POM 16 – Release Management

**Terminology**

• **Continuous** **integration**: technique where members of a team integrate their work frequently. Usually each person integrates at least daily, leading to multiple integrations per day.

• **Continuous** **delivery**: approach in which teams keep producing valuable software in short cycles and ensure that the software can be reliably released at any time.

• **Continuous** **deployment**: every change that passes automated tests is deployed automatically.

• **Continuous** **software** **engineering**: organizational capability to develop, release and learn from software in short cycles.

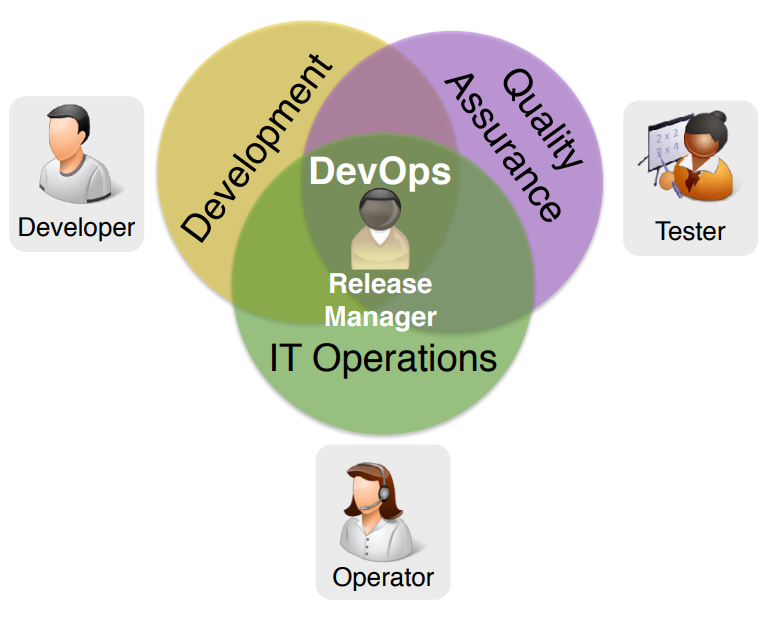
**Release Manager**

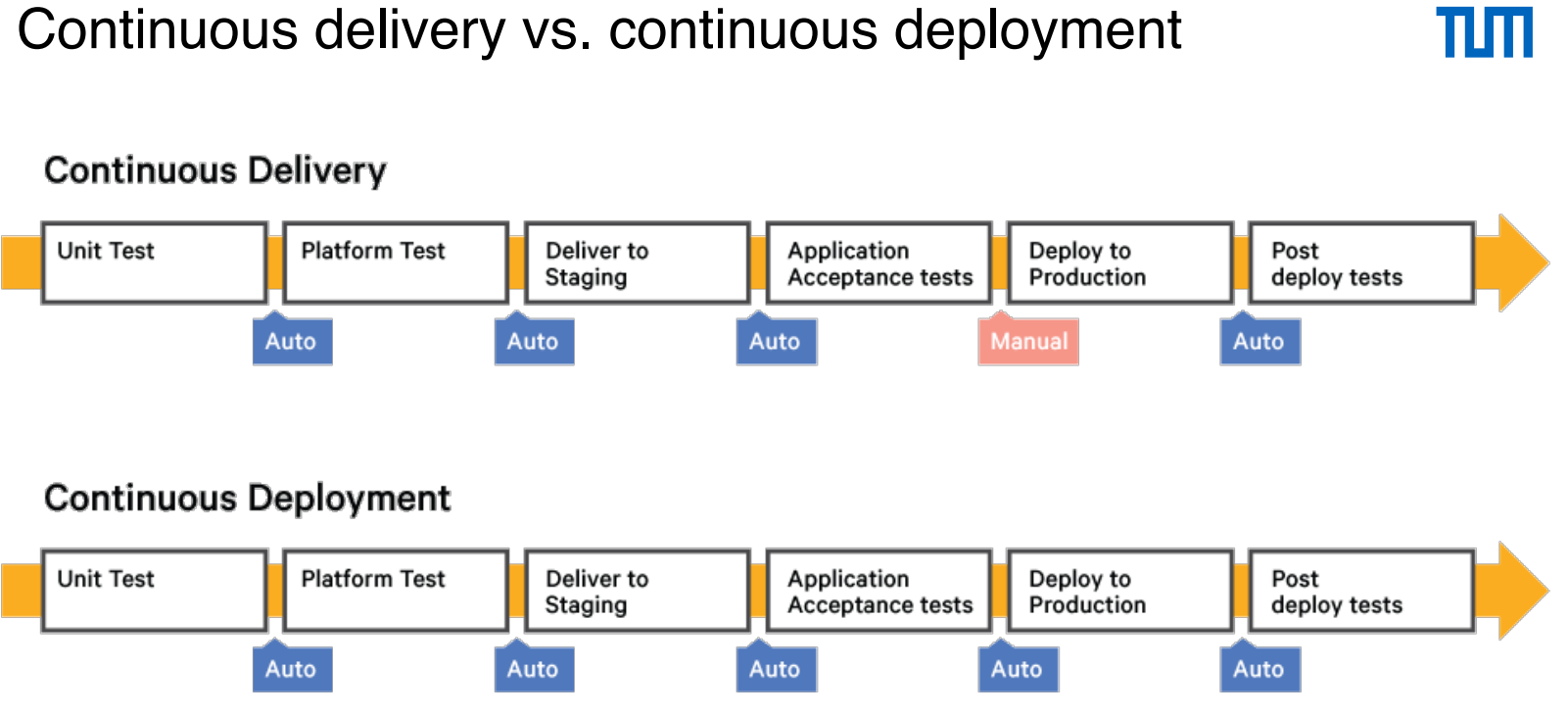
* Release management is the process of managing software releases from development to the user
* A **release manager** has different responsibilities:
  + **Facilitator**: Connection between different business units to promote smooth and timely delivery of software products or updates.
  + **Coordinator**: coordinate disparate source trees, projects, teams and components
* Development and operations departments work together: DevOps

**DevOps**

* DevOps (**Dev**elopment + **Op**erations)

promotes communication, collaboration, integration and automation

* A release manager is on duty (im Einsatz)



**Continuous delivery: benefits and challenges**

* Benefits

+ Accelerated (beschleunigen) time to market

+ Building the right product: improved product quality and customer satisfaction

+Improved productivity and efficiency

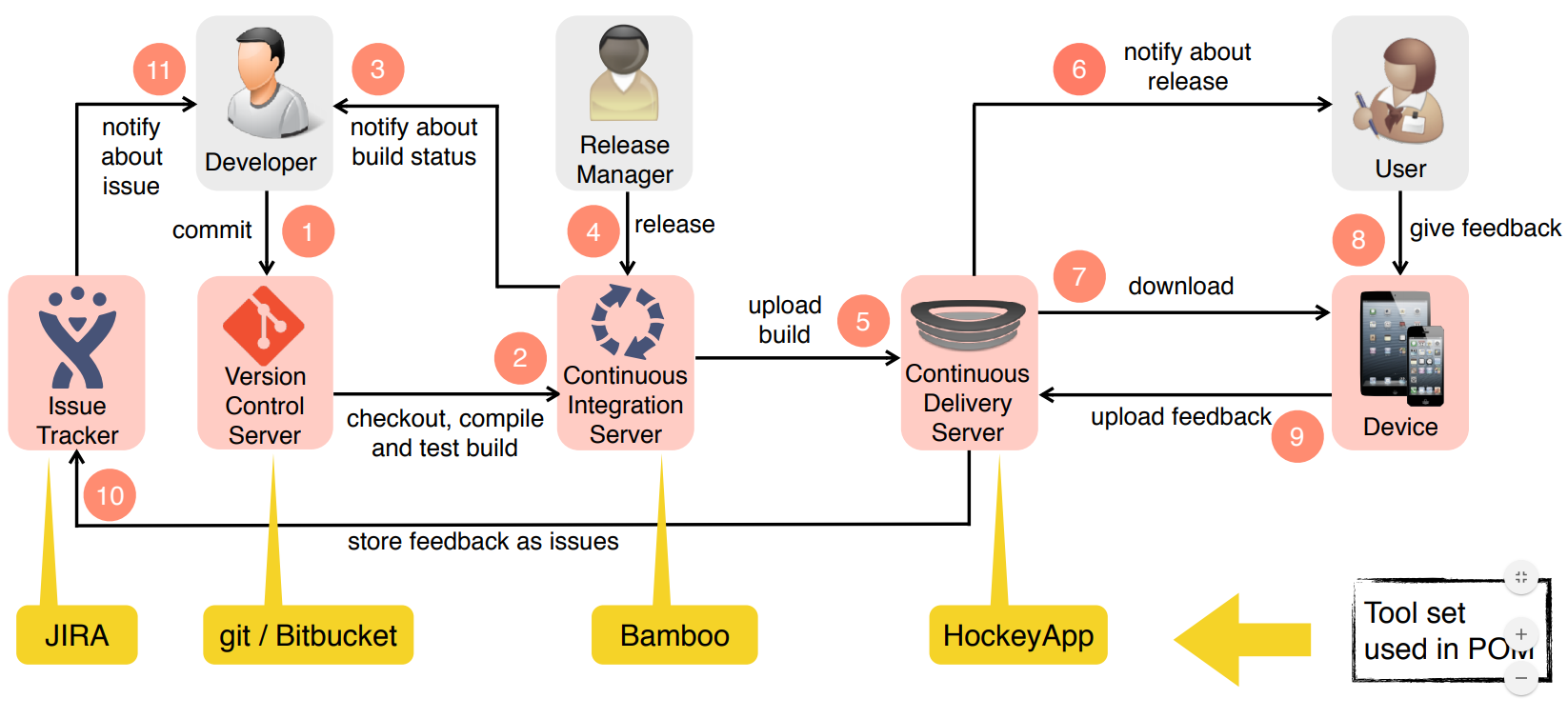
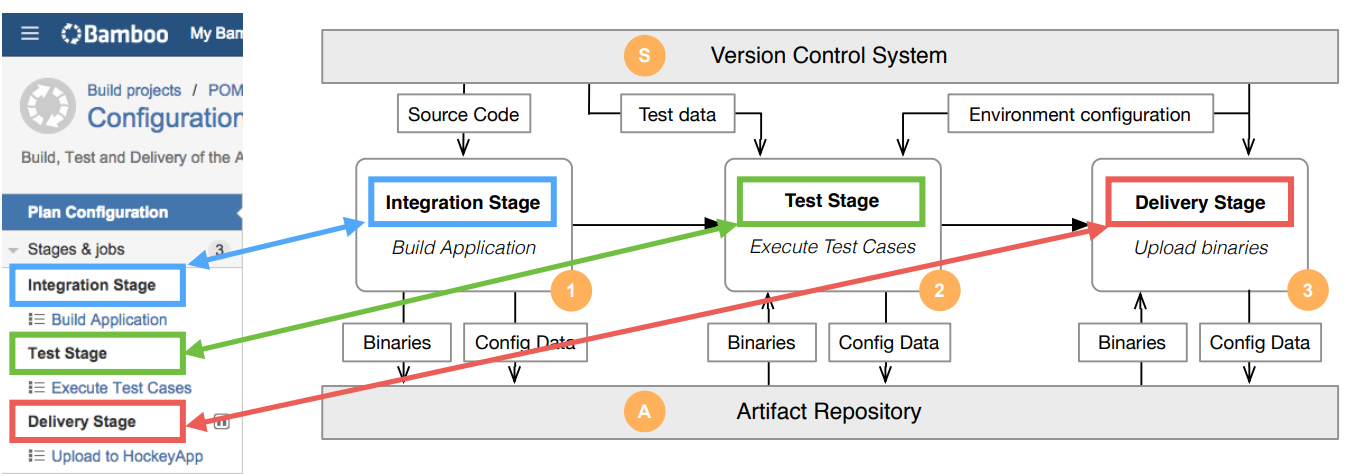
+ Reduced risk of a release failure

* Challenges

- Organizational: Varying interests in different departments of an organization

- Process: Traditional processes hinder continuous delivery

- Technical: Maintainability of the source code & tailorable delivery workflows for -heterogeneous project environments



**Build management: continuous integration server**

* Regularly checks for changes published to the version control server (pull) or is notified when changes are available (push)
* Builds the software
  + Compiles the source code
  + Execute test cases (regression testing)
  + Packages the software
  + Code signs the software
* Notifies developers about build status
* Automatically detects multiple branches and creates a build plan for each branch (Bamboo)

**Release management: continuous delivery server**

* Distributes software to customers and users (a.k.a. Enterprise AppStore)
* Obtains user feedback about the usage of the software
* Collects crash reports
* Stores feedback and crash reports in an issue tracker (e.g. JIRA)
* Release early and often to obtain feedback

**Management Issues: Continuous Delivery**

* How often do you deliver your application? (every change?)
* Which process changes do you have to implement to make customers aware of multiple releases during development?
* How do you handle customers that do not react to releases?
* How many releases can a customer / user handle?
* How do you handle feedback within the Sprint?

**Best practices for build and release management**

* Integrate your changes early and often
* Release early and often to obtain feedback
* Release internally first
* Test it thoroughly
* Then promote the release to your users
* React to crashes and user feedback

➡Improve continuously

**Conway’s Law**

“Any organization that designs a system (defined broadly) will inevitably produce a design whose structure is a copy of the organization's communication structure.”

[Jede Organisation, die ein System entwickelt (das allgemein definiert ist), wird unausweichlich ein Design produzieren, dessen Struktur eine Kopie der Kommunikationsstruktur der Organisation ist]

POM 20