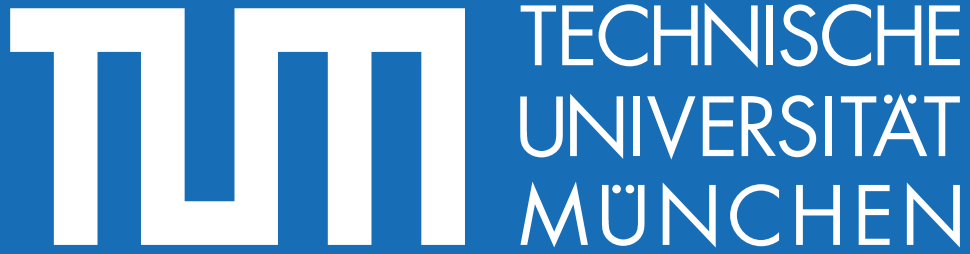


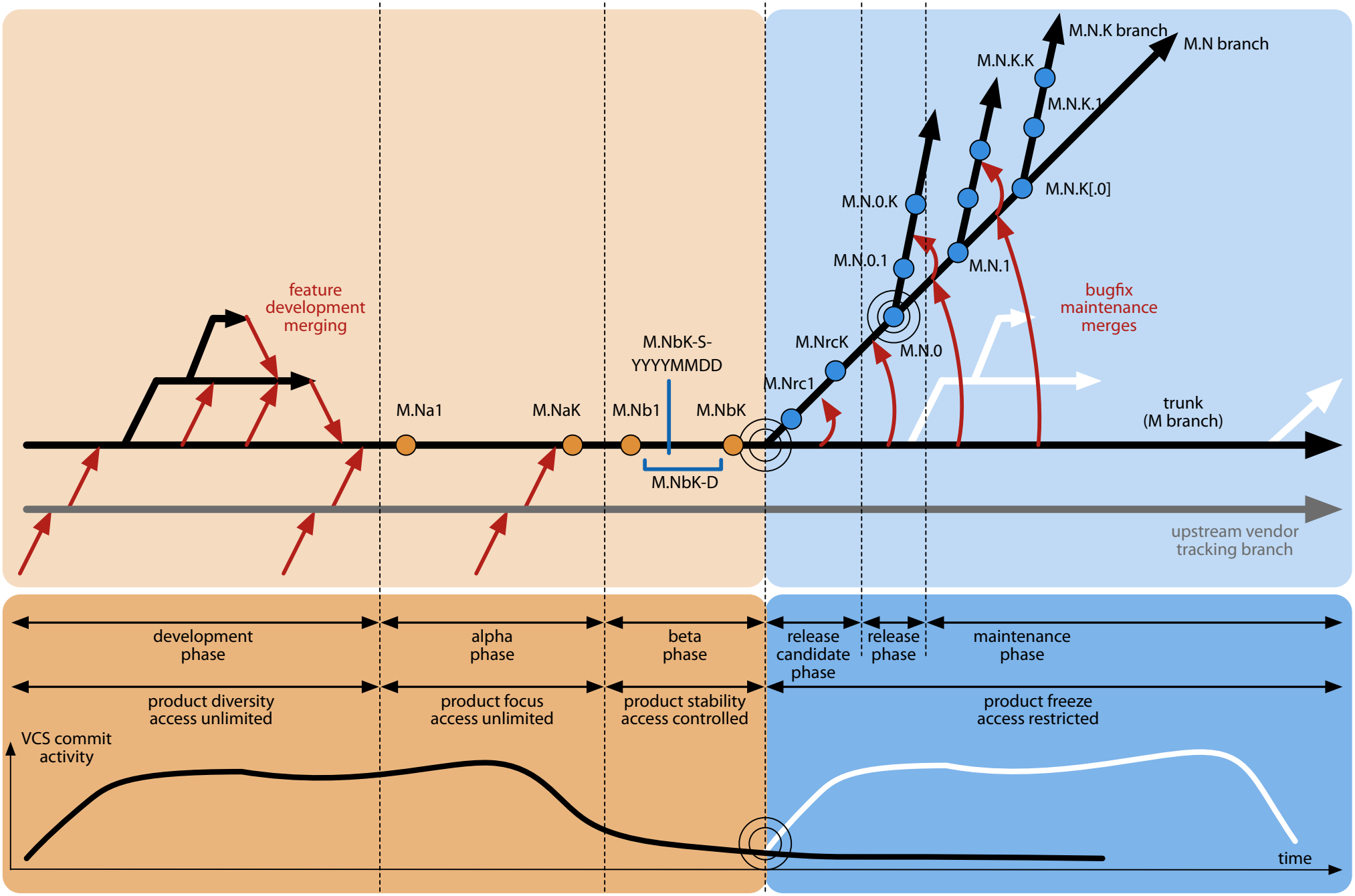
Software Engineering in Industrial Practice (SEIP)

Dr. Ralf S. Engelschall

ARCHITECTURE FUNDAMENTALS

Version Control Architecture

TECHNISCHE UNIVERSITÄT MÜNCHEN

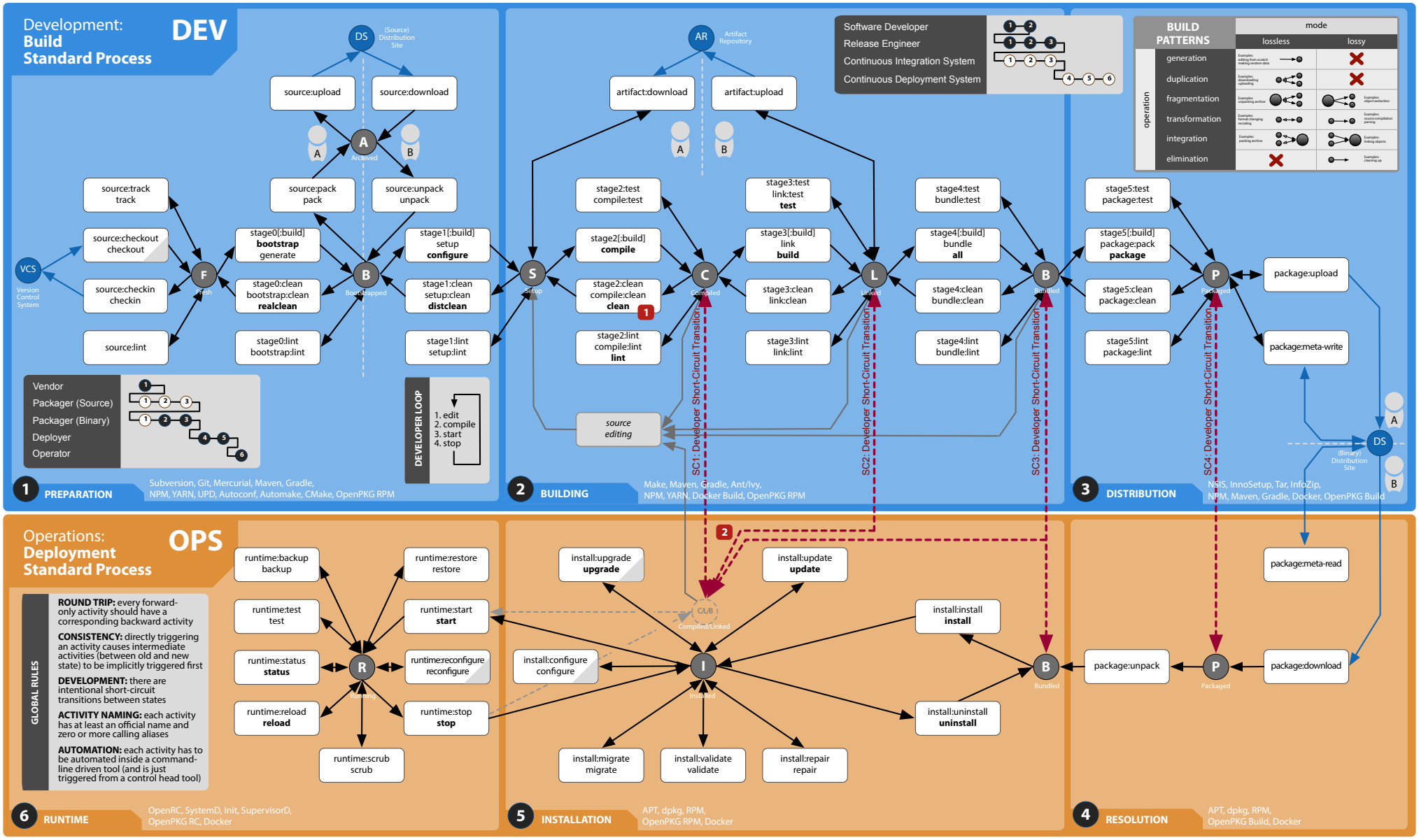


Intellectual Content: Version 1.3.0 (2012-09-19), Authored 2011-2012 by Dr. Ralf S. Engelschall
Graphical Illustration: Version 1.2.1 (2012-09-19), Copyright © 2008-2012 Dr. Ralf S. Engelschall <http://engelschall.com>, All Rights Reserved
Unauthorized Reproduction Prohibited. Licensed to Technische Universität München (TUM) for reproduction in Computer Science lecture contexts only.

ARCHITECTURE FUNDAMENTALS

Assembly Process Architecture

TECHNISCHE UNIVERSITÄT MÜNCHEN



Intellectual Content: Version 1.3.3 (2020-12-19). Authored 2009-2020 by Dr. Ralf S. Engelschall
Graphical Illustration: Version 1.3.2 (2023-11-26). Copyright © 2009-2023 Dr. Ralf S. Engelschall <http://engelschall.com>. All Rights Reserved.
Unauthorized Reproduction Prohibited. Licensed to Technische Universität München (TUM) for reproduction in Computer Science lecture contexts only.

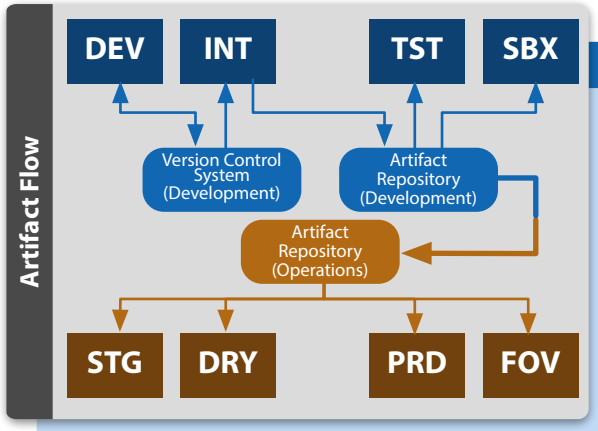
ARCHITECTURE FUNDAMENTALS

Environments & Quality Assurance

TECHNISCHE UNIVERSITÄT MÜNCHEN

AF 14.3

Intellectual Content: Version 1.06 (2023-11-26), Authored 2018-2023 by Dr. Ralf S. Engelschall
Graphical Illustration: Version 1.04 (2019-09-15), Copyright © 2018-2019 Dr. Ralf S. Engelschall <http://engelschall.com>, All Rights Reserved
Unauthorized Reproduction Prohibited. Licensed to Technische Universität München (TUM) for reproduction in Computer Science lecture contexts only.



Development Environments

DEV Development

(Potentially destructive) Environment for the (separated) development of all artifacts of the system, typically located on each developer's respective system.

INT Integration

(Potentially destructive) Environment for integrating all artifacts of the system, usually located centrally to the developers and driven automatically.

TST Test

Environment for testing the system as a whole in a production-resembling context.

SBX Sandbox

Destructive environment mirroring the Test environment for demonstration and experimentation purposes, including trainings and showcases.

Operations Environments

STG Staging

Environment for staging the system as a whole in a production-equal context in order to deploy it to the Production environment subsequently.

PRD Production

Environment for running the system as a whole in production.

DRY Dry-Run

Destructive environment mirroring the Staging environment for experimentation purposes, including operating system and major dependency upgrades.

FOV Failover

Destructive environment mirroring the Production environment for failover situation, including disaster recovery situations.

Environment Mirroring

Replicated Environments

Original Environments

Testing Targets:

Quality Promise, Bugfix Effort

Testing Approaches

UT Unit Testing

Whitebox testing the inner details of individual units (components) against their functional specification.
Specialization: Regression Testing

IT Integration Testing

Whitebox testing the outer interplay of individual units (components) against the technical design of the solution.
Specialization: Smoke Testing

ST System Testing

Blackbox testing the system as a whole against the functional and non-functional requirements of the solution.
Specialization: Load Testing

UAT (User) Acceptance Testing

Blackbox testing and formal approval of the system as a whole against the end-to-end functionality and user experience requirements.

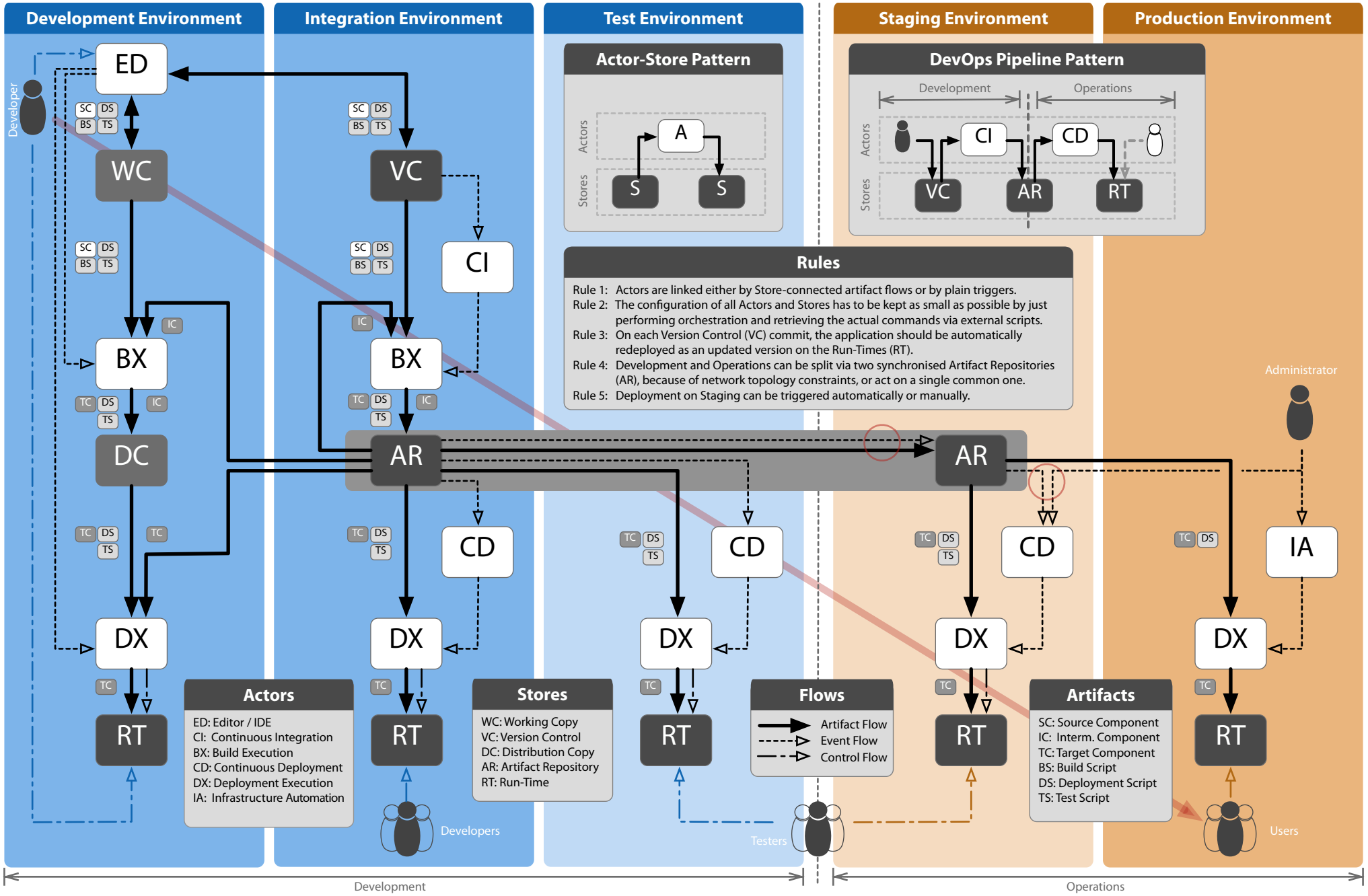
OT Operation Testing

Blackbox testing the system as a whole against the availability and proper operation of the intended services.
Specialization: Service Monitoring

ARCHITECTURE FUNDAMENTALS

DevOps Toolchain

TECHNISCHE UNIVERSITÄT MÜNCHEN



AF 14.4

Intellectual Content: Version 1.0.2 (2020-12-19), Authored 2019-2020 by Dr. Ralf S. Engelschall with input from Christian Reiber and Dr. Thomas Schöpf
Graphical Illustration: Version 1.0.2 (2020-10-11), Copyright © 2019-2020 Dr. Ralf S. Engelschall <http://engelschall.com> All Rights Reserved.
Unauthorized Reproduction Prohibited. Licensed to Technische Universität München (TUM) for reproduction in Computer Science lecture contexts only.

ARCHITECTURE FUNDAMENTALS

Software Release Management

TECHNISCHE UNIVERSITÄT MÜNCHEN

AF 15.2

Evolution Stage — What?

WF Wireframe
Distraction-free low-fidelity illustration of the solution and its base features, displaying its pure structure and core functionality only.

WS Walking Skeleton
Realization of all technical, fundamental aspects of the solution, ensuring the domain specific aspects can be realized later on top of it without problems.

PT Prototype
High-fidelity mostly interactive sample, mockup, model or simulation of the solution and its base features, showcasing its structure and functionality.

MVP Minimum-Viable Product
Early version of solution with just enough functionality to enable full turn of Build-Measure-Learn loop with minimal amount of effort and time.

PoC Proof of Concept
Pure realization of most-risky aspects of the solution, proofing their feasibilities. Might still be based on a different technology than WS, MVP and FP.

FP Full Product
Final version of the solution with all intended functionality and targeting the mainstream market.

arbitrary technology | target technology

Version Number — When?

M Major Version
Major Version of solution. Usually bumped on major technical or domain-specific changes only. A bump resets the Minor Version and the Revision, too.

N Minor Version
Minor Version of the solution within the Major Version. Usually bumped on new features. A bump resets the Revision, too.

R Revision
The Revision of the Release Phase within Major and Minor Version. Bumped for every A/B/C/R Release Phase.

Release Phase (p) — When?

A Alpha
Early version of the solution with incomplete and unstable functionalities to get feedback on product. Usually tagged as "M.NaR" (R > 0).

C Candidate
Mature version of solution with complete and stable functionalities to catch last-minute problems. Usually tagged as "M.NrcR" (R > 0) around RTM.

B Beta
Early version of the solution with complete but still unstable functionalities to stabilise product. Usually tagged as "M.NbR" (R > 0).

R Release
Release version of the solution with complete and stable functionalities, available for production use. Usually tagged as "M.N.R" (R >= 0).

Points-In-Time (PiT) — When?

DEV Development
Arbitrary permanent points-in-time during development. This is the default tag for the source code. Intended for no availability releases.

SNP Snapshot
Distinct temporary point-in-time for a release of the current version without a version increase. Intended for limited availability releases.

REL Release
Distinct temporary point-in-time for a release of the current version with a version increase. Intended for early and general availability releases.

Product Edition — Which?

CE Community Edition
Edition of the solution for the Open Source Community. Contains just the base functionality and has limited volunteering support.

STD Standard Edition
Edition of the solution with just the standard functionality and regular support.

EE Enterprise Edition
Edition of the solution for the Enterprise market. Contains the base and additional functionality and has full commercial support.

PRO Professional Edition
Edition of the solution with both the standard and extra functionalities and extended support.

Availability Scope (S) — Who?

XA No Availability
No public availability of solution at all. The scope for all Development and sometimes Snapshot point-in-times.

EA Early Availability
Early public availability of solution for early market. Usually for Beta or Release Candidate levels or for Release and initial Release Update levels.

LA Limited Availability
Limited public availability of solution. Usually for releases after the End-of-Life-Announcement (EOLA) or for releases with specific customer features.

GA General Availability
Late public availability of solution for mainstream market. Usually for Release and sometimes just for Release Update levels.

Distribution Channel — Where?

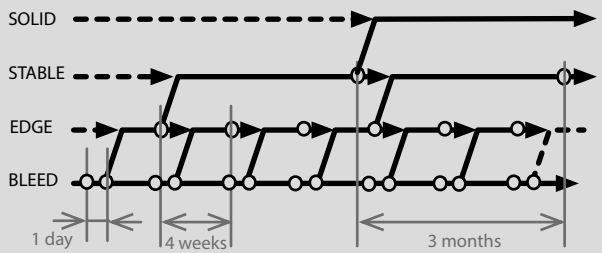
BLEED Bleed Channel
Distribution channel for all daily snapshots ("YYYY.MM.DD") with experimental features turned on. Intended for testing purposes only.

STABLE Stable Channel
Distribution channel for all quarterly releases ("YYYY.QN") with experimental features turned off. Intended for fast mainstream market and production use.

EDGE Edge Channel
Distribution channel for all monthly releases ("YYYY.MM") with experimental features turned on. Intended for early market or testing purposes.

SOLID Solid Channel
Distribution channel for all (half-)year releases ("YYYY[.N]") with experimental features turned off. Intended for slow mainstream market and production use.

Distribution Channel Artifacts



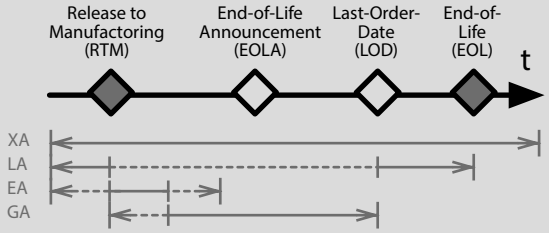
Versioning Scheme (stdver.org)

M . N p R [. D] [+ H] [- S]

Examples:

```
1.2a3.20230101+F42A-XA
1.2b3-EA
1.2.3-GA
1.2.3
```

Product Life-Cycle



Intellectual Content: Version 1.0.8 (2019-11-16), Authored 2018-2019 by Dr. Ralf S. Engelschall
Graphical Illustration: Version 1.1.7 (2023-11-26), Copyright © 2018-2023 Dr. Ralf S. Engelschall <http://engelschall.com>, All Rights Reserved
Unauthorized Reproduction Prohibited. Licensed to Technische Universität München (TUM) for reproduction in Computer Science lecture contexts only.