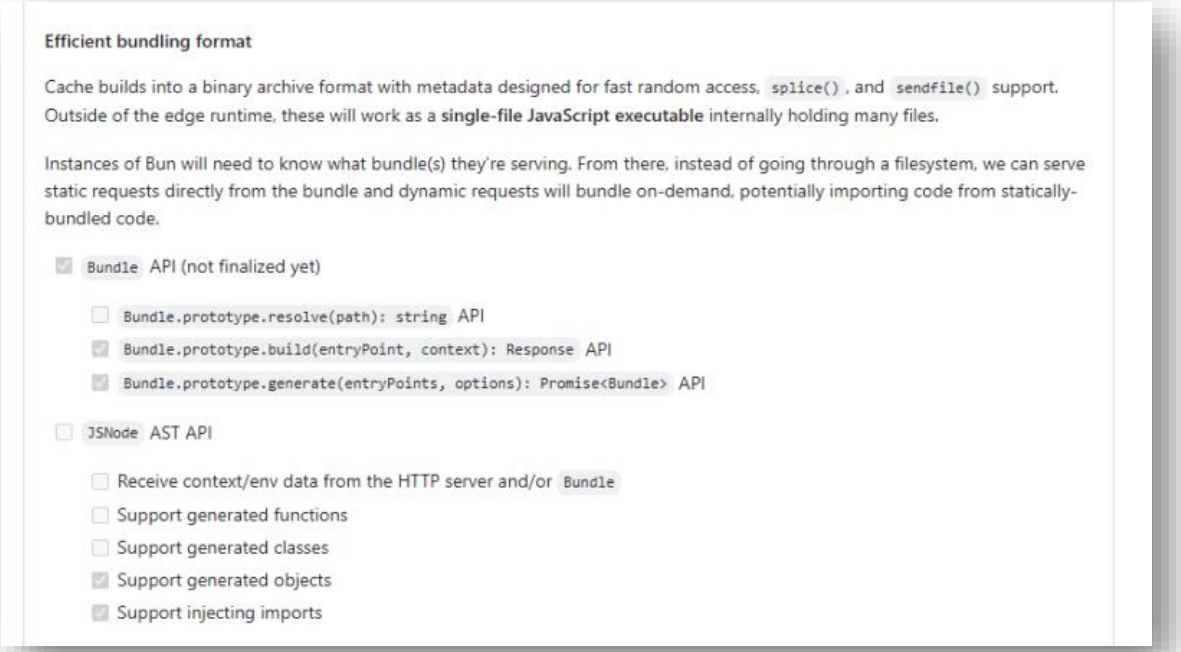
István: vagy böngészni

Gábor: definíció

See the **Requirements Management** lecture

Developing New Features (3): Detailed Plans

- **Detailed plans** based on the so far available information (iteratively?)
- Diverse levels of elaboration: steps / pseudo code / (UML) models

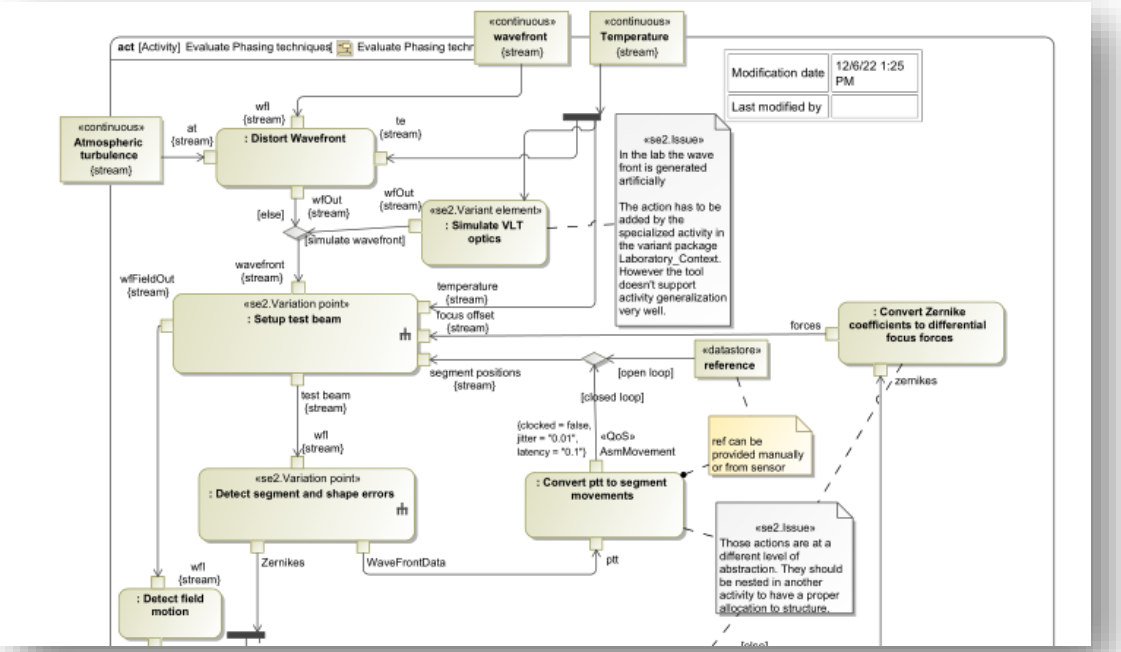


Efficient bundling format

Cache builds into a binary archive format with metadata designed for fast random access, `splice()`, and `sendfile()` support. Outside of the edge runtime, these will work as a **single-file JavaScript executable** internally holding many files.

Instances of Bun will need to know what bundle(s) they're serving. From there, instead of going through a filesystem, we can serve static requests directly from the bundle and dynamic requests will bundle on-demand, potentially importing code from statically-bundled code.

- [x] `Bundle` API (not finalized yet)
  - [ ] `Bundle.prototype.resolve(path): string` API
  - [x] `Bundle.prototype.build(entryPoint, context): Response` API
  - [x] `Bundle.prototype.generate(entryPoints, options): Promise<Bundle>` API
- [ ] `JSNode` AST API
  - [ ] Receive context/env data from the HTTP server and/or `Bundle`
  - [ ] Support generated functions
  - [ ] Support generated classes
  - [x] Support generated objects
  - [x] Support injecting imports



See the **Design and Architecture** lecture, part **UML Modelling**

Repairing Errors

- Similar process, but here you repair already existing software
- Source of the bug reports: user / developer / automatic test / …

Open | 3 tasks done | XCUITestDriver returns StaleElementReferenceError: Element does not exist in cache #18839

ilendemli opened this issue on Jul 4 · 23 comments

Is the component officially supported by the Appium team?

- [x] I have verified the component repository is present under the Appium organization in GitHub

Is there an existing issue for this?

- [x] I have searched the existing issues

Current Behavior

For some reason, after quickly switching back and forth between the NATIVE_APP and WEBVIEW context (doing getContexts before switching) in our test cases, locating elements fail because of StaleElementReferenceError. `getPageSource` fails with `Method threw 'org.openqa.selenium.WebDriverException' exception.`

```
[debug] [ios-device] Received message from Web Inspector:
[debug] [ios-device] {
[debug] [ios-device]   "__argument": {
[debug] [ios-device]     "WIRApplicationIdentifierKey": "PID:33769",
[debug] [ios-device]     "WIRMessageDataTypeKey": "WIRMessageDataTypeFull",
[debug] [ios-device]     "WIRDestinationKey": "d5038e8c-c470-4150-989e-ccb31941c4fd",
[debug] [ios-device]     "WIRMessageDataKey": "{\"method\":\"Target.dispatchMessageFromTarget\",\"params\":{\"targetId\":
[debug] [ios-device]   },
[debug] [ios-device]   "__selector": "_rpc_applicationSentData:"
[debug] [ios-device] }
```

The webpage is shown in a hybrid app. It is inspectable through Safari.

Expected Behavior

Getting Element after switching context to WEBVIEW should not fail.

Minimal Reproducible Example

I have no example to provide at the moment but will try to create one to reproduce the issue.

Environment

- Operating system: MacOS & iOS
- Appium server version (output of `appium --version`): 2023.5.2
- Appium driver(s) and their version(s): WDA 5.5.1
- Appium plugin(s) and their version(s):

Bug report

- In the best case a lot of information
- Exact failure description
- Environment (OS, language, version, …)
- Steps to reproduce
- Screen shots, sample data

Reproduction

Bug hunting

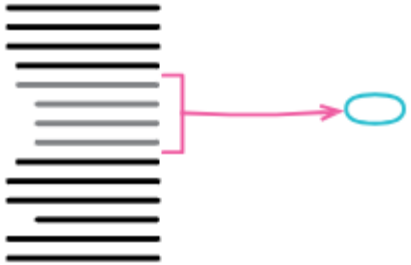
Repairing

Refactoring

> **refactoring:** a disciplined technique for restructuring an existing body of code, altering its internal structure without changing its external behavior.

Martin Fowler, https://refactoring.com/

Example: Extract Method

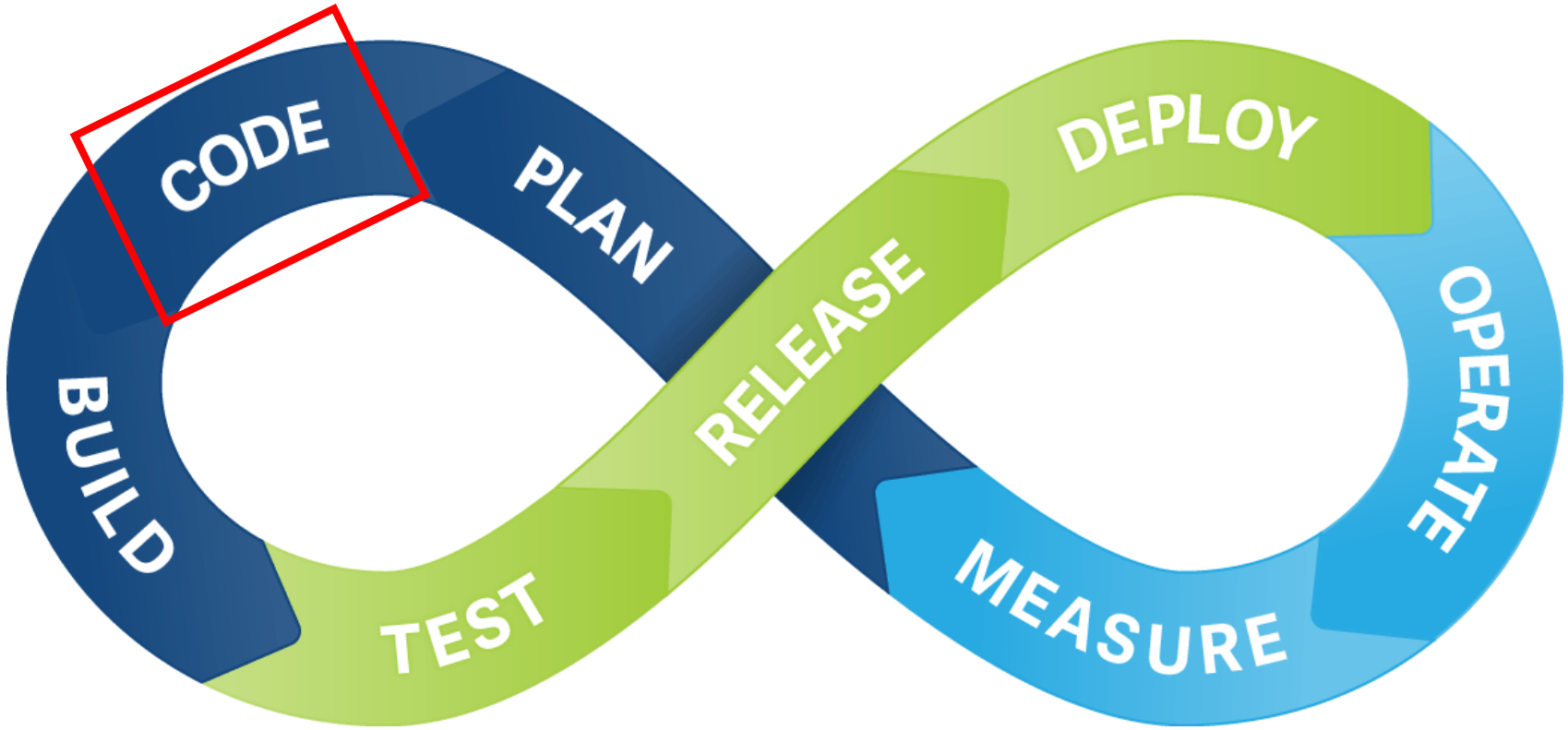


Typical usage

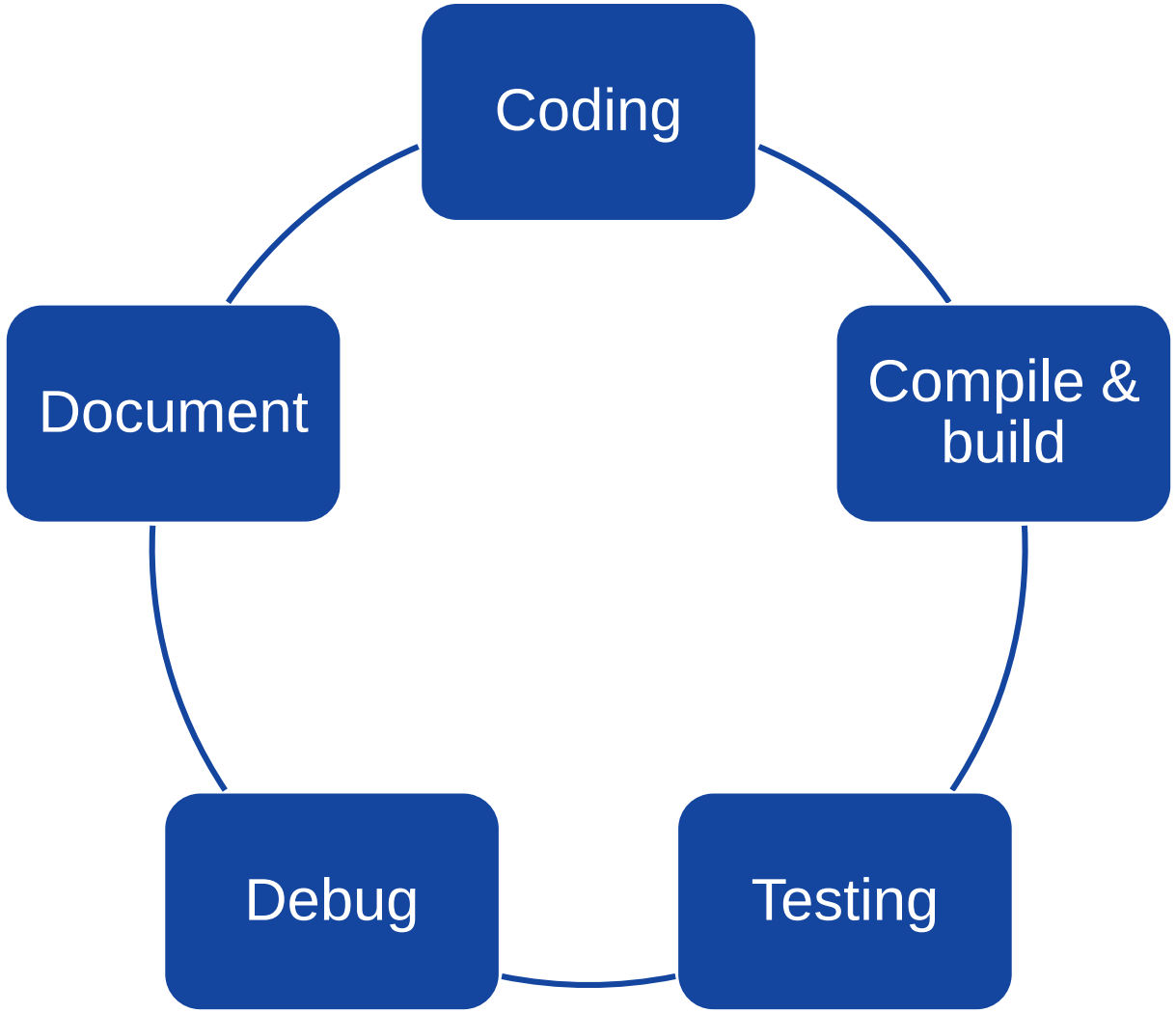
- Improve: maintainability, readability, performance
- Extensibility (for later)
- Continuously during coding

For the steps of refactoring: https://refactoring.com/catalog/

The Tasks of Software Development



Get the Code Ready (in your local environment)



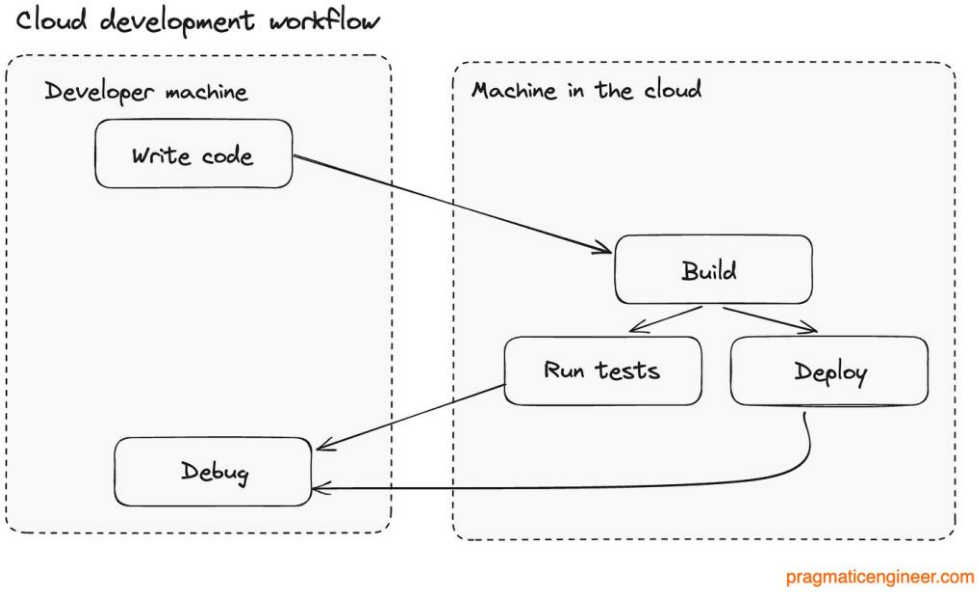
Where to Create the Code?

Local development environment

- Laptop / PC …
- Installing further tools (to IDE): compiler, webserver, package manager, …)
- Common development configuration (version control it!)
- Manual / automatic
- Developer productivity (!)

Remote (cloud) development environment

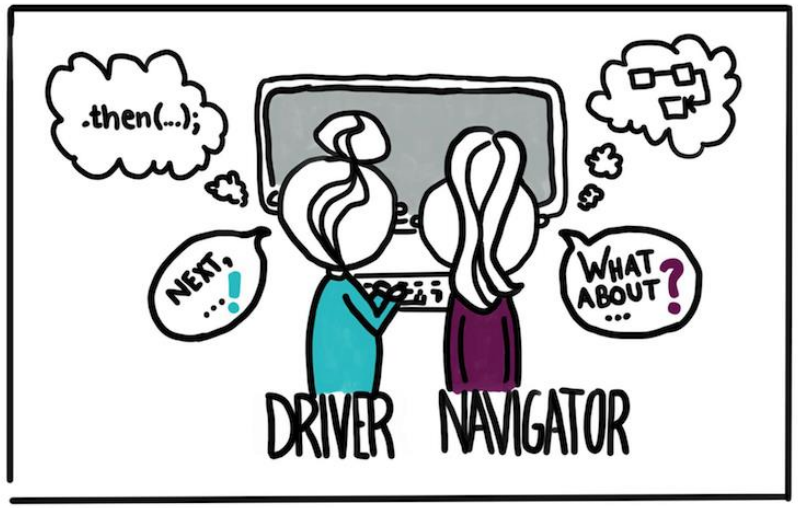
- (Partially) runs on remote machines
- More resource, common environment



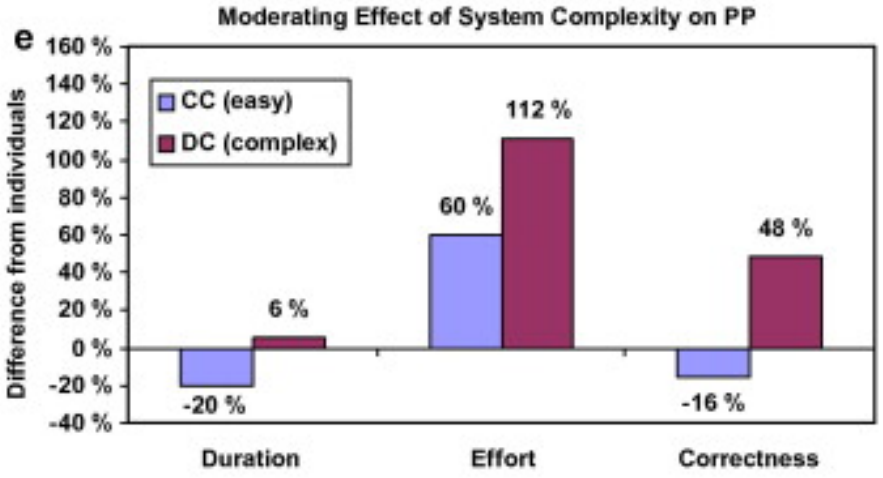
Codespaces Gitpod

How: Pair Programming

- **Pair programming**: two in front of one computer
  - One types and implements, the other observes and helps (change!)
- Knowledge sharing, mentoring, tactical/strategic thinking
- Can result is better quality, but it must be learnt either!



source



Source: „The effectiveness of pair programming: A meta-analysis", DOI: 10.1016/j.infsof.2009.02.001

How: Coding Guidelines & Rules

- Industry, language or company specific sets
- **Do** & **don't** (possibly consider)
- Typical fields:
  - Formatting (parentheses, spaces, …)
  - Conventions (structure, naming, …)
  - Patterns (suggested and forbidden elements, …)
- Tool support (!)
  - IDE, linter…

Google Java Style Guide

Table of Contents

Source: https://google.github.io/styleguide/javaguide.html

See the **High quality source code** lecture

How: Static Analysis Tools

- Static analysis: examining the code without running it
- Identifying generic faults, problems (e.g. division by zero)
- Feedback during writing the code

"sizeof" should not be called on pointers

Analyze your code

Bug Major Quick Fix cwe cert

Why is this an issue?

`sizeof` returns the size in bytes of a type. One common usage pattern, especially in C, is to use `sizeof` to determine the size of an array. However, arrays decay to pointers when passed as arguments to a function, and if `sizeof` is applied to such an argument, it will return the size of the pointer, not of the array. A similar issue happens when the array is used in an arithmetic operation.

This rule raises issues when:

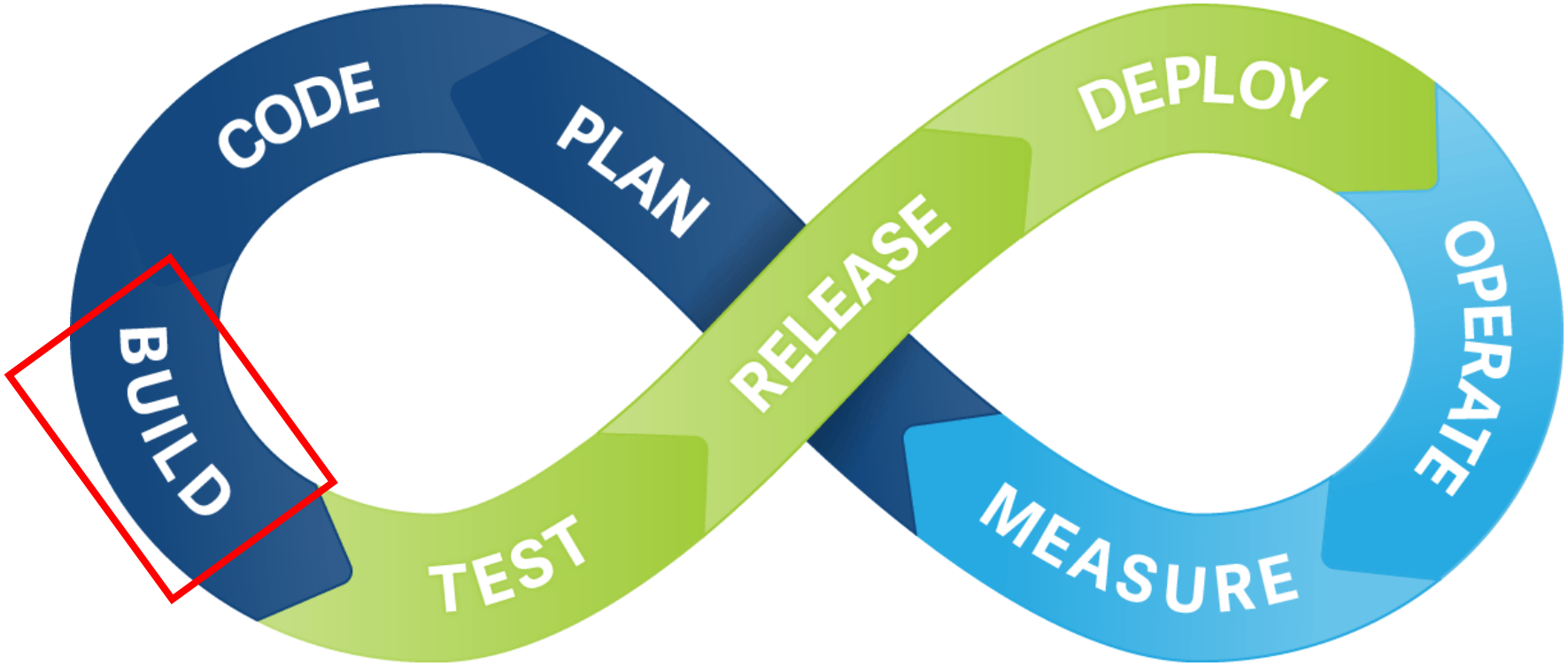
- `sizeof` is used to compute the array size of a pointer passed as a function argument.
- `sizeof` is called on the result of an arithmetic operation involving an array.

Note: C++17 provides a `std::size` function that will correctly compute the number of elements of an array and fail to compile if provided with a pointer. It is simpler and safer to use this variant when available. C++20 also provides the functions `std::ssize`, `std::ranges::size`, and `std::ranges::ssize` with similar effects.

Noncompliant code example

```
void fun(int *data, int array[10]) {
  size_t const dataSize = sizeof data / sizeof(int); // Noncompliant, type of data is int *
  size_t const arraySize = sizeof array / sizeof(int); // Noncompliant, type of array is int * too
  int primes[] = { 1, 2, 3, 5, 7, 13, 17, 19};
  size_t const primesSize = sizeof primes / sizeof(int); // Compliant, type of primes is int[8]
  size_t const primesSize2 = sizeof(primes + 1) / sizeof(int); // Noncompliant, type of primes + 1
}
```

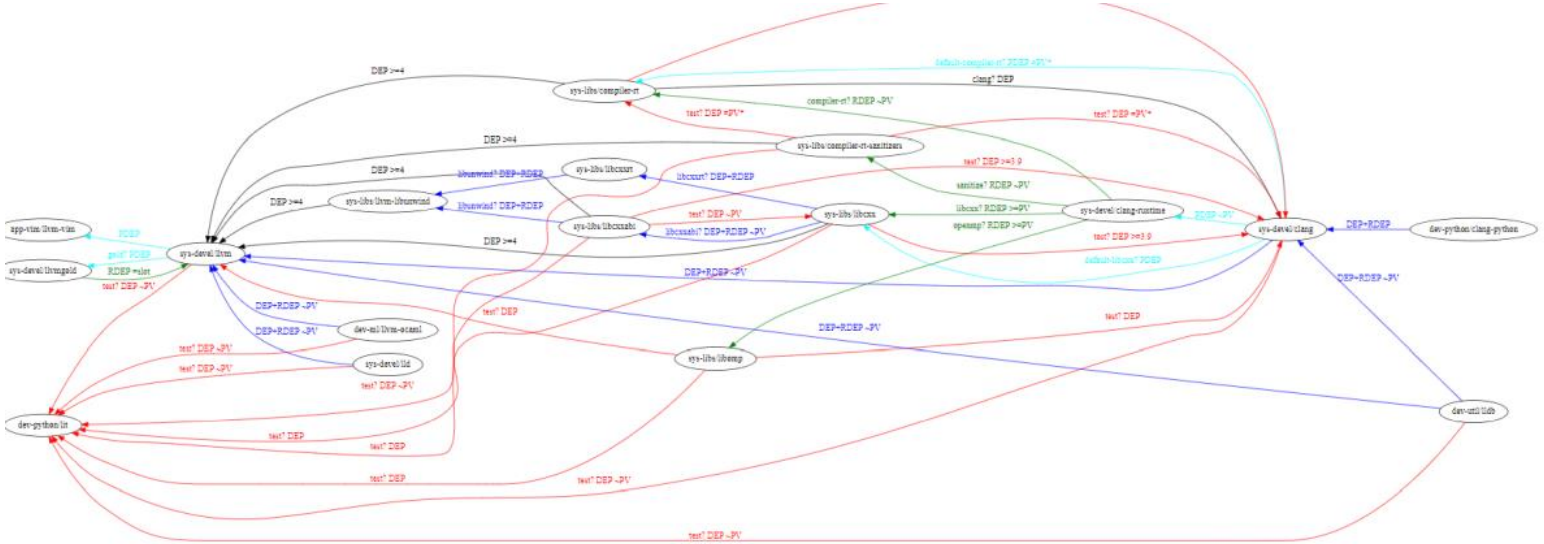
The Tasks of Software Development



Is the Compiler Not Enough?

```
> gcc hello.c -o hello
```

.github	workflows: Split new-prs into two workflows
bolt	[BOLT][RISCV] Implement R_RISCV_PCREL_LO12_S (#65204)
clang-tools-extra	[clang-tidy] Exclude bitwise operators in bugprone-non-zero-enum-to-b...
clang	Fix some typos in comments: evalute -> evaluate (NFC) (#65906)
cmake	[CMake] Switch the CMP0091 policy (MSVC_RUNTIME_LIBRARY) to the n...
compiler-rt	[NFC][sanitizer] Return nullptr from PlatformDemangle
cross-project-tests	Fix Dexter test after 37a20cc. Adding the XFAIL line caused the line ...
flang	Fix some typos in comments: evalute -> evaluate (NFC) (#65906)
libc	[libc] Implement stdio writing functions for the GPU port (#65809)
libclc	libclc: Fix signed integer underflow in abs_diff
libcxx	[libc++] Use inline instead of static in headers.
libcxxabi	Revert "[Demangle] demangle builtin type transformations"
libunwind	[libunwind] Haiku: Initial support
lld	[ELF] Priorize the last catch-all pattern in version scripts
lldb	[lldb] Reinstate default constructor for SBCommandInterpreter
llvm-libgcc	[llvm-libgcc] Don't export __{,de}register_frame_info* and __register...
llvm	[M68k] Swap operands of atomic_store after D123143
mlir	Fix some typos in comments: evalute -> evaluate (NFC) (#65906)
openmp	Fix some typos in comments: evalute -> evaluate (NFC) (#65906)
polly	Bug fix for multi-line labels in CFG dot graph
pstl	Clear release notes for 18.x
runtimes	Reland "[CMake] Bumps minimum version to 3.20.0.
third-party	[lldb] Fix duplicate word typos: NFC
utils	[bazel] Add new TableGen output files for 46f3ade
.arcconfig	Add modern arc config for default "onto" branch
.arclint	PR46997: don't run clang-format on clang's testcases.



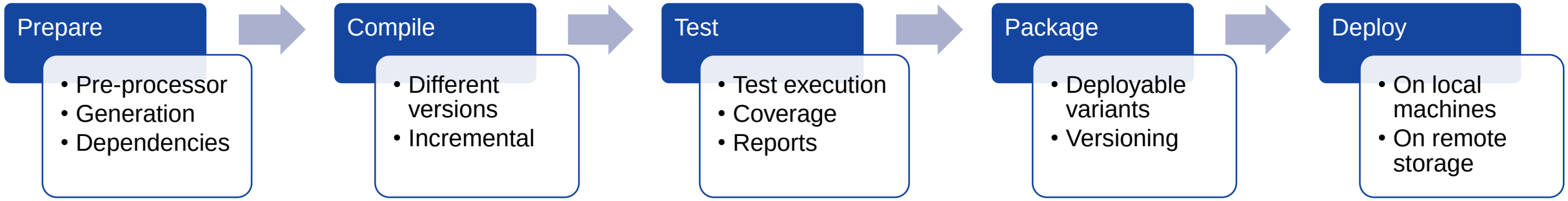
Compilation and build challenges

- Many source and other files (several hundreds …)
- Several variants (architecture, debug/release)
- External libraries (and their dependencies)

Build

- Build: Creating the software product from its source (code)
  - Compilation is only one step of it!

- Build frameworks: configurable steps towards specific goals
  - Maven, Gradle, CMake, MSBuild, Bazel…



Example: Build Tools (Make)

```
CFLAGS ?= -g

all: helloworld

helloworld: helloworld.o
    # Commands start with TAB not spaces
    $(CC) $(LDFLAGS) -o $@ $^

helloworld.o: helloworld.c
    $(CC) $(CFLAGS) -c -o $@ $<

clean: FRC
    $(RM) helloworld helloworld.o
```

Parameters and variables

Goals and dependencies among them

There are always many repeating steps to be defined

Example: Build Tools (Maven)

```
1. <project xmlns="." xmlns:xsi=". " xsi:schemaLocation=".">
2.   <groupId>com.mycompany.app</groupId>
3.   <artifactId>my-app</artifactId>
4.   <version>1.0-SNAPSHOT</version>
5.
6.   <properties>
7.     <maven.compiler.source>1.7</maven.compiler.source>
8.     <maven.compiler.target>1.7</maven.compiler.target>
9.   </properties>
10.
11.  <dependencies>
12.    <dependency>
13.      <groupId>junit</groupId>
14.      <artifactId>junit</artifactId>
15.      <version>4.12</version>
16.      <scope>test</scope>
17.    </dependency>
18.  </dependencies>
19.</project>
```

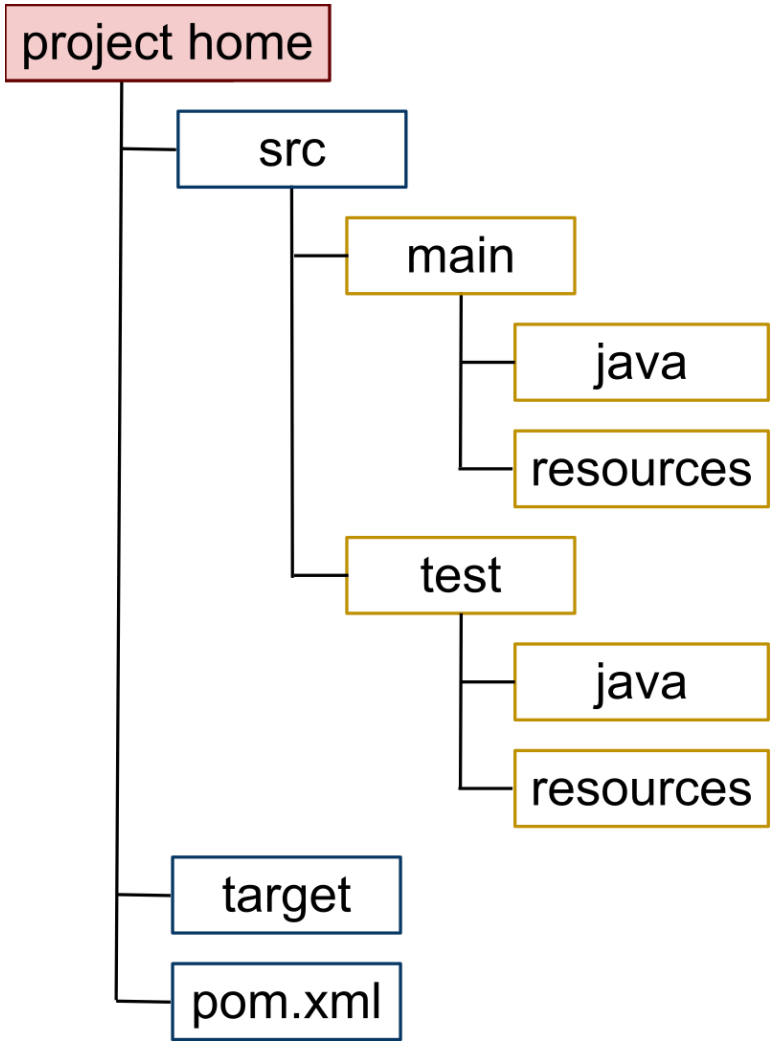
Name and version of the software

Parameters and properties

Managing dependencies

Principle: „convention over configuration”

Example: Standard File Structure



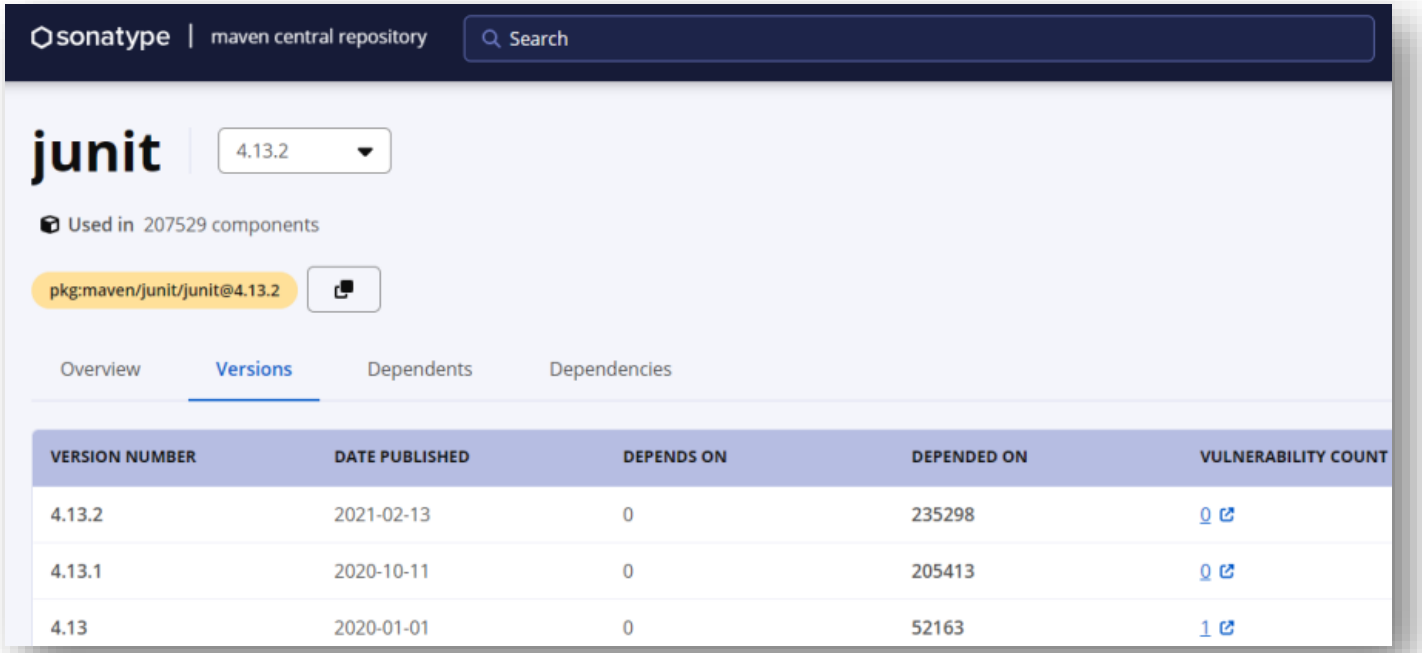
- `src`
  - `main`: code of the application
    - `java`: for each language separately
    - `resources`: anything except source code (images, audio, data, …)
  - `test`: test code in similar structure
- `target`:
  - build results come here, will not be version controlled
- `pom.xml`:
  - build configuration (Maven) must be version controlled!

Managing Dependencies

- **Typical dependencies**
  - Another part of the current project (common definitions, auxiliary libs, …)
  - Anything that is not part of a standard libraries (graphics, net, …)
- **Challenges**
  - Define, on what exactly our project depends
  - Obtain the right version
  - Make it available to the local compiler
- **Where to store the dependencies?**
  - In the version control system (as source code or as binary?)
  - In own central storage or on the Internet

Centralized Dependency Management

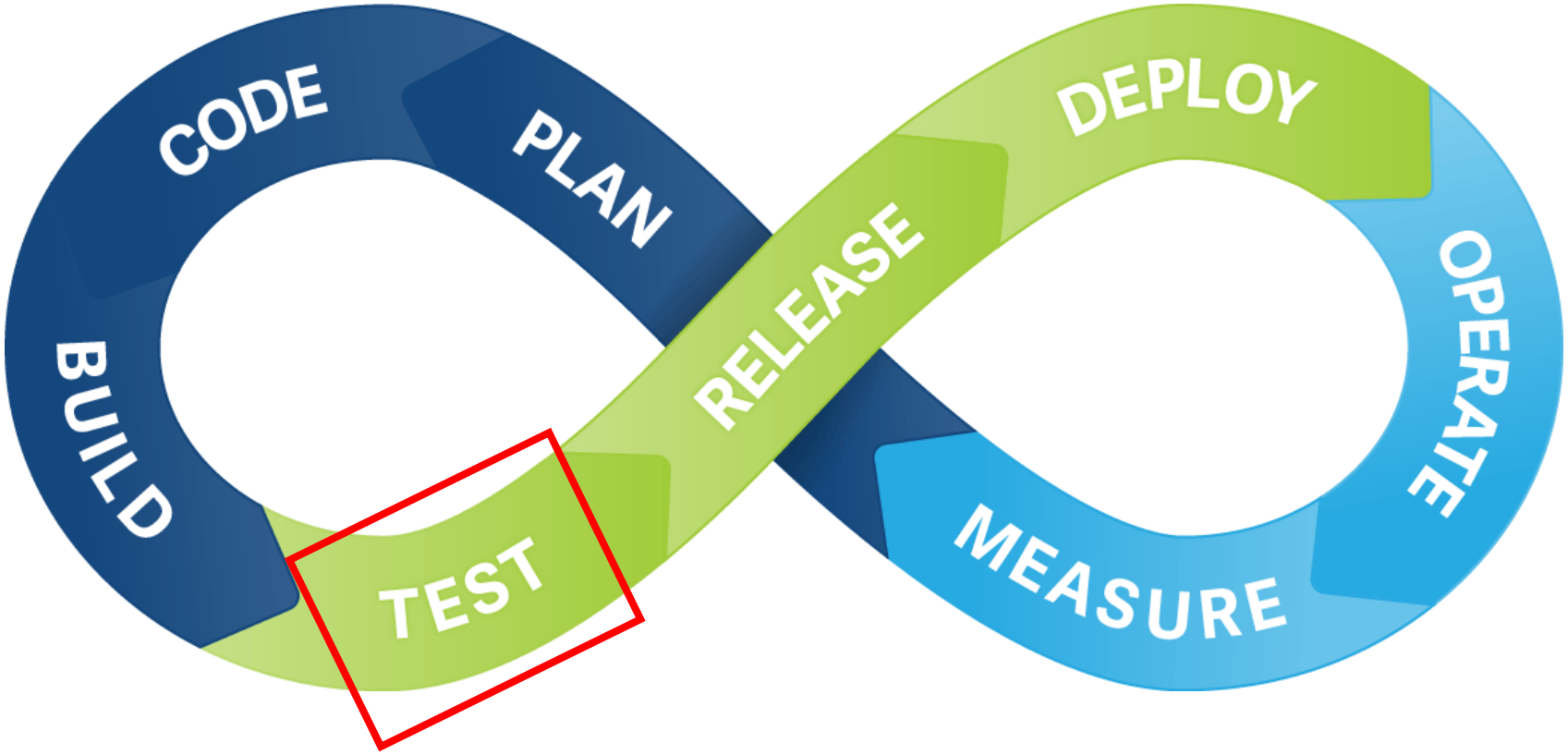
- **Unique** identification of dependencies (group + product + version)
- Publicly (e.g. maven central) or own server
- Managing transitive dependencies



VERSION NUMBER	DATE PUBLISHED	DEPENDS ON	DEPENDED ON	VULNERABILITY COUNT
4.13.2	2021-02-13	0	235298	0
4.13.1	2020-10-11	0	205413	0
4.13	2020-01-01	0	52163	1

```
[INFO] -------------< hu.bme.mit.spaceship:hu.bme.mit.spaceship >--------------
[INFO] Building hu.bme.mit.spaceship 0.5.0-SNAPSHOT
[INFO] --------------------------------[ jar ]---------------------------------
Downloading from central: https://repo.maven.apache.org/maven2/net/bytebuddy/byte-buddy/1.12.8/byte-buddy-1.12
Downloaded from central: https://repo.maven.apache.org/maven2/net/bytebuddy/byte-buddy/1.12.8/byte-buddy-1.12.
Downloading from central: https://repo.maven.apache.org/maven2/net/bytebuddy/byte-buddy-parent/1.12.8/byte-buc
Downloaded from central: https://repo.maven.apache.org/maven2/net/bytebuddy/byte-buddy-parent/1.12.8/byte-budc
Downloading from central: https://repo.maven.apache.org/maven2/net/bytebuddy/byte-buddy-agent/1.12.8/byte-budc
Downloaded from central: https://repo.maven.apache.org/maven2/net/bytebuddy/byte-buddy-agent/1.12.8/byte-buddy
Downloading from central: https://repo.maven.apache.org/maven2/junit/junit/3.8.1/junit-3.8.1.pom
Downloaded from central: https://repo.maven.apache.org/maven2/junit/junit/3.8.1/junit-3.8.1.pom (998 B at 14 k
Downloading from central: https://repo.maven.apache.org/maven2/classworlds/classworlds/1.1-alpha-2/classworlds
Downloaded from central: https://repo.maven.apache.org/maven2/classworlds/classworlds/1.1-alpha-2/classworlds-
Downloading from central: https://repo.maven.apache.org/maven2/net/bytebuddy/byte-buddy/1.12.8/byte-buddy-1.12
Downloading from central: https://repo.maven.apache.org/maven2/junit/junit/3.8.1/junit-3.8.1.jar
Downloading from central: https://repo.maven.apache.org/maven2/net/bytebuddy/byte-buddy-agent/1.12.8/byte-budc
Downloading from central: https://repo.maven.apache.org/maven2/classworlds/classworlds/1.1-alpha-2/classworlds
Downloaded from central: https://repo.maven.apache.org/maven2/junit/junit/3.8.1/junit-3.8.1.jar (121 kB at 2.0
Downloaded from central: https://repo.maven.apache.org/maven2/classworlds/classworlds/1.1-alpha-2/classworlds-
Downloaded from central: https://repo.maven.apache.org/maven2/net/bytebuddy/byte-buddy-agent/1.12.8/byte-buddy
Downloaded from central: https://repo.maven.apache.org/maven2/net/bytebuddy/byte-buddy/1.12.8/byte-buddy-1.12.
[INFO]
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ hu.bme.mit.spaceship ---
Downloading from central: https://repo.maven.apache.org/maven2/commons-cli/commons-cli/1.0/commons-cli-1.0.por
Downloaded from central: https://repo.maven.apache.org/maven2/commons-cli/commons-cli/1.0/commons-cli-1.0.pom
Downloading from central: https://repo.maven.apache.org/maven2/classworlds/classworlds/1.1/classworlds-1.1.por
Downloaded from central: https://repo.maven.apache.org/maven2/classworlds/classworlds/1.1/classworlds-1.1.pom
Downloading from central: https://repo.maven.apache.org/maven2/commons-cli/commons-cli/1.0/commons-cli-1.0.jar
Downloading from central: https://repo.maven.apache.org/maven2/classworlds/classworlds/1.1/classworlds-1.1.jar
Downloaded from central: https://repo.maven.apache.org/maven2/classworlds/classworlds/1.1/classworlds-1.1.jar
Downloaded from central: https://repo.maven.apache.org/maven2/commons-cli/commons-cli/1.0/commons-cli-1.0.jar
```

The Tasks of Software Development



Testing

- Testing: running, observe and evaluate the software (more precise definition coming soon!)

- On different levels and for different purposes
  - Level: module/unit, integration, system, …
  - Purposes: function, performance, security, …
  - Automatically or manually (test vs. check)

- Test development techniques (specification, structure, …)

See the **Testing** lecture

Unit Testing

- Managing independent components (module, class, …)
- During the development of the feature: *happy path* and *exceptions*
- Measuring source code coverage

TESTING

Filter (e.g. text, !exclude, @tag)

1/2 tests passed (50.0%)

- hu.bme.mit.spaceship 59ms
  - { } hu.bme.mit.spaceship 59ms
    - GT4500Test 59ms
      - fireTorpedo_Single_Success() 28ms
      - fireTorpedo_All_Succes...

GT4500Test.java 1

src > test > java > hu > bme > mit > spaceship > GT4500Test.java > GT4500Test > fireTorpedo_All_Success()

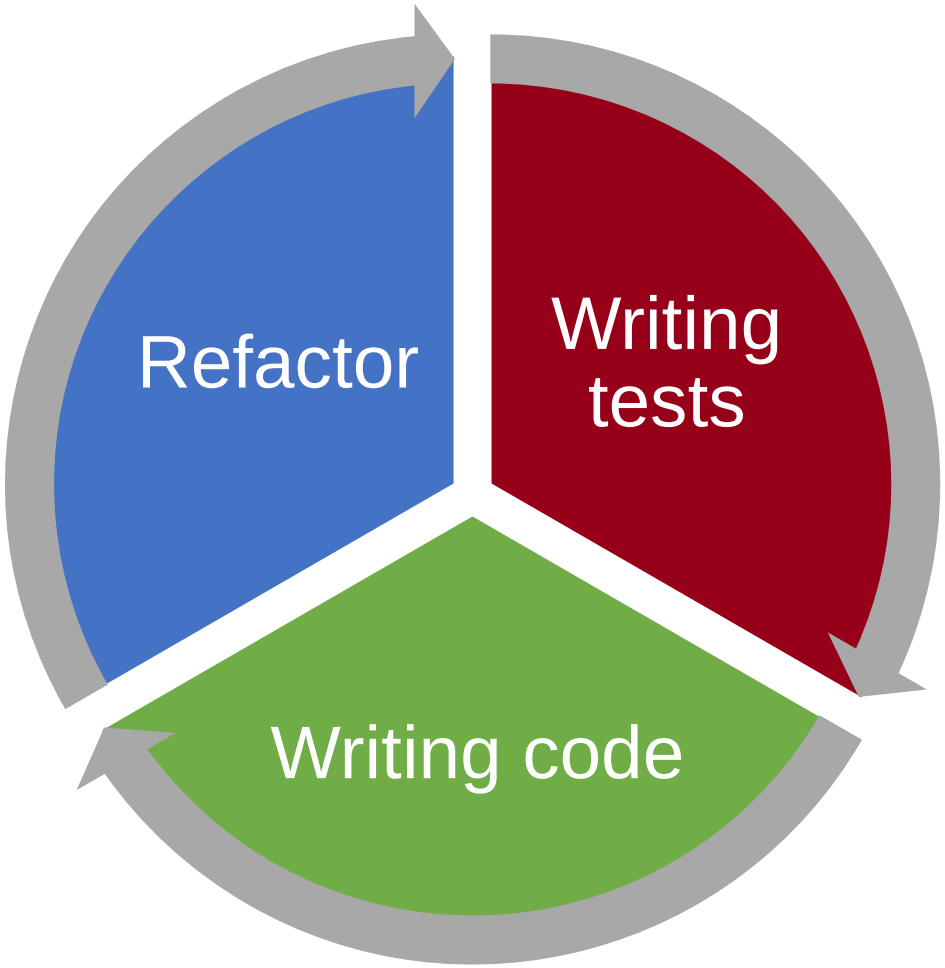
```
28
29  @Test
30  public void fireTorpedo_All_Success(){
31      // Arrange
32
33      // Act
34      boolean result = ship.fireTorpedo(FiringMode.ALL);
35
36      // Assert
37      assertEquals(true, result);    Expected [true] but was [false]
```

Expected [true] but was [false] fireTorpedo_All_Success()

Expected	Actual
-true	+false

```
38  }
```

Test-Driven Development



- Creating the code in multiple steps
  - Writing tests (for not yet existing code)
  - Writing code until the tests are passed
  - Refactor: new and existing code, test code
- We focus on the from outside observable behaviour of the code
- Quality of the tests is crucial

Definition of Done

Is the feature ready?

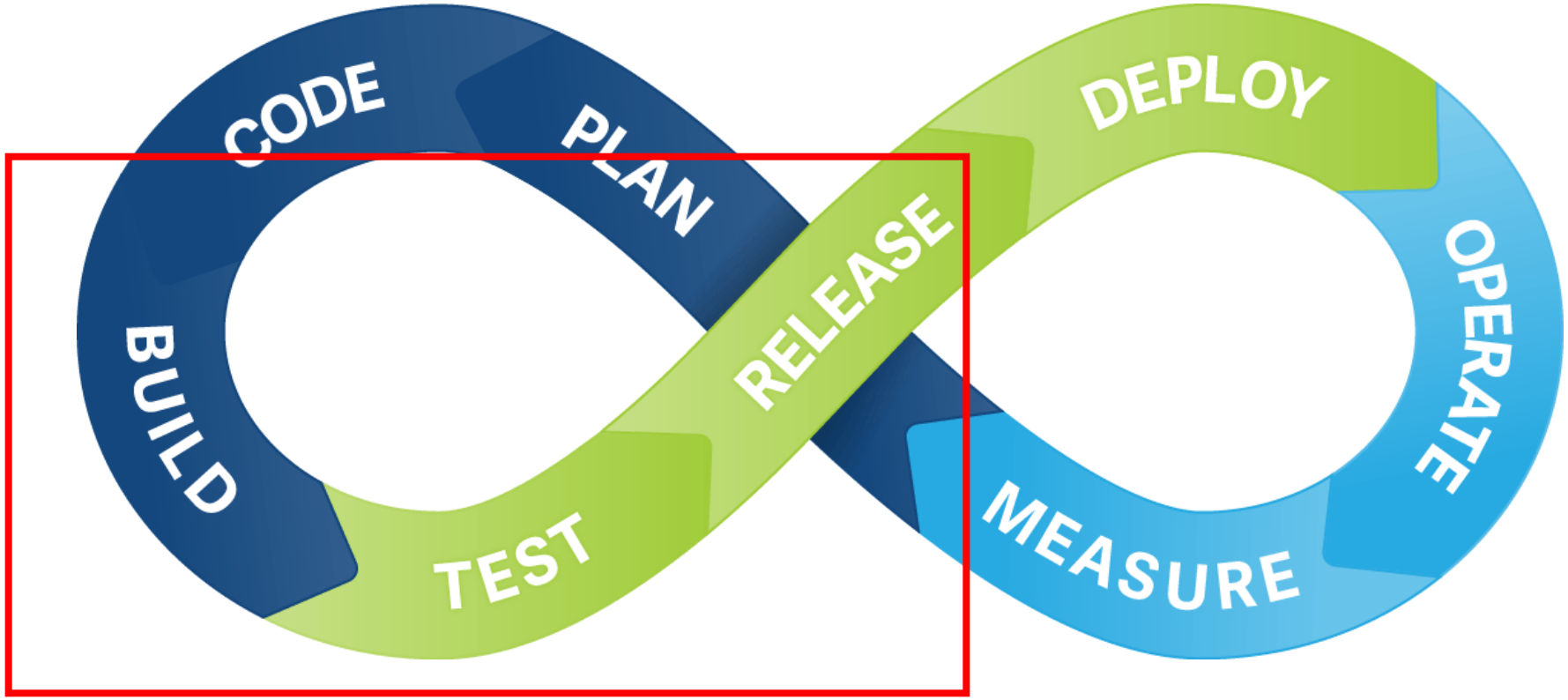
~~Of course, 95% are already done!~~
~~Nearly done, we almost started.~~
~~It works well, just …~~

Definition of Done

- Check list to see when it is “done”
- Code, tests, test results, deployment, …
- Depends on the project / group / method / …

See: https://www.agilealliance.org/glossary/definition-of-done

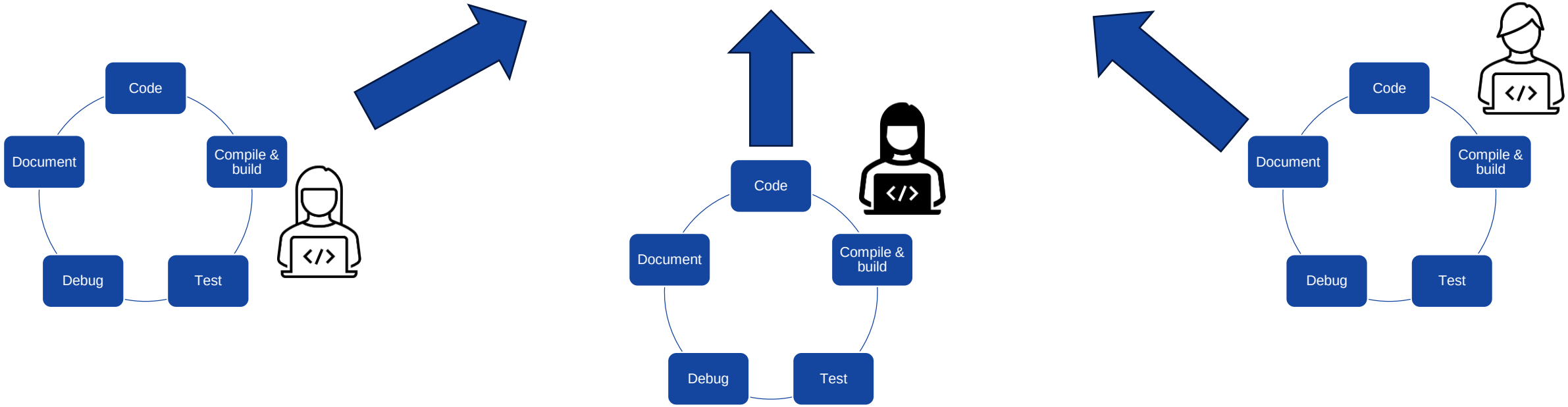
The Tasks of Software Development



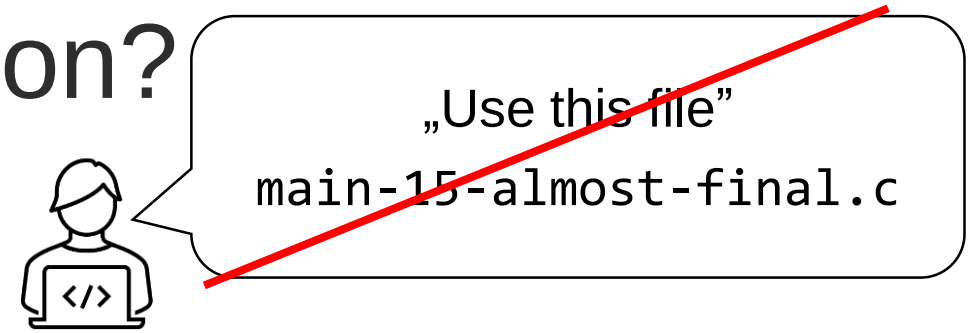
The Code Integration Problem

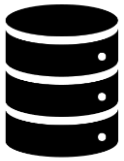
Until now, there was one single developer, …

…, but in the practice multiple developers work in parallel



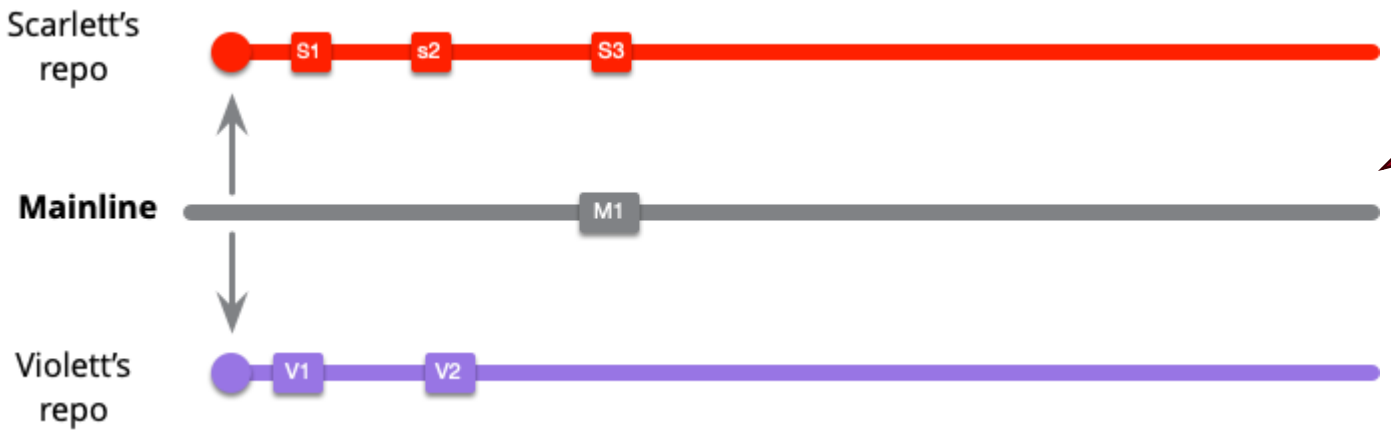
What Can Help in the Integration?





Version control system

- Identified versions of (any) files and their order
- Repository: different versions of files and their meta-data



BUT: version controlling cannot solve the integration alone
REQUIRES: agreed process

See the **Version management** lecture

Typical Code Integration Patterns

- Separate source code branch for each developer
  - Their own modifications collected there
- Merge: merging the modifications on two branches
  - If I am “ready”, I merge in my own modifications to make them visible
  - But: what if someone else made their own modifications → **merge conflict**

Code management and integration patterns:
When, where, who works and merges on which branches?

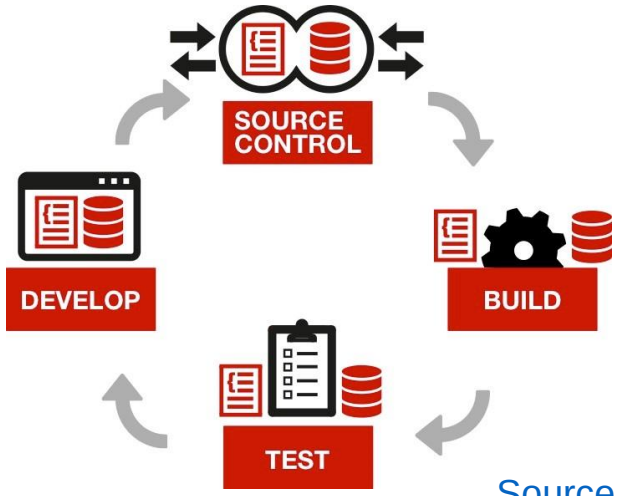
The more rare you integrate, the harder it is!
„if it hurts… do it more often"”

Continuous Integration (CI)

Continous Integration *a software development practice where members of a team integrate their work frequently, usually each person integrates at least daily*

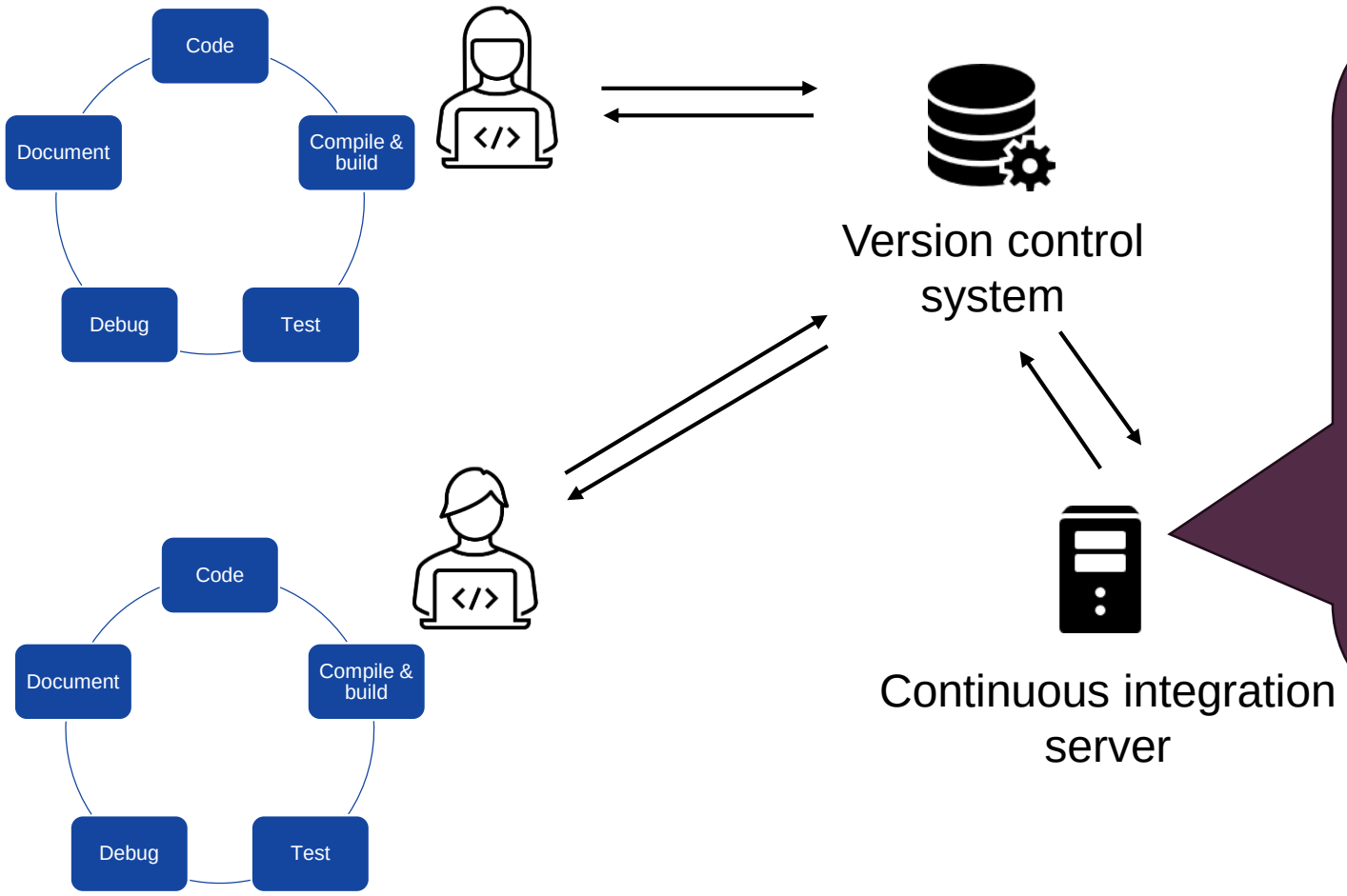
Martin Fowler

„Each integration is verified by an automated build (including test) to detect integration errors as quickly as possible. ”



Source

Supporting Continuous Integration



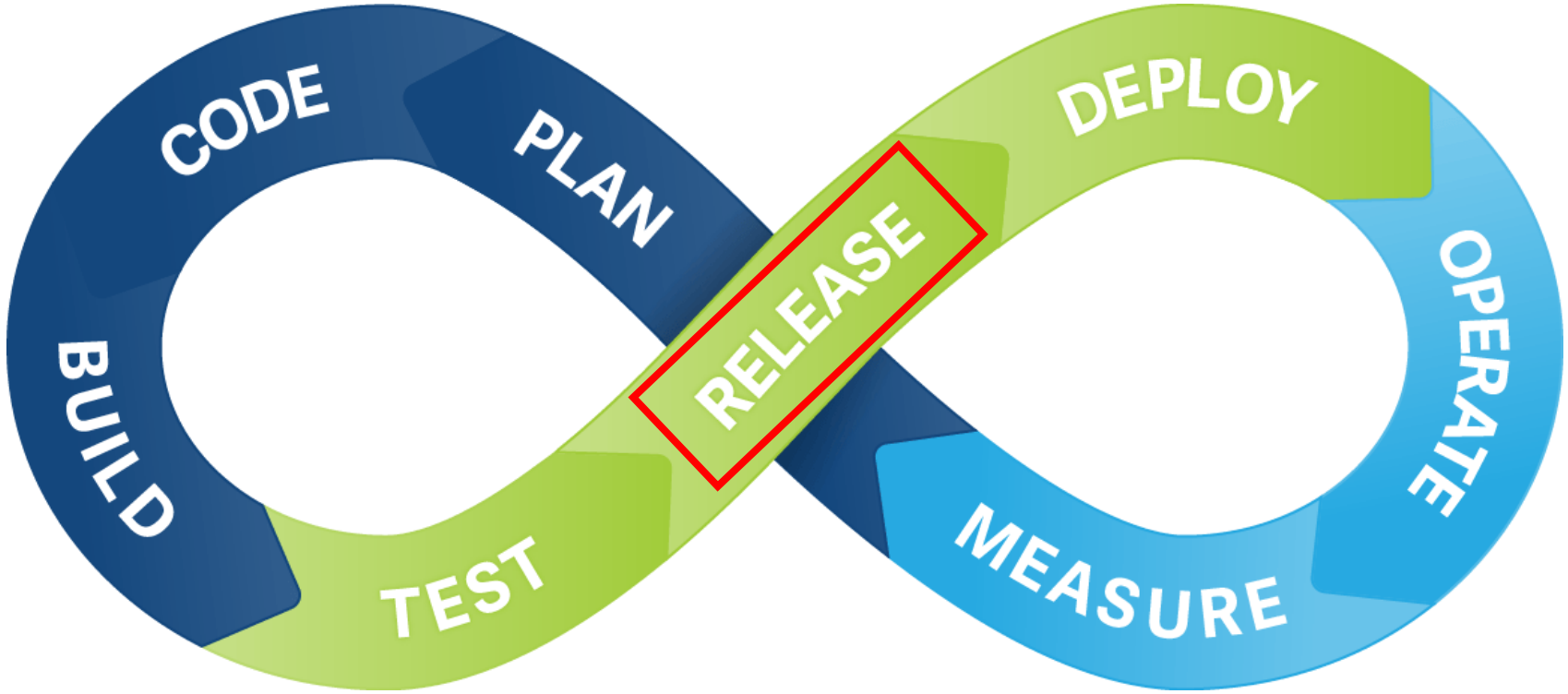
After each change:

- Recompile code, run (unit) tests
- (Almost) immediate feedback
- To not have any omitted files, or burnt in settings / local paths
- (Even if on several platforms)

GitHub Actions

Jenkins

The Tasks of Software Development



Assemble Releases

release particular version of a configuration item that is made available for a specific purpose

IEEE 828-2012

- Can be delivered to a customer, or deployed into a production environment
- Each necessary files (binary, config, database, documentation, …)
- Identified by a unique version (ID)

Versioning the Releases

- Several conventions or options, for example:

Semantic Versioning 2.0.0

Summary

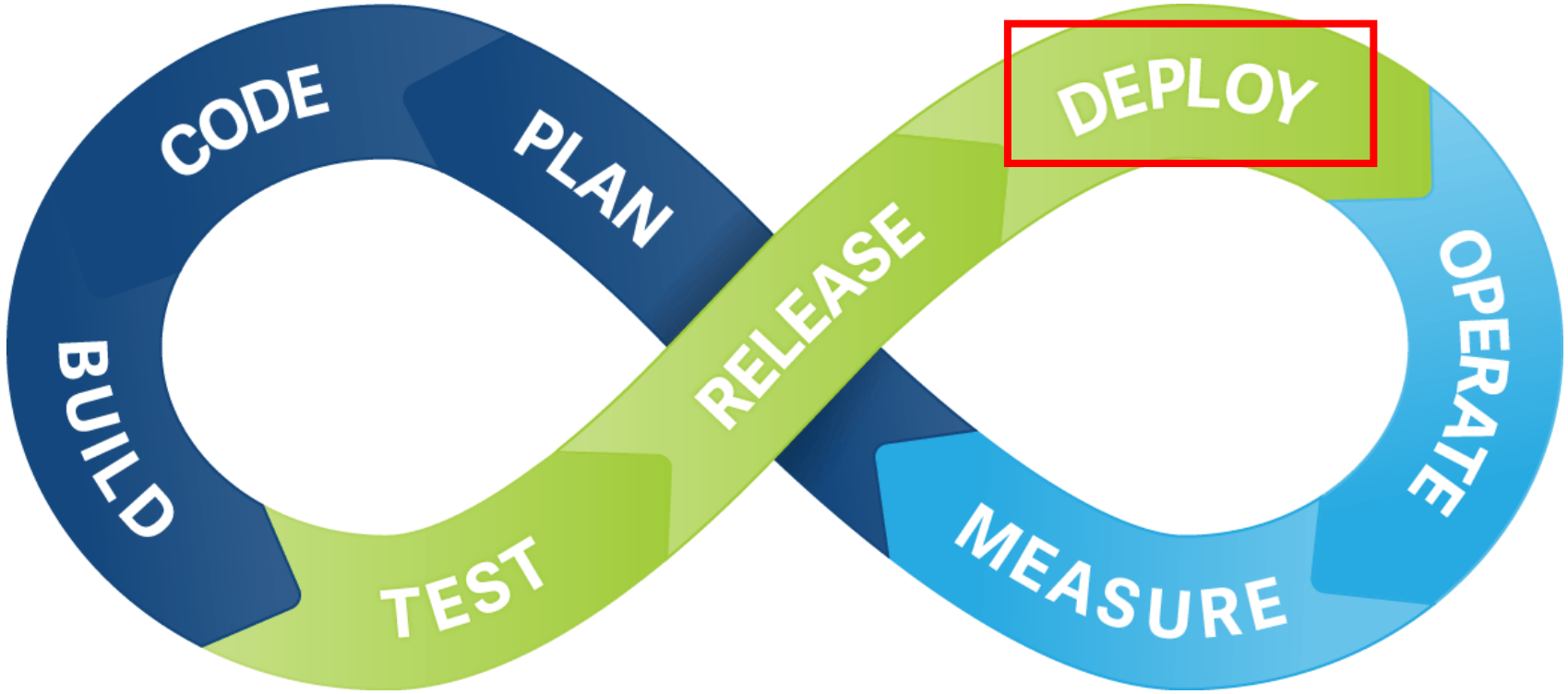
Given a version number MAJOR.MINOR.PATCH, increment the:

1. MAJOR version when you make incompatible API changes
2. MINOR version when you add functionality in a backward compatible manner
3. PATCH version when you make backward compatible bug fixes

Additional labels for pre-release and build metadata are available as extensions to the MAJOR.MINOR.PATCH format.

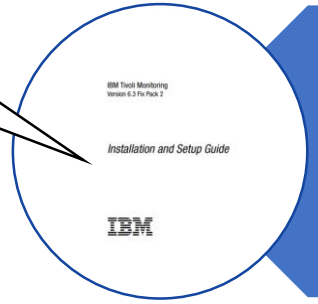
source: https://semver.org/

The Tasks of Software Development

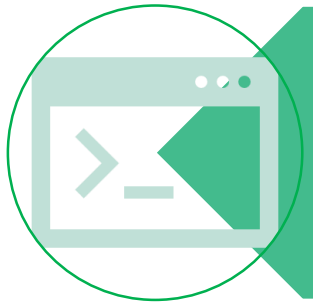


Deploy

954 pages!!!



Manual installation (based on an installation guide)



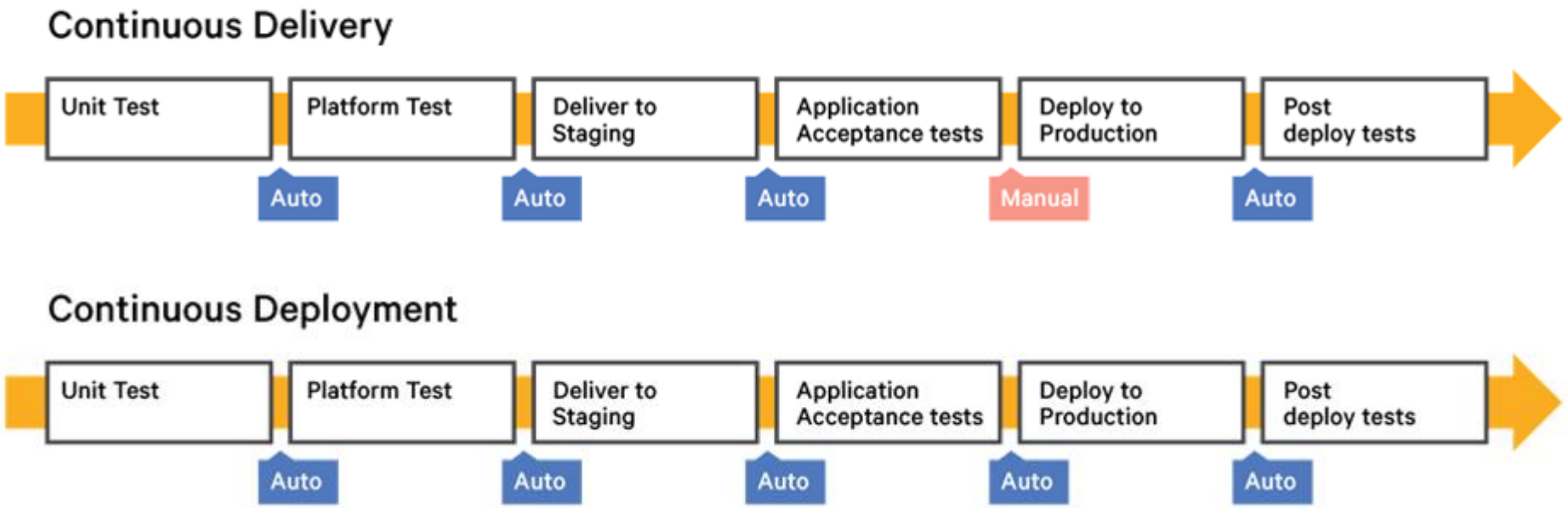
Automatized installation package / scripts

Infrastructure as Code (+ cloud resources)

Continuous Delivery (CD)

> „build software so that it is always in a state where it could be put into production”

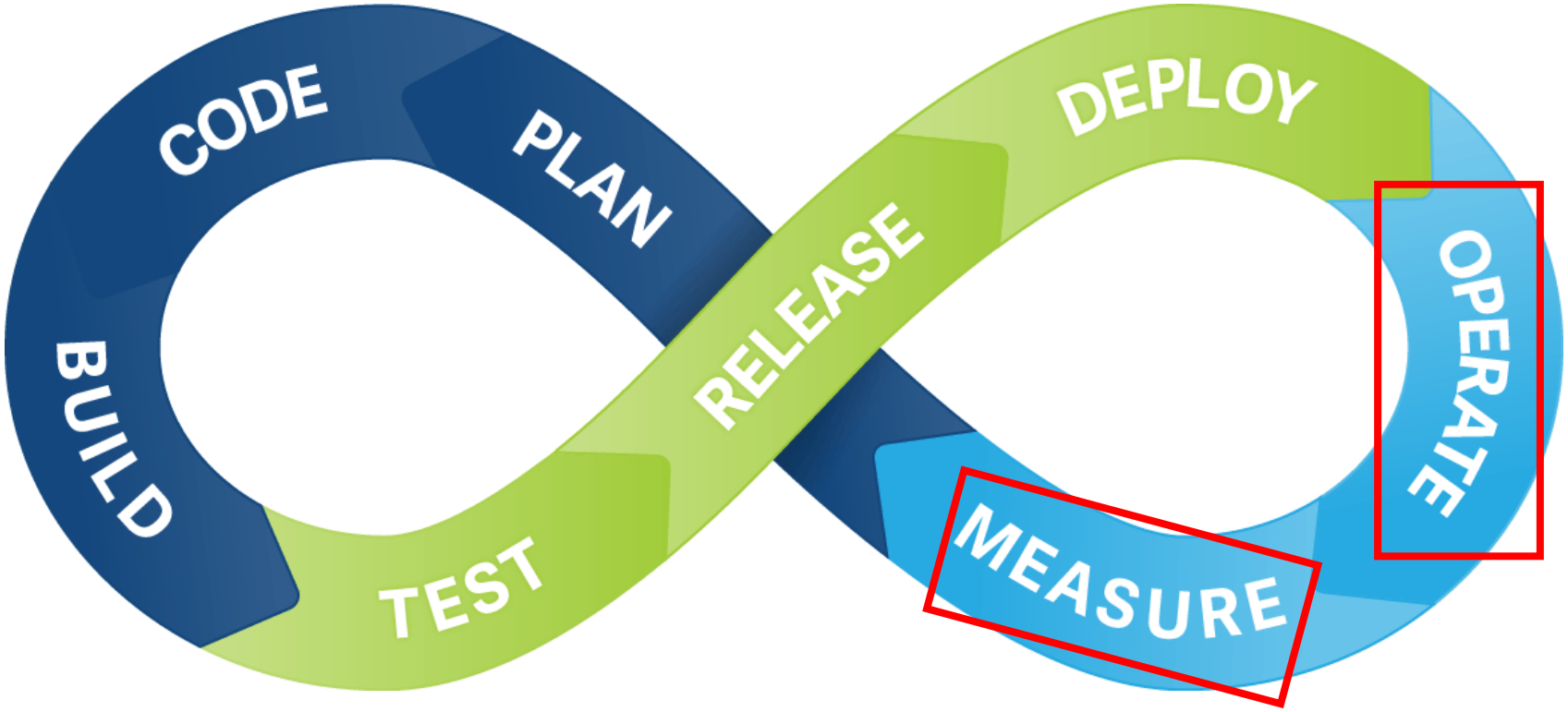
Forrás: https://martinfowler.com/bliki/ContinuousDelivery.html



delivery pipeline

Source

The Tasks of Software Development



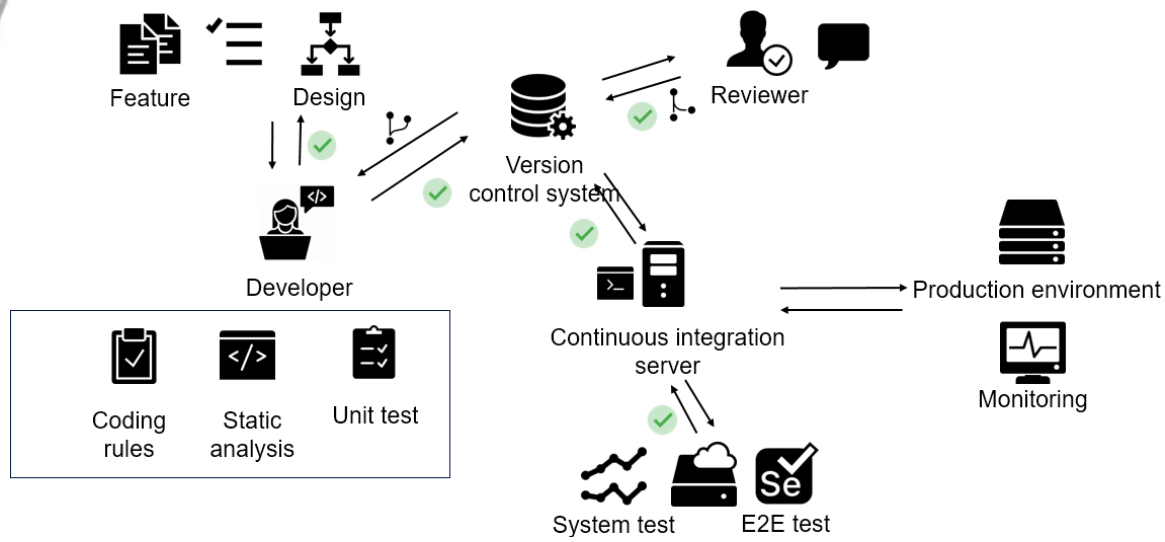
Operate and Measure

- Observing the software system in its real operation (HW, SW, business, … metrics)
- In case of any issues: Alarm (SMS, phone – even during the night)

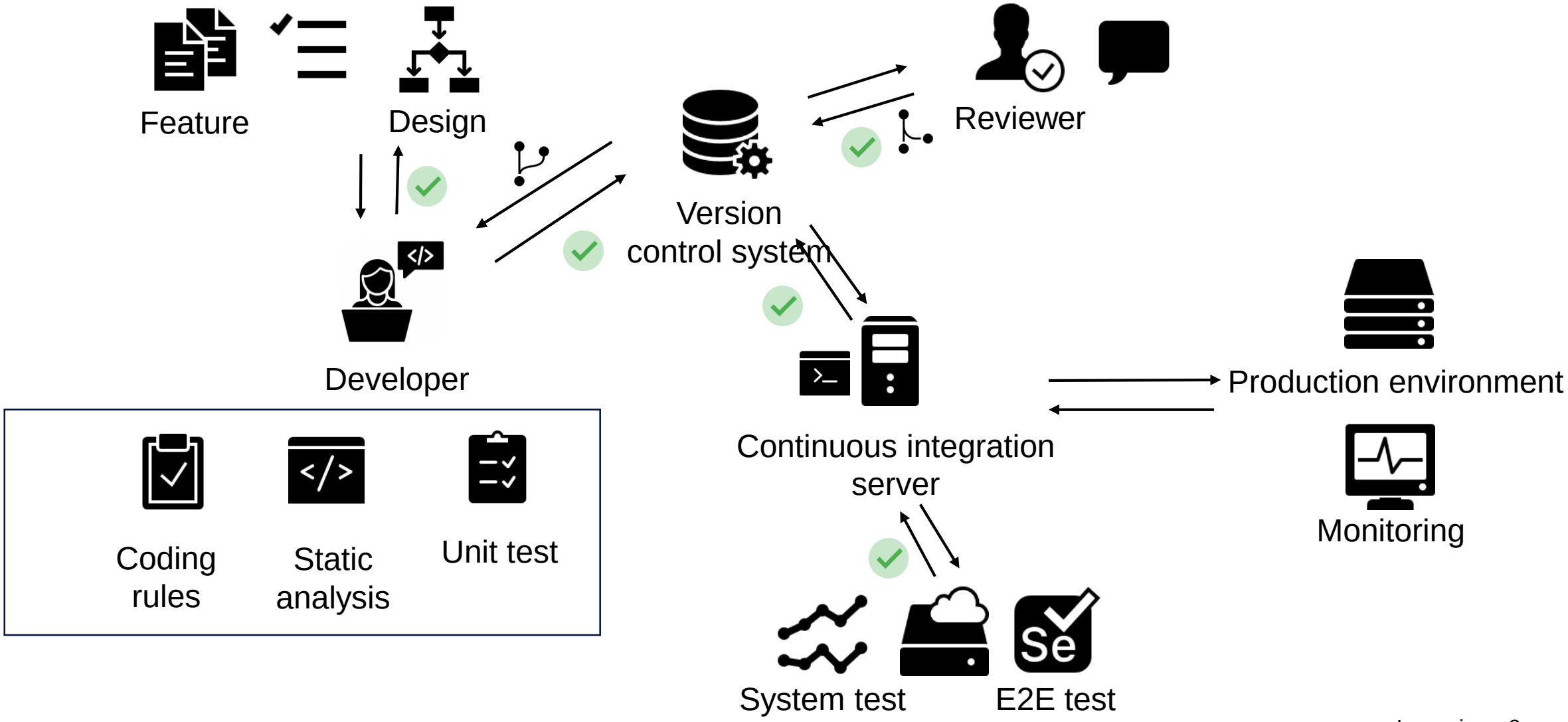
source

Development Workflow

Summary



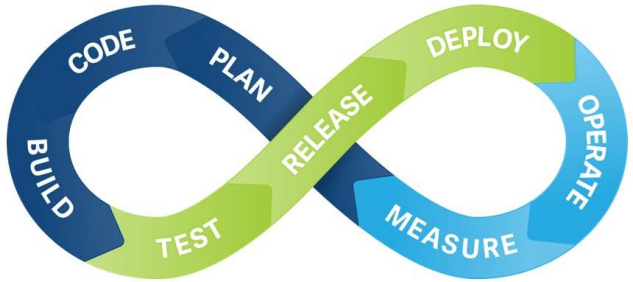
Typical Development Workflow



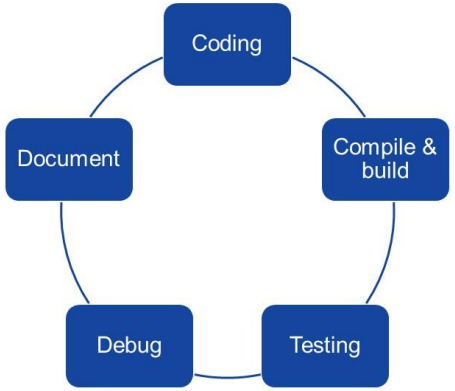
Icons: icons8.com

Summary

DevOps = Development + Operations



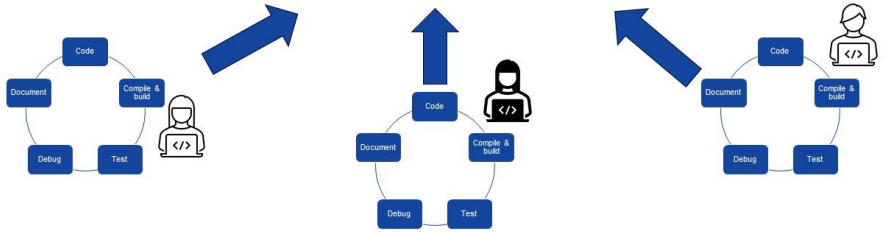
Get the Code Ready (in your local environment)



The Code Integration Problem

Until now, there was one single developer, …

…, but in the practice multiple developers work in parallel



Typical Development Workflow