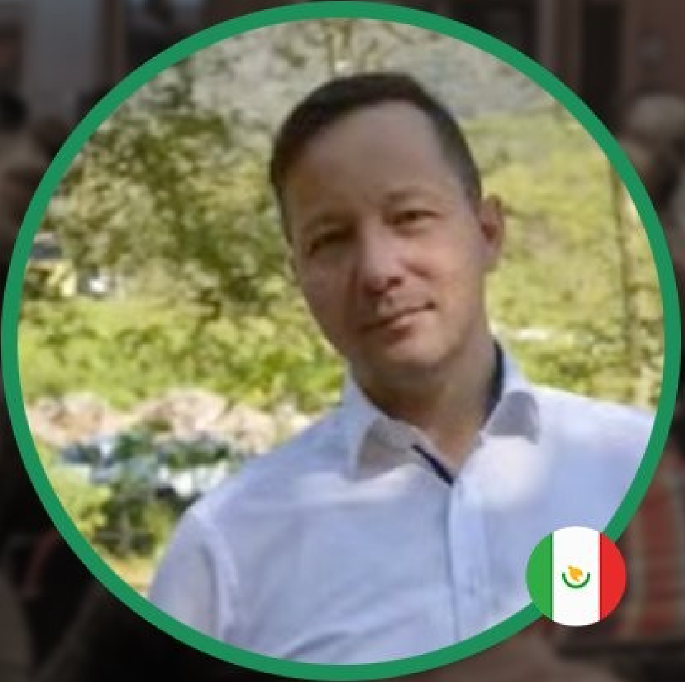
JCONF PERU 2022

MARCO SCHULZ

ROLLING STONES ON STAGE:

RELEASE ME

@ELMARDOTT



SPEAKER

ELMAR DOTT

(M. Schulz) studied at HS Merseburg, Germany, computer science and holds an engineers degree in software engineering. He tweets regularly about several technical topics. The main topics in his field of work are Build and Configuration Management, Software Architecture and Release Management.

About more than 15 years he is working in different large Web Application projects all over the world. He is an independent consultant / trainer. To share his knowledge he gives talks on conferences, if he is not writing on a new article about software engineering. https://elmar-dott.com

+ Consultant + Writer + Speaker + Trainer +

AGENDA

- Definitions & Meanings
- DevOps & Release Management
- Semantic Versioning
- Sprints & Agility
- DevOps Release & Deploy Strategies

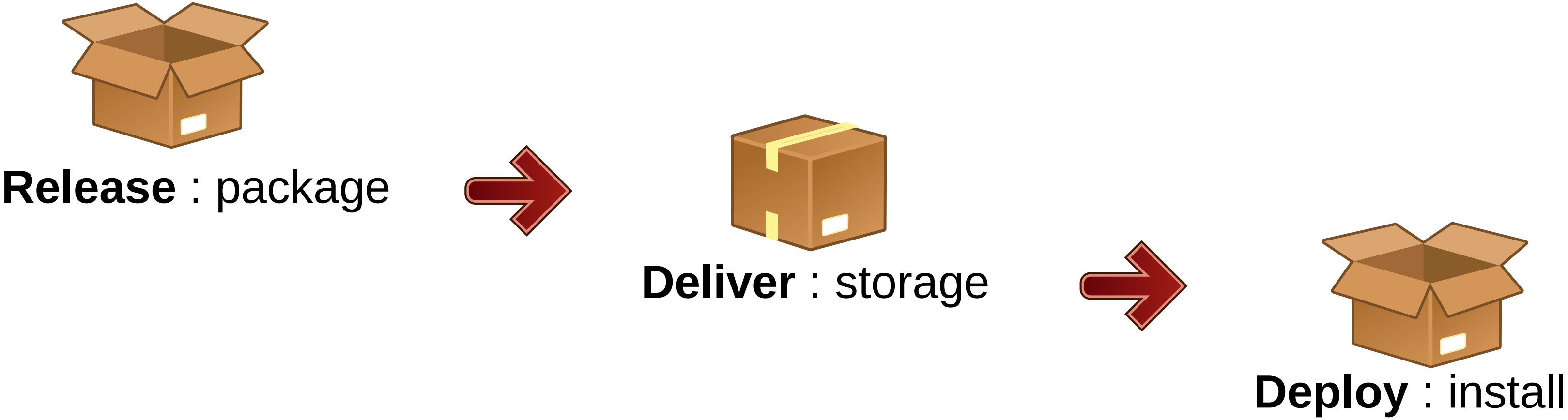
DEFINITIONS

Release: thus denotes a defined state of an artifact whose development cycle has been completed so that it is released for use.

Deliver: signifies the provision of the artifacts for further use, e. g. by installing on a repository server.

Deploy: means the installation of software in the target environment, primarily to PRODUCTION.

PROCESS CHAIN

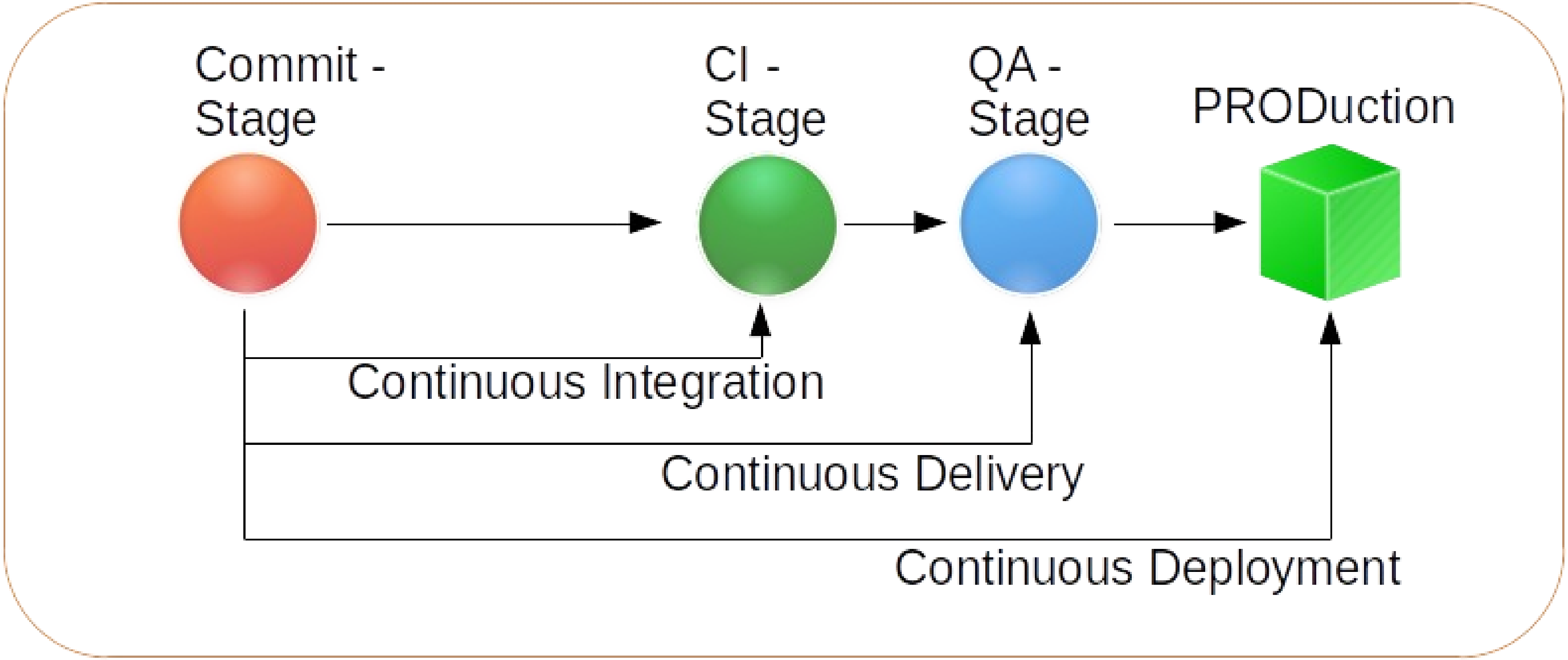


Release : package

Deliver : storage

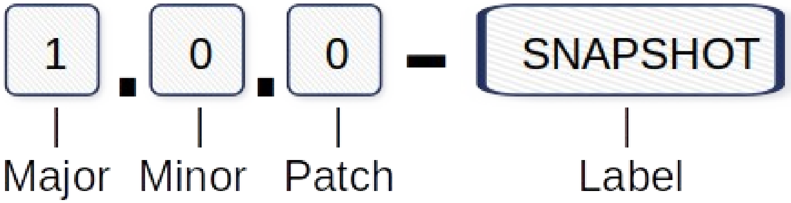
Deploy : install

LEVELS OF AUTOMATIZATION



SEMANTIC VERSIONING

Increasing of the Version number [MAJOR . MINOR . PATCH]:



1. **MAJOR** – incompatible API changes
2. **MINOR** – Adding new functionality, backward compatible
3. **PATCH** – Error corrections, backward compatible

Optional existing Labels, to mark Pre-Releases e.g. in Maven SNAPSHOT.

MYTHS & MISUNDERSTANDINGS

- Versioning applications can be individual (e. g. Firefox)
- SemVer should used by force for Modules, Libraries & Micro Services (Integration)
- Version numbers always belong in the artifact and cannot be changed!
- Summarize API changes and plan for the long term
- Build numbers should not used in version numbers

SINGLE POINT OF FAILURE

src/main/resources/org/europa/together/configuration/**version.properties**

```
version=${project.version}
```

src/main/java/org/europa/together/utils/**Constraints.java**

```
public static final String MODULE_VERSION = getVersion();

private static String getVersion() {
    String filePath
       = "org/europa/together/configuration/version.properties";
    PropertyReader propertyReader = new PropertyFileReader();
    propertyReader.appendPropertiesFromClasspath(filePath);
    return propertyReader.getPropertyAsString("version");
}
```

Source: https://github.com/ElmarDott/TP-CORE

Reproducibility & Traceability

To reproduce means to obtain a result that has already been achieved under the same conditions as often as desired and without deviations.

→ **Repetition**

Understanding means to fully understand how a result was achieved in order to be able to adjust it if necessary.

→ **Understanding**

Error Levels

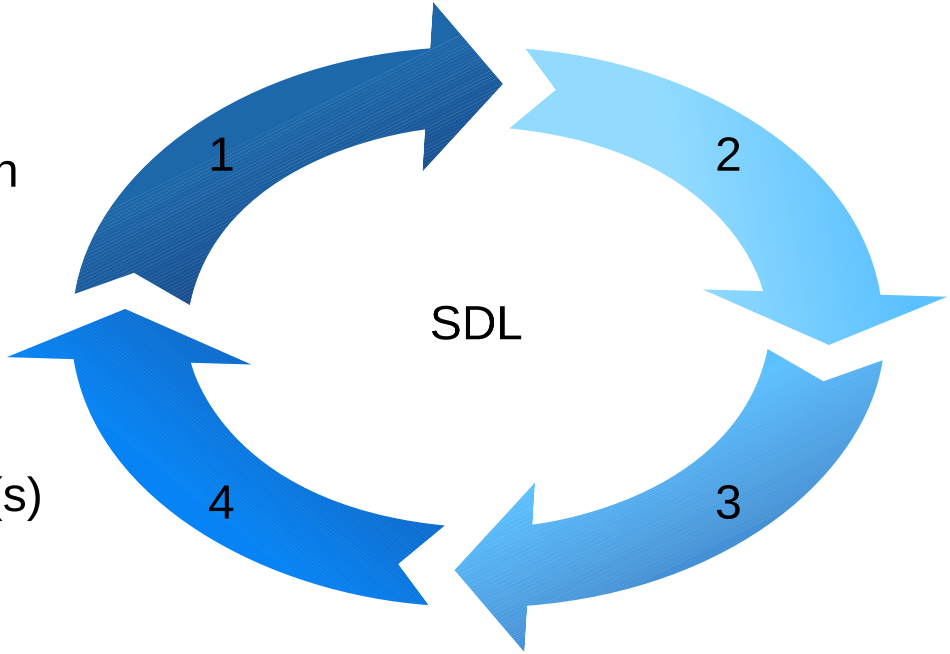
Error levels that can prevent the release from being used:

- ➔Simple Errors (GUI/ Layout/ Labeling)
- ➔Medium Errors (‚simple‘ Logic errors)
- ➔Heavy Errors (Acceptance preventing)

~~Release Candidate~~ → **Production Candidate**

Sprint vs. Release

1 Iteration = 1 Sprint = *n* Weeks = Release



Planning / Conception

1

2

Implementation

SDL

Operation(s)

4

3

Test

PROJECT PLAN

Weeks	01	02	03	04	05	06	07	08	09	Version
Sprint 1	D	T								1.1.0
Sprint 2		D	T							1.2.0
Sprint 3			D	T						1.3.0
Sprint 4				C/T						**1.4.0 Prod.Cand.**
Sprint 5				D	T					1.5.0
Sprint 6					D	T				1.6.0

D = Develop / T= Test / C = Corrections

TEST STAGES

DEVELOP

Line A: 1.1.0

Hourly Builds (Commit Stage)
Permanent compile and Unit Tests.

INTEGRATION

Line A: 1.1.0

Nightly Build (Integration & Deploy Tests)
The application is running. Preparing for Release

ACCEPTANCE

Line A: 1.1.0

External Quality Check (Retest)
Released – BugFixes

PRODUCTION

Line A: 1.1.0

Live System
Released – only critical BugFixes

Release Types & Deploy Strategien

Blue/ Green Deployment: Redundant deployment with rollback option.

Canary Releases: gradual Roll-out

A/B Tests: Evaluation of the conversion (user acceptance) according to defined criteria.

Feature Toggles: Turning functionality on and off.

REFERENCS

[1] Marco Schulz, 2021, Continuous Integration mit Jenkins, Rheinwerk, ISBN: ISBN 978-3-8362-7834-8 https://www.rheinwerk-verlag.de/continuous-integration-mit-jenkins/

[2] https://www.youtube.com/watch?v=RVCLIP7vOzo

[3] Sematic Versioning: https://semver.org

[3] https://dzone.com/articles/version-number-anti-pattern

[4] https://elmar-dott.com/articles/modern-times/

[5] https://elmar-dott.com/articles/automatisierungs moglichkeiten-im-software-konfigurations-management/

CREDENTIALS

BUSSINESS ITC SOLUTIONS

Dipl. Inf. Marco Schulz

Expert for Enterprise Applications

CONSULTANT • TRAINER • SPEAKER • WRITER

```
------------------------------------------------------------
Homepage      : https://elmar-dott.com
GitHub        : https://github.com/ElmarDott
AnchorFM      : https://anchor.fm/elmar-dott
Twitter       : https://twitter.com/ElmarDott
Speaker Deck  : https://speakerdeck.com/elmardott

Lbry          : https://lbry.tv/@elmar.dott:8
BitChute      : https://www.bitchute.com/channel/3IyCzKdX8IpO/
------------------------------------------------------------
```

Danke / tank you / Gracias

creative commons

open source

ubuntu MATE

Libre Office
The Document Foundation

BITCHUTE

HECHO EN MÉXICO