SOFTWARE ENGINEERING 2

KONFIGURATIONSMANAGEMENT

- Komplexe Systeme
  Unterschiedliche Komponenten in Beziehung zueinander

- Unterschiedliche Ausprägungen
  Komponenten in verschiedenen Versionen
  Verschiedenartige Komposition

- Mehrere Beteiligte

- Zeitliche Parallelität
  Mehrere Ausprägungen existieren zur selben Zeit

Aufgabe: Herstellung und Beibehaltung (=Sicherstellung) des ordnungsgemäßen Zustands

- Ursprünge des Konfigurationsmanagement in den 1950er Jahren

- Militärische Versuche in den USA mit Flugkörpern
  - Flugkörper in verschiedenen Varianten
  - Ergebnisse nicht verwertbar, da Flugkörper explodiert

- Einführung mittels Vorschriften

- Ziele
  - Geführtes und überwachtes Vorgehen
  - Reproduzierbarkeit

Konfigurationsmanagement (KM)
Koordinierte Tätigkeiten zur Leitung und Lenkung der Konfiguration.

ANMERKUNG Das Konfigurationsmanagement konzentriert sich üblicherweise auf technische und organisatorische Tätigkeiten, die die Lenkung eines Produkts und der dazugehörigen **Produktkonfigurationsangaben** in allen Phasen des Produktlebenszyklus einleiten und aufrechterhalten.

Konfiguration
miteinander verbundene funktionelle und physische Merkmale eines Produkts, wie sie in den Produktkonfigurationsangaben beschrieben sind.

Produktkonfigurationsangaben
Anforderungen an Entwicklung, Realisierung, Verifizierung, an Funktionstüchtigkeit und Unterstützung des Produkts.

Aufgaben des Konfigurationsmanagement

- Konfigurationsidentifizierung
- Konfigurationsüberwachung
- Konfigurationsbuchführung
- Konfigurationsaudit

- Auswahl der Konfigurationseinheiten (engl. configuration items)
  → Welche Artefakte sind relevant?

- Definition der Produktstruktur
  → Wie werden die Artefakte verwaltet?

- Dokumentation der Merkmale der Konfigurationseinheiten
  → Produktkonfigurationsangaben

- System zur Bezeichnung
  → Identifizierbarkeit (Seriennummern, Ablagesystem etc.)

- Dokumentation und Begründung von Änderungen
  → Was wird warum und wann geändert?

- Beurteilung von Änderungen
  → Welche Auswirkungen hat eine Änderung?

- Freigabe von Änderungen
  → Wer darf was ändern?

- Genehmigung von Ausnahmen

- Rückverfolgung von Änderungen
  → Wer hat was wann wo geändert?
  → Welchen Qualitätsstand hat eine Konfigurationseinheit?

- Nebenprodukte (Metainformationen)

- Beispiele
  - Änderungshistorie in Schriftstücken
  - Änderungskommentare in KM-Systemen

```
*
* $Author: root $
* $Date: 2012-10-16 13:24:43 +0200 (Di,
* $Revision: 18 $
* $URL$
*
```

Certified Tester
Foundation Level Syllabus

ISTQB International Software Testing Qualifications Board

Revision History

Version	Date	Remarks
ISTQB 2011	Effective 1-Apr-2011	Certified Tester Foundation Level Syllabus Maintenance Release – see Appendix E – Release Notes
ISTQB 2010	Effective 30-Mar-2010	Certified Tester Foundation Level Syllabus Maintenance Release – see Appendix E – Release Notes
ISTQB 2007	01-May-2007	Certified Tester Foundation Level Syllabus Maintenance Release
ISTQB 2005	01-July-2005	Certified Tester Foundation Level Syllabus
ASQF V2.2	July-2003	ASQF Syllabus Foundation Level Version 2.2 “Lehrplan Grundlagen des Software-testens“
ISEB V2.0	25-Feb-1999	ISEB Software Testing Foundation Syllabus V2.0 25 February 1999

- Funktionsbezogenes Konfigurationsaudit
  → Stimmen die Dokumente formal
  → Wurde richtig gearbeitet?

- Physisches Konfigurationsaudit
  → Stimmt das Produkt mit der Dokumentation überein?

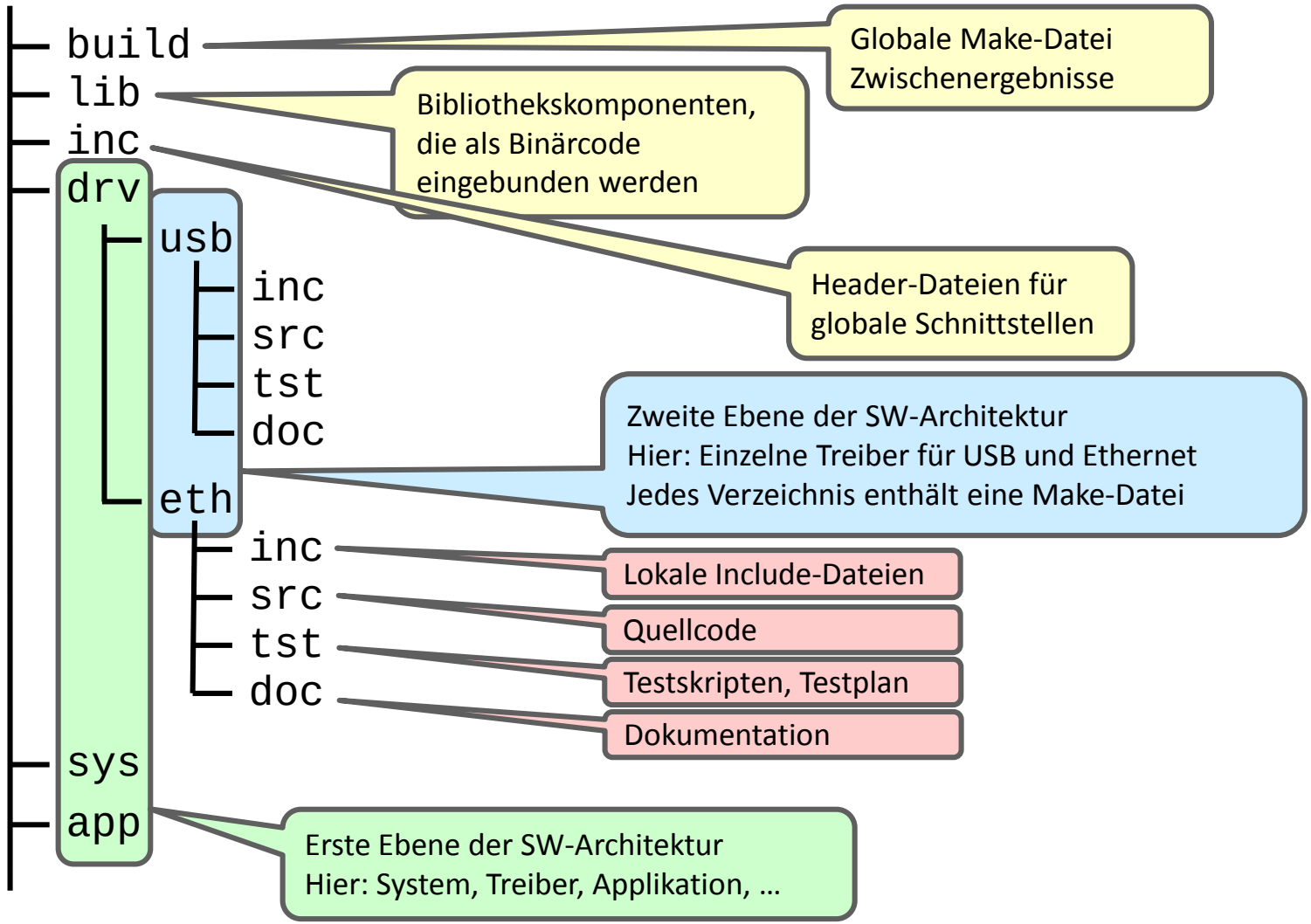
- Durchführung jeweils zu bestimmten Meilensteinen
  → Bezugskonfiguration (engl. Baseline)

- Das Vorgehen zum Konfigurationsmanagement wird im Konfigurationsmanagementplan (**KM-Plan**) beschrieben.

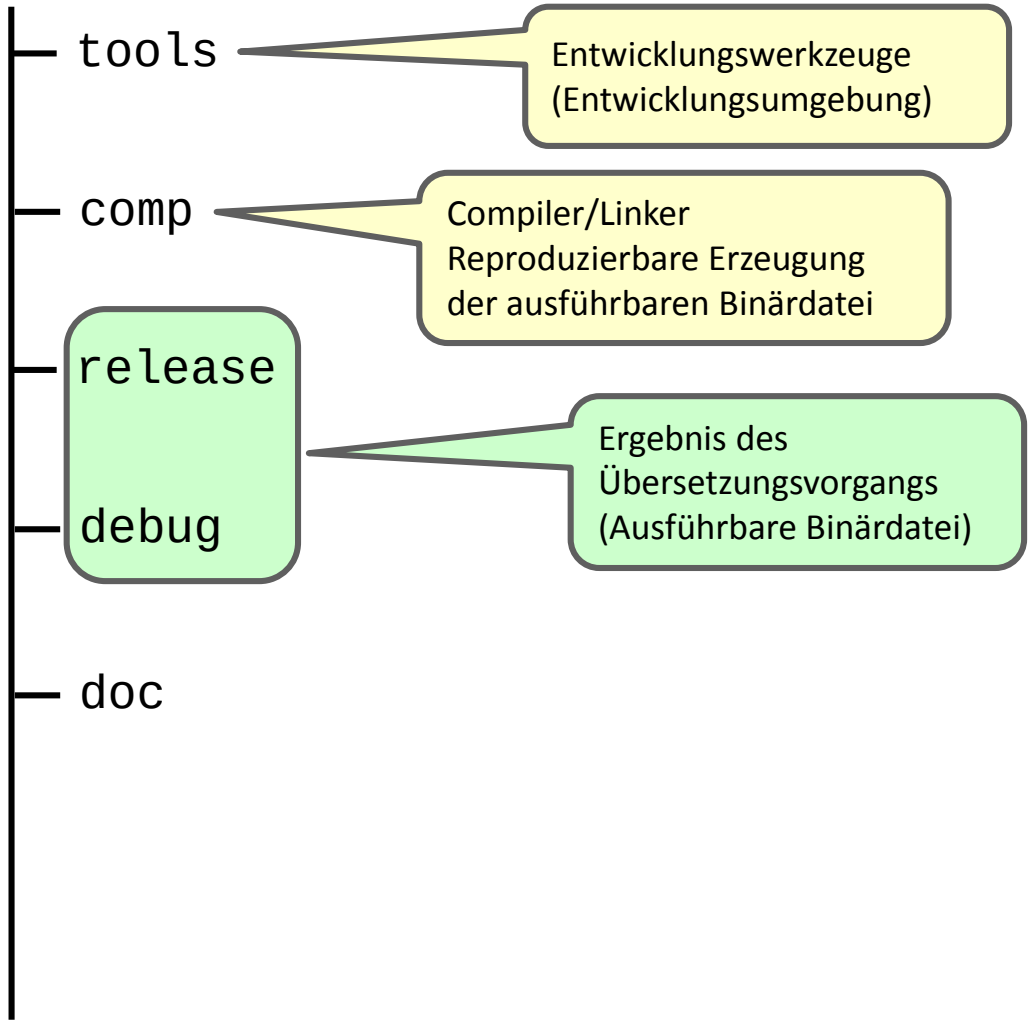
- Zuständigkeiten
  → Wer hat die Verantwortung?

- Änderungsprozess
  → Änderungsausschuss (Change Control Board)
- Bezeichnung von Artefakten
- Zeitliche Planung

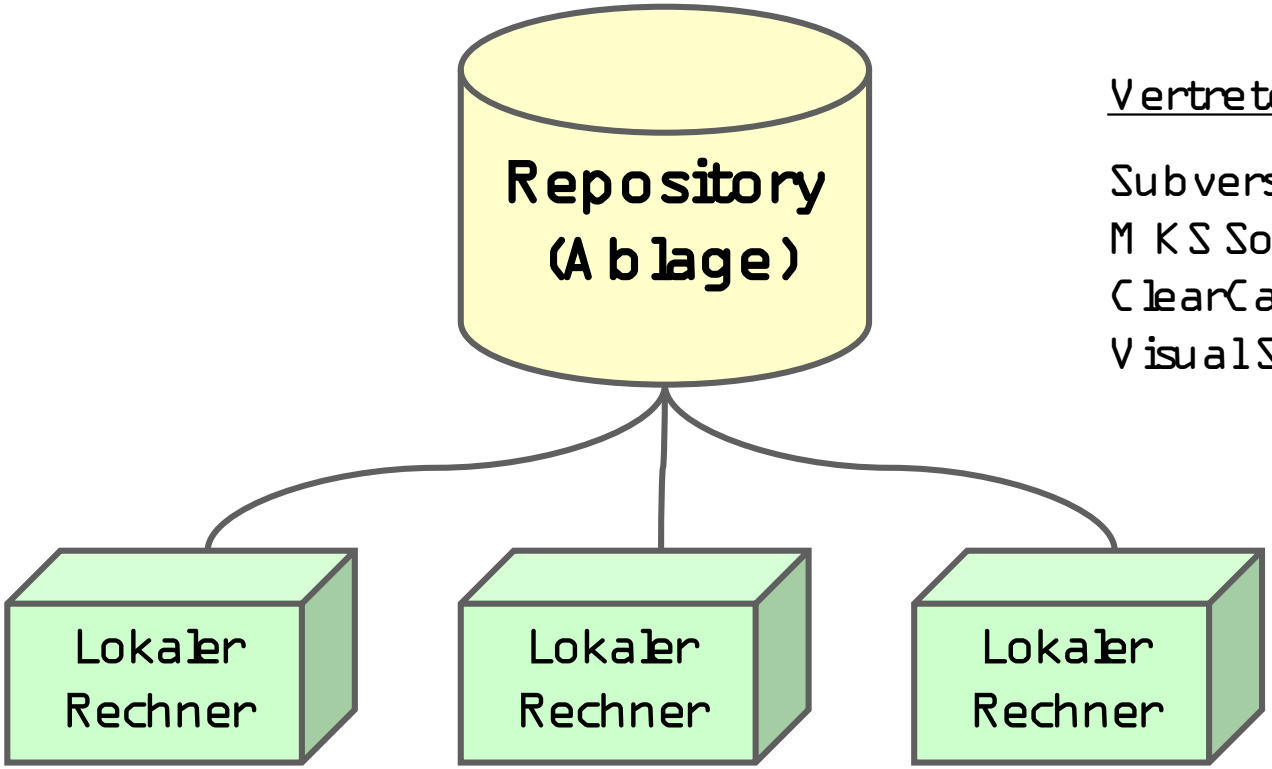
Ablagestruktur von SW-Projekten (1/2)



Ablagestruktur von SW-Projekten (2/2)



- Protokollieren der Änderungen
  (Konfigurationsbuchführung)
- Reproduzierbarkeit von alten Ständen einzelner Dateien
  (nicht Gesamtstand)
- Archivierung
- Unterstützung der verteilten Entwicklung:
  - Koordinierung des Zugriffs auf gemeinsam genutzte Dateien
  - Einrichten von Entwicklungspfaden (Branches) für parallele Entwicklung



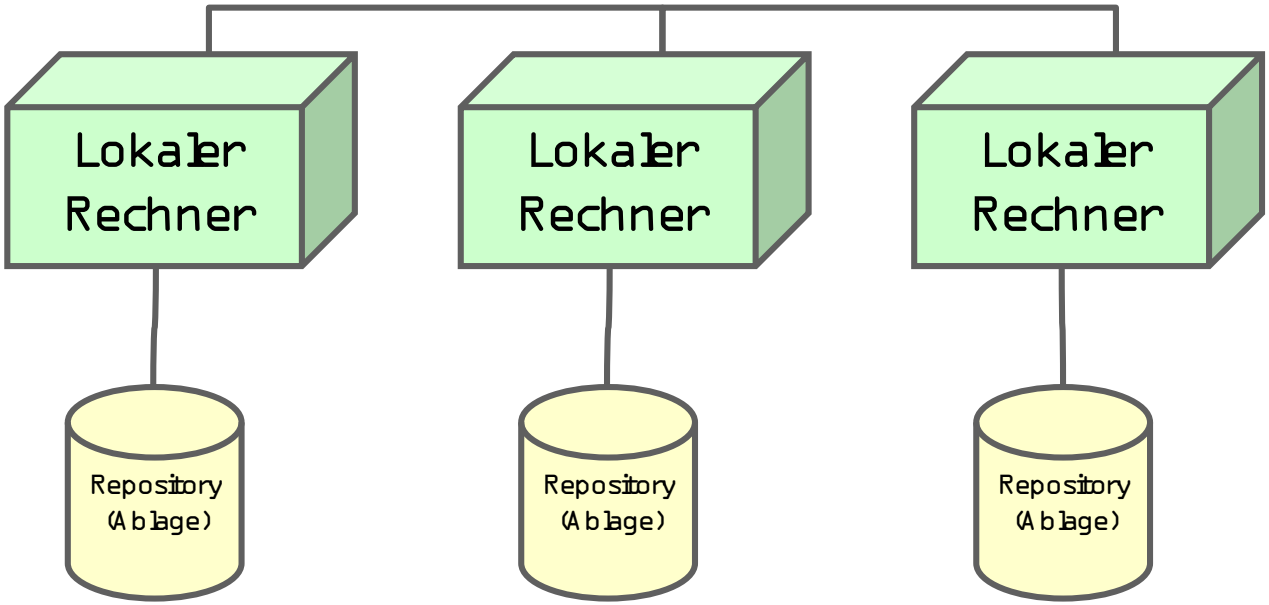
Vertreter:

Subversion
MKS Source Integrity
ClearCase
VisualSource Safe

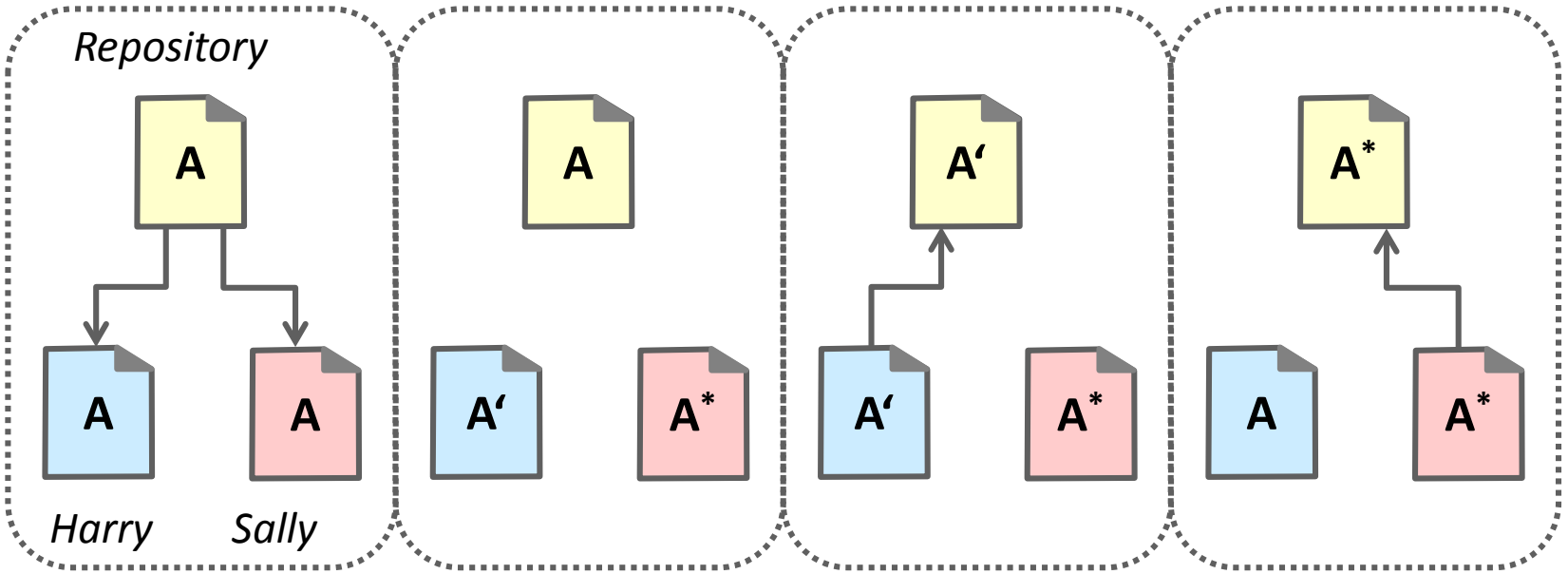
Vertreter:

GIT

Bazaar



Verteilter Dateizugriff – Problemstellung



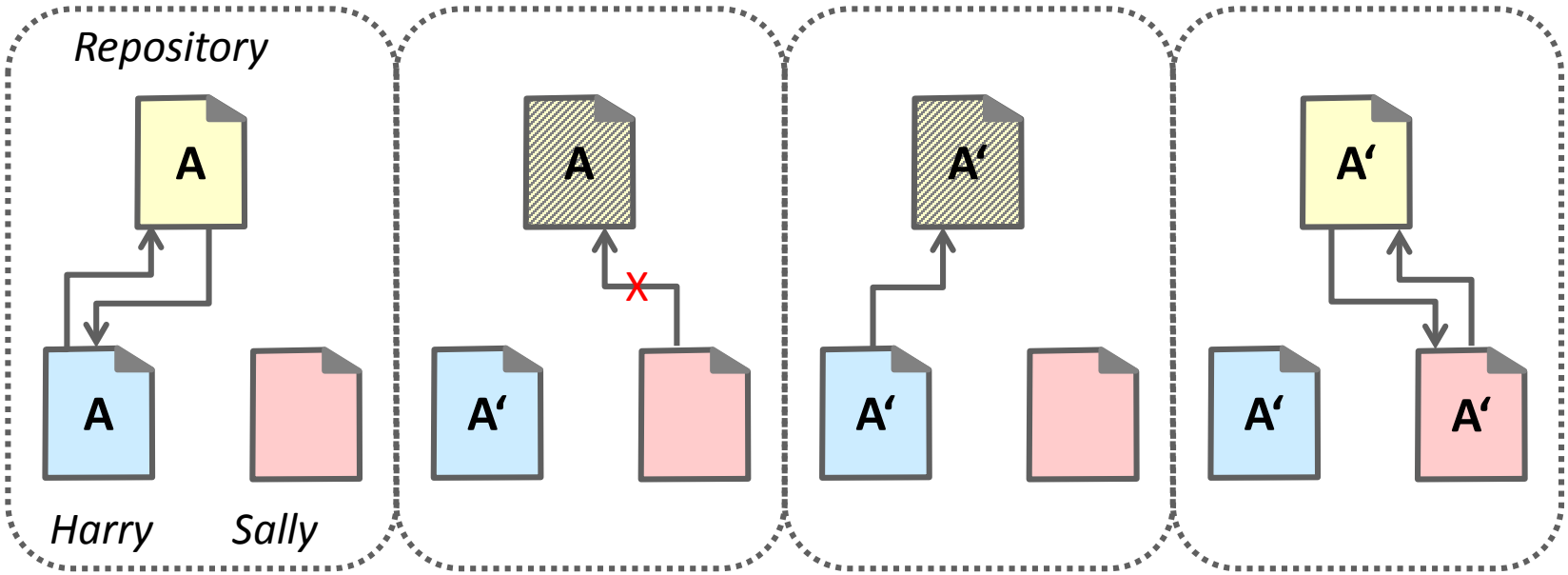
Zwei Benutzer lesen ein Dokument aus dem Repository
→ lokale Kopien

Beide nehmen unabhängig voneinander lokal Änderungen vor.

Ein Benutzer schreibt seine Änderungen in das Repository.

Der zweite Benutzer schreibt seine Änderungen in das Repository und überschreibt die Änderungen des ersten.

Lösung 1: Lock – Modify – Unlock



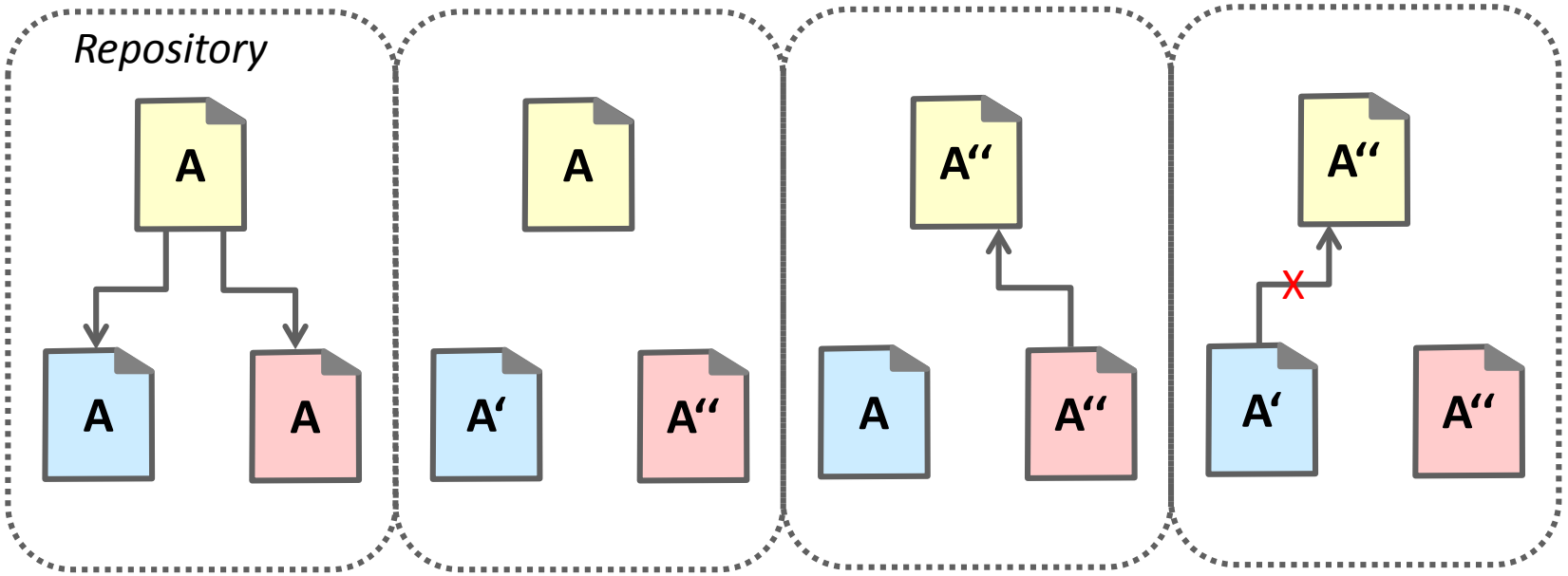
Benutzer 1 sperrt das Dokument (**Lock**) und liest es dann (lokale Kopie).

Während Benutzer 1 das Dokument bearbeitet (**Modify**), kann kein anderer Benutzer das Dokument sperren.

Benutzer 1 schreibt seine Änderungen ins Repository zurück und gibt das Dokument wieder frei (**Unlock**).

Benutzer 2 kann jetzt das Dokument sperren und lokal bearbeiten.

Lösung 2: Copy – Modify – Merge (1/2)



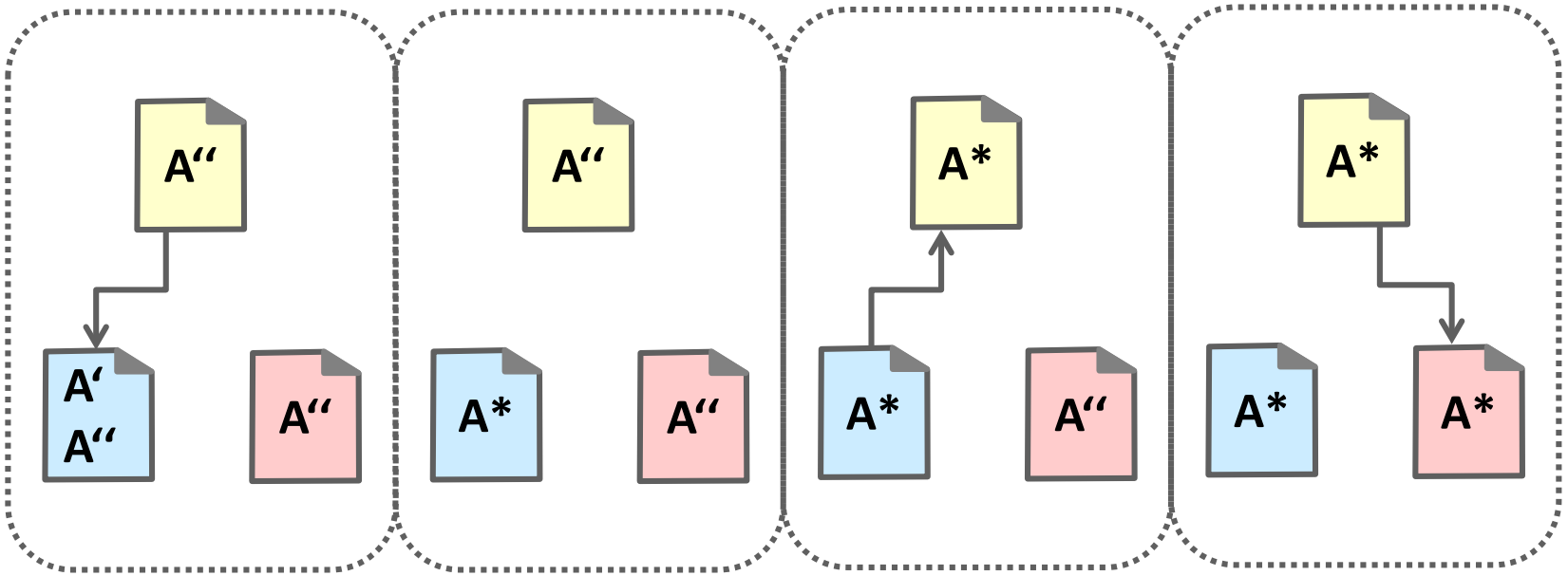
Zwei Benutzer lesen ein Dokument aus dem Repository (**Copy**)

Beide nehmen unabhängig voneinander lokal Änderungen vor (**Modify**).

Benutzer 2 schreibt seine Änderungen in das Repository.

Beim Versuch, seine Änderungen ins Repository zu schreiben, bekommt Benutzer 1 eine „Out-of-date“ Fehlermeldung.

Lösung 2: Copy – Modify – Merge (2/2)



Benutzer 1 vergleicht seine lokale Version mit der im Repository.

Er führt seine Änderungen mit denen im Repository zusammen (**Merge**). Das Ergebnis liegt zunächst lokal vor.

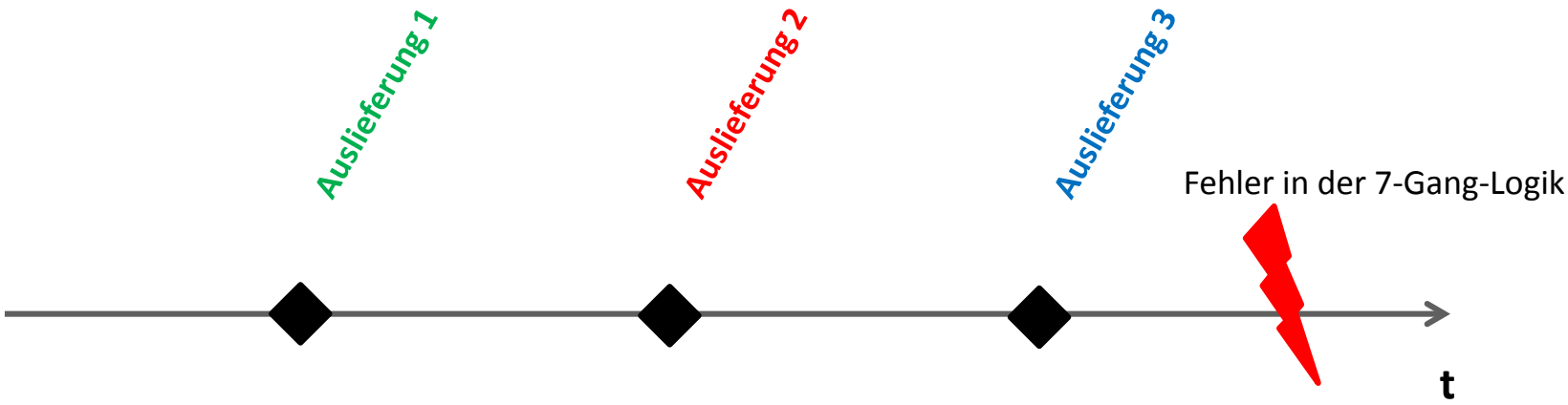
Benutzer 1 schreibt die zusammengeführte Version ins Repository.

Benutzer 2 liest die Änderungen aus dem Repository lokal ein.

Lock – Modify – Unlock	**Copy – Modify – Merge**
Serielles Arbeiten	Paralleles Arbeiten
Keine Konflikte an einem Dokument	Konflikte müssen aufgelöst werden
Benutzer haben gleichen Stand	Unterschiedliche Stände
Abhängigkeiten in Schnittstellenänderungen werden u.U. erst spät erkannt	Abhängigkeiten werden bei Änderung erkannt
Für jeden Dateityp möglich	Nur falls Zusammenführen möglich ist

- Checkout
  Kopieren einer Datei aus dem Repository auf das lokale Dateisystem (bei manchen System mit Lock verbunden)
- Commit/Checkin
  Schreiben der lokalen Kopie in das Repository
  (ggf. mit Rückgabe des Lock)
- Rebase/Resynchronize/Update
  Aktualisieren der lokalen Dateien auf einen bestimmten Stand (i.a. aktueller Stand) aus dem Repository
- Head
  Aktuelle Arbeitsversion im Repository
- Revision/Version
  Bestimmter Versionsstand einer Datei
- Trunk/Master
  Hauptentwicklungspfad

Notwendigkeit von Baselines



Eco-Mode unvollständig; 7-Gang-Schaltung

Eco-Mode unvollständig; 9-Gang-Schaltung

Eco-Mode vollständig; 9-Gang-Schaltung

- Rücksetzen auf alten Projektstand um dort (lokale) Änderungen vorzunehmen
- Vergleich zwischen zwei Ständen
- Auswirkungsanalyse: Welche Stände sind von Fehlern betroffen?
- Eindeutiger Bezugspunkt für
  - Funktionalität
  - Dokumentation
  - Qualitätssicherung
- Artefakte liegen zu unterschiedlichen Zeitpunkten vor, gehören aber inhaltlich zusammen

Configuration baseline

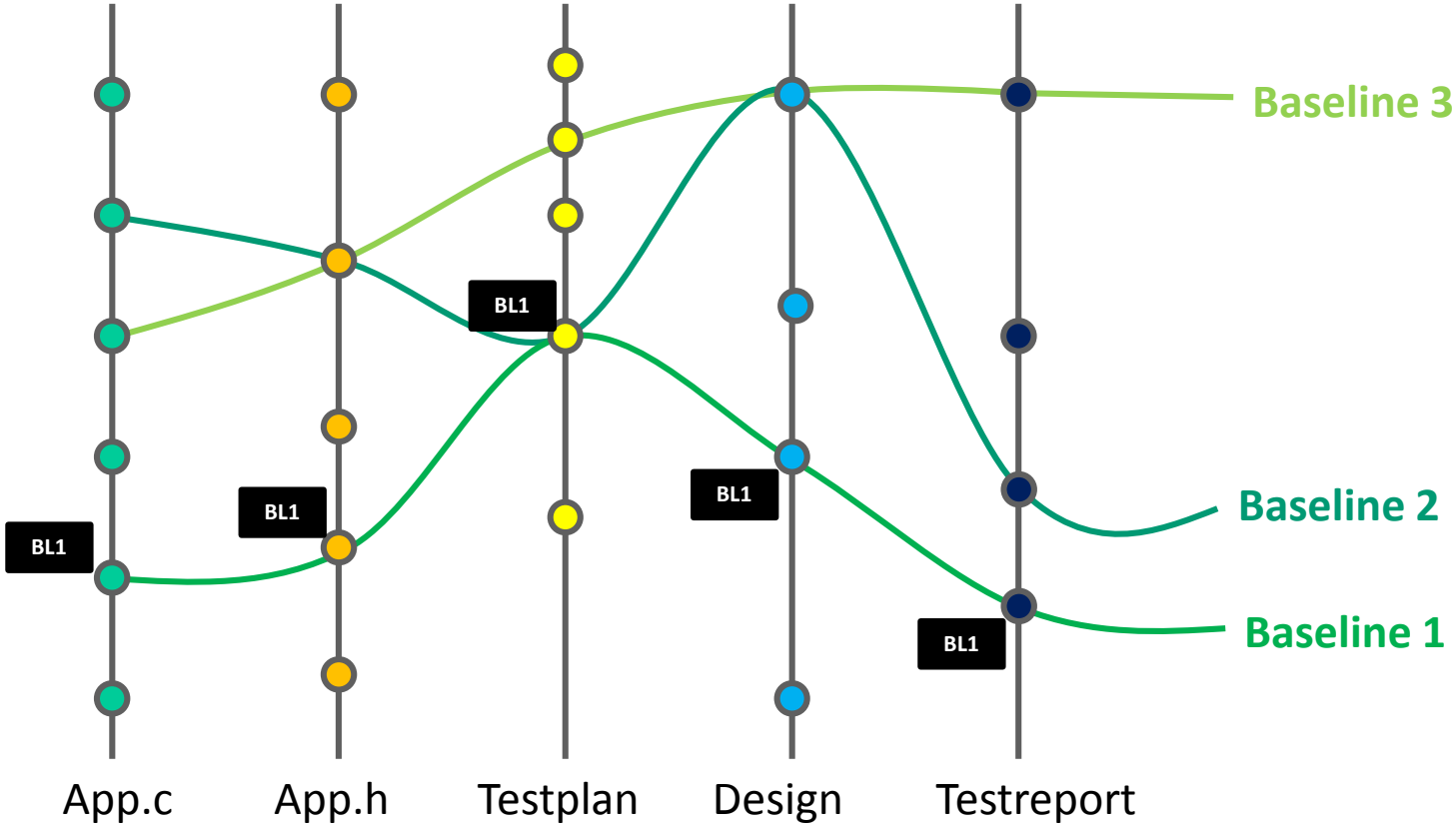
approved product configuration information that establishes the characteristics of a product at a point in time that serves as reference for activities throughout the life cycle of the product.

Referenzkonfiguration

Eine Bezugs- oder Referenzkonfiguration ist ein zu einem bestimmten Zeitpunkt **ausgewähltes** und **freigegebenes** Zwischenergebnis, das zu einer Konfiguration zusammengefasst wird, um sich später darauf **beziehen** zu können.

- Eine Baseline wird zu einem bestimmten Zeitpunkt erstellt, aber
  - die zugehörigen Dokumente können zu einem anderen Zeitpunkt erstellt worden sein.
  - die zugehörigen Dokumente können aktuell in neueren Versionen vorliegen.
- Eine Baseline umfasst alle Konfigurationseinheiten
  - Nicht nur Quellcode
  - Dokumentation
  - Testprotokolle
- Eine Baseline hat einen definierten Qualitätsstand
  - „Freigegeben“ → Formale Kriterien erfüllt
  - Typische Kriterien:
    - Testabdeckung
    - Funktionalität
- Eine Baseline hat einen eindeutigen Namen
  - Oft: Alphanumerische Bezeichnung mit Qualitätsstand

- KM-Systeme unterstützen die Erstellung von Baselines
  - Manchmal: Label-Mechanismus

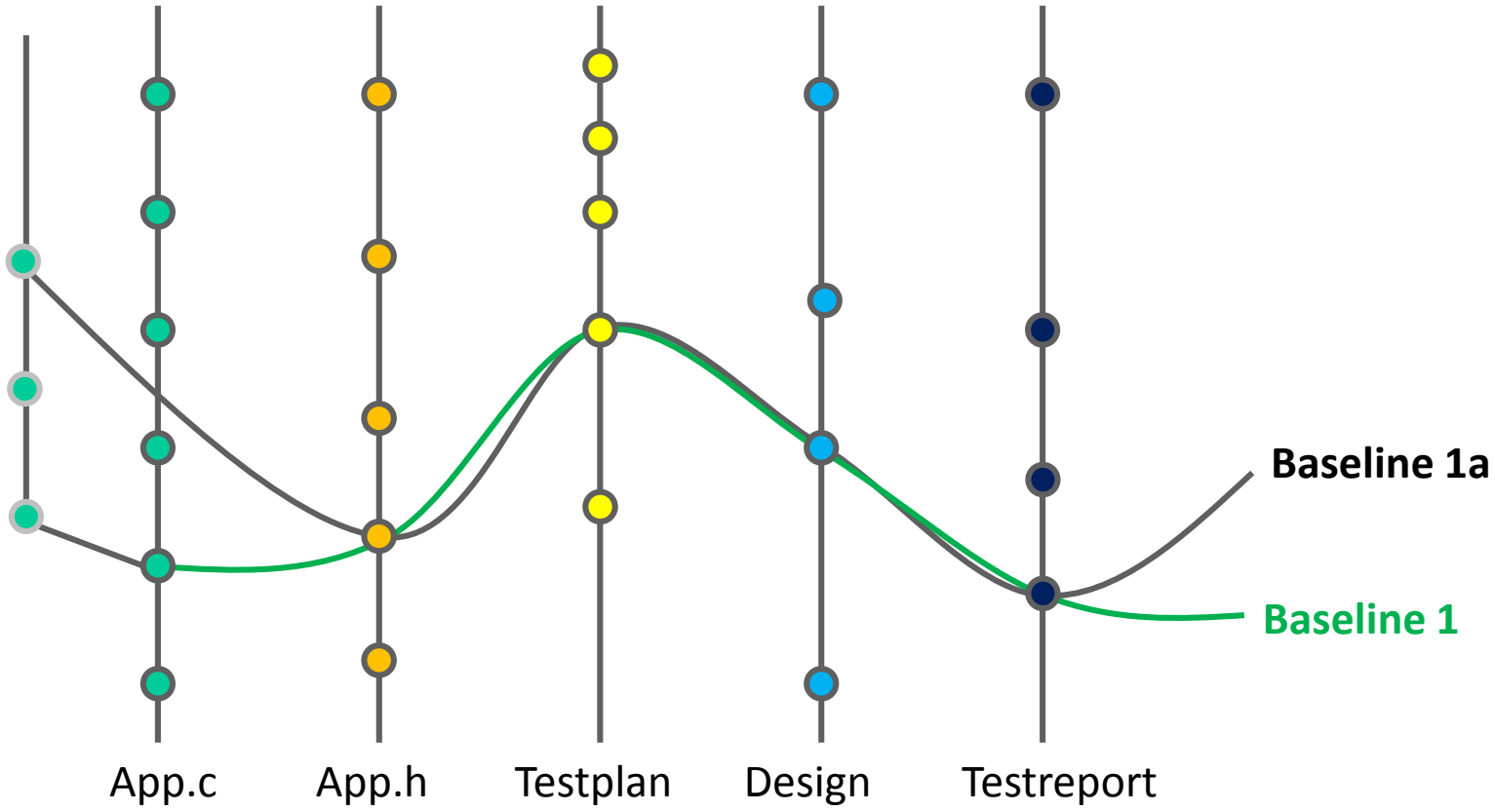


- Nicht alle Artefakte liegen im KM-Werkzeug
  - Aufkleber (Labels) mit Baseline-Bezeichnung bei Gegenständen
  - Liste von Dokumenten mit eindeutiger Versionsbezeichnung

- Umsetzung mit Versionswerkzeug Subversion
  - Snapshot → „Tags“
  - Eigener Zweig im Versionsbaum
  - HEAD zu einem Zeitpunkt! → Keine echte Konfiguration

- Allgemeine Umsetzung
  - Baseline-Beschreibung in einer Datei
  - Aufzählung der Artefakte und der jeweiligen Versionen
  - Versionierung der Baseline-Beschreibung

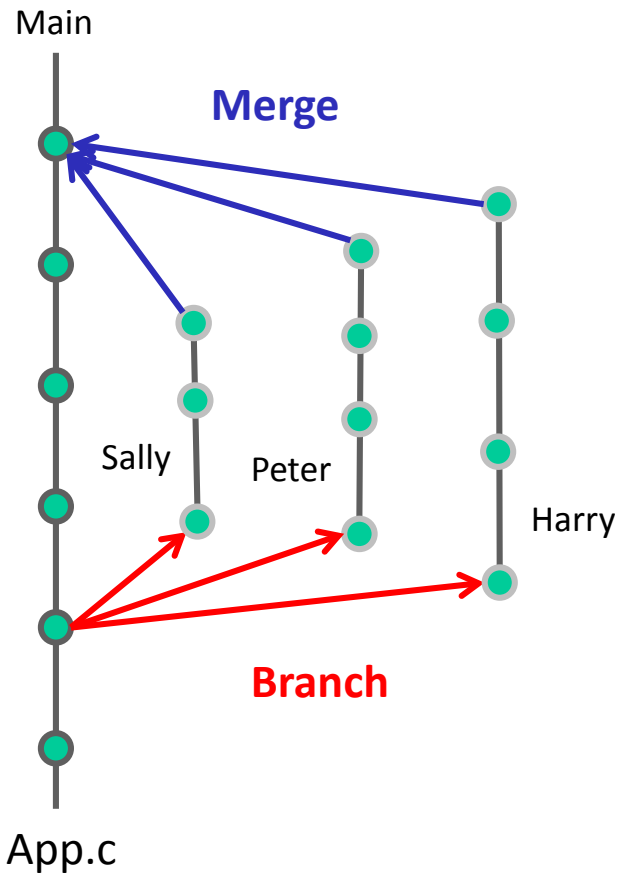
- Beispiel: Bugfix auf Basis alter Baseline



- Entwicklungspfade (Branches)
- Aufzweigen der Entwicklung auf Referenzstand
- Unabhängige Entwicklung

- Möglichkeit 1: „Toter Pfad“
  - „Rückwärtspflege“ von Vorgängerversionen
  - Änderungen werden im Hauptpfad unabhängig vorgenommen
  - Typischer Anwendungsfall: Bugfix

- Möglichkeit 2: Parallele Entwicklung
  - Eigene Pfade für parallele Entwicklung
  - Zusammenführen bei der Integration
  - Keine weitere Pflege nach Integration



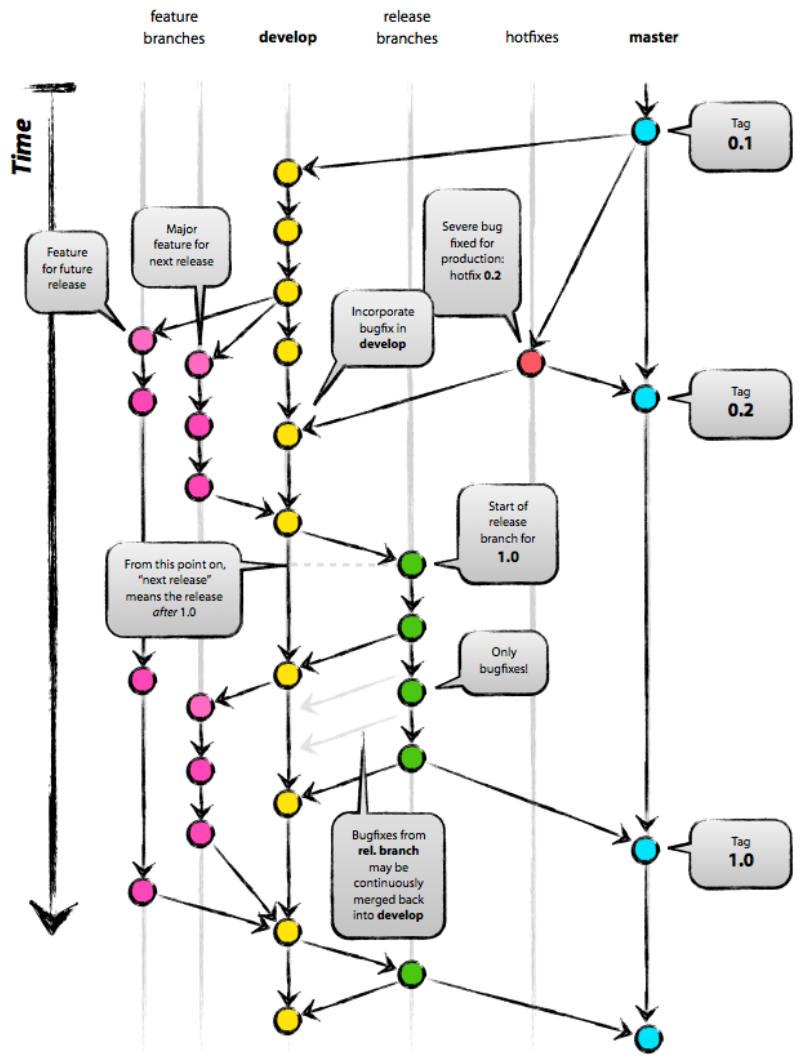
- Verzweigen auf Referenzstand
- Zusammenführen zu nächstem Referenzstand
- Oft auch Zwischenintegration
- Integration u.U. sukzessive

- Subversion unterstützt Branches

- Konventionen
  - Verzeichnis „Trunk“ → Hauptentwicklungspfad/Integration
  - Verzeichnis „Tags“ → Referenzkonfigurationen
  - Verzeichnis „Branches“ → Entwicklungspfade

- Entwicklungspfad wird angelegt durch „Branch“ Befehl
- Ursprungsversion ist bekannt

Ansatz zur Bildung von Pfaden



Quelle: nvie.com

- Master:
  Nur stabile und freigegebene Software
- Develop:
  Integration von Features zur Gesamtsoftware
- Feature Branches:
  Entwicklung einzelner Features unabhängig voneinander
- Release Branches:
  Freigabeprozess für stabile Stände