POM 11

**Change management**

• Change management is the handling of change requests

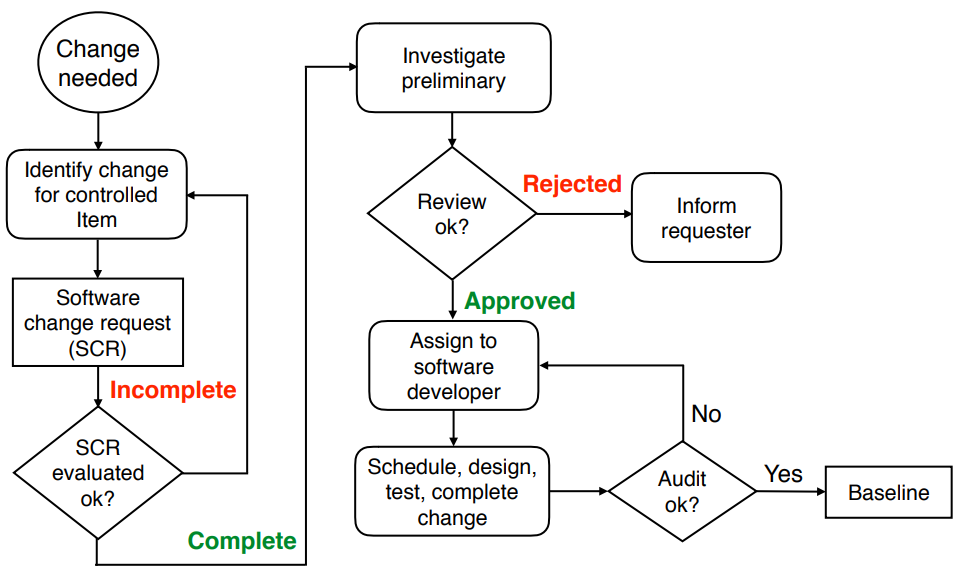
• The general change management process:

• The change is requested

• The change request is evaluated against requirements and project constraints and reviewed by the configuration control board

• Following these assessments, the change request is approved or rejected

• If it is approved, the change is assigned to a developer who will design, implement and test the change accordingly



**Change request**

Specifies the process for requesting a change to a configuration item and the documented informations:

* Nameand version of the configuration item where the need for change appears
* Originator’s (Gründer) name and address
* Date of request
* Indication of urgency (Dringlichkeit)
* Description of the requested change

**Change policies**

The purpose of a change policy is to guarantee that each promotion or release confirms to the accepted criteria.

Examples for change policies: “No developer is allowed to promote source code that was compiled with errors or warnings.”

**Change management activities and responsibilities**

1. Software configuration control: managing a change request

Define a change request form (formular)

Define management procedures for:

• Identification of the need for a change request

• Analysis and evaluation of a change request

• Approval (Genehmigung) or disapproval of a change request

• Implementation, verification and release of the change

1. Software configuration status accounting

Answers the following questions:

• What elements are tracked and reported for baselines and changes?

• What types of status accounting reports are generated? What is their frequency?

• How is information collected, stored and reported?

• How is access to the configuration management status data controlled?

1. Software configuration auditing (Überwachung)

• Includes the identification when, how and how often audits (Prüfungen) are necessary for the project

• An audit determines for each configuration item if it meets the required physical and functional characteristics

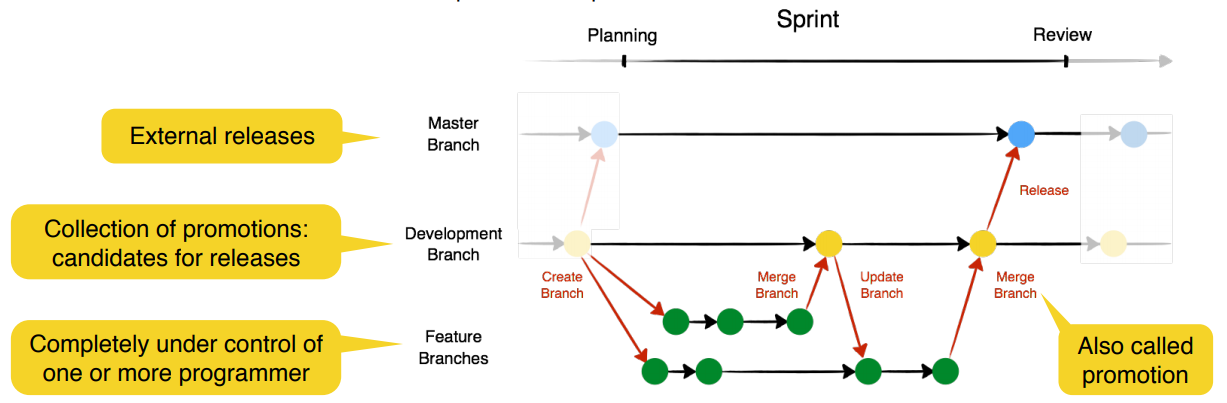
• Audits are conducted according to a well-defined process consisting of various auditor roles and responsibilities

➡Successful completion of an audit can be a prerequisite for the establishment of the product baseline

**Branch management**

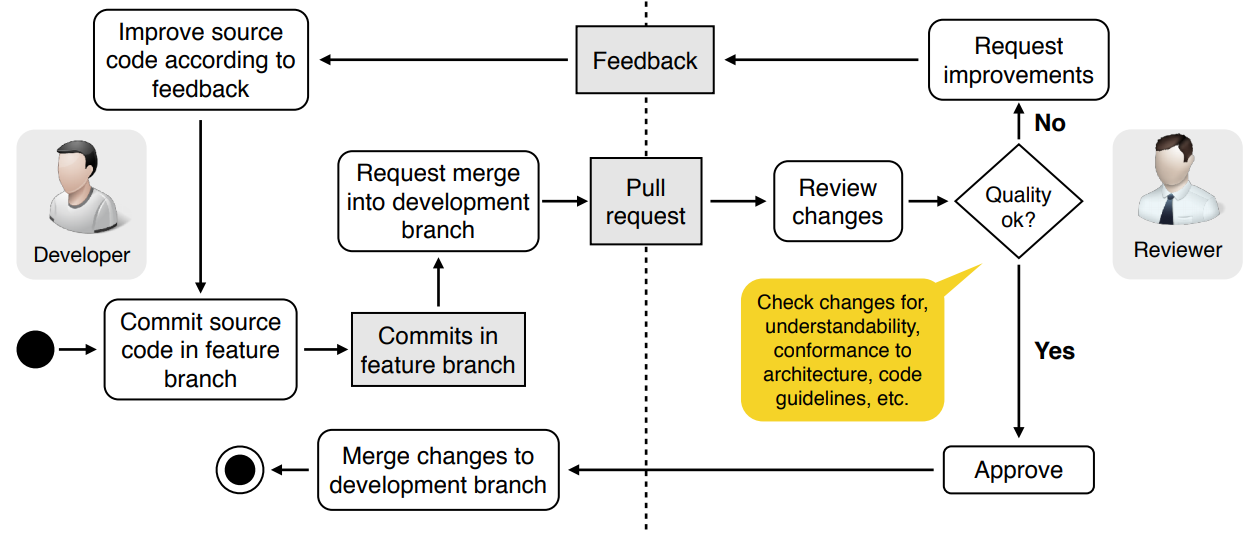
A branching model controls the concurrent development and defines rules when branches are created and merged (promoted)

* The **master branch** contains external release candidates (potential product increment)
* **Hotfix** **branches** include revisions (Verbesserung/Änderungen)
* **Release** **branches** contain external releases (to be used in production)
* The **development** **branch** contains internal release candidates
* **Feature** **branches** are used for the actual implementation work of new requirements but important: minimize feature branch lifetime (not more than “some” days)



**Merge management with pull requests and code reviews**

• Before the changes in a feature branch are merged (promoted) into the development branch, a reviewer looks at the changes and only approves the code if it is all right.



**The broken window theory**

Don't leave "broken windows" (bad designs, wrong decisions, or poor code) unrepaired.

One broken window, left unrepaired for any substantial length of time, instills in the developers of the code a sense of abandonment - a sense that the developers don’t care about the source code.

➡ Fix each broken window (i.e. bad source code) as soon as it is discovered

➡ Otherwise: clean, functional systems deteriorate (verschlechtern) quickly once windows start breaking