Software Engineering

Informatik II.

10. Software-Entwicklung

– Konfigurations-Management –

Dipl.-Inform. Hartmut Petters

Vorwort – was ich noch zu sagen hätte ...

Basis dieser Vorlesung sind vor allem die folgenden Ausarbeitungen

- Vorlesungsskript „Software Engineering“
  von Prof. Dr. Martin Glinz Universität Zürich
  http://www.ifi.unizh.ch/groups/req/courses/ses/
- Vorlesungsskript „Informatik II – Software Engineering“
  von Frau Prof. Dr. Kühn FH Karlsruhe FB W
  http://www.home.fh-karlsruhe.de/~kuin0001/inhalt.htm
- Das Buch „Software Engineering“ 6. Auflage/2001
  von Prof. Ian Sommerville University of Lancaster (UK)
  Addison Wesley ISBN 3-8273-7001-9

Konkret entnommene Beiträge sind i.d.R. mit einem Quellen-Verweis gekennzeichnet – sollte dieser fehlen bitte ich um Nachsicht.

Den „roten Faden“ durch die Vorlesung habe ich dem Skript der Vorlesung von Prof. Dr. Martin Glinz entnommen und um eigene Beiträge erweitert bzw. aus den beiden anderen Quellen ergänzt.

Für die Möglichkeit der Verwendung der wesentlichen Inhalte möchte ich mich an dieser Stelle bei den Autoren herzlich bedanken.

Problemdarstellung

Ändern Sie doch mal eben schnell ...

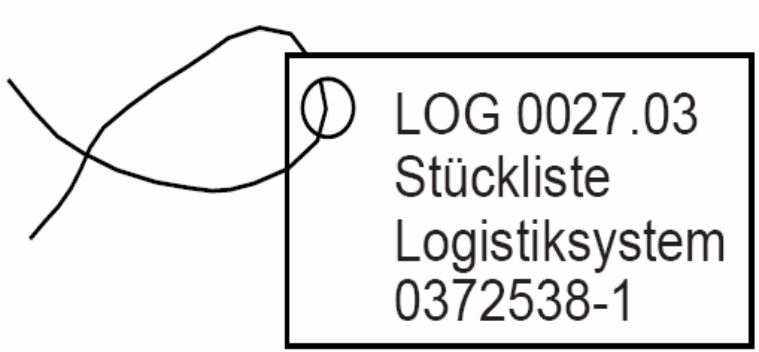
- Software ist *scheinbar leicht änderbar*
- Während der Entwicklung entstehen *viele Artefakte* in vielen *Versionen*
- Typische Probleme
  - Codierung anhand der *falschen Version* des Entwurfs
  - *Inkonsistente Unterlagen* bzgl.
    - Anforderungen, Detail-Konzepte
    - Datenmodell, Schnittstellen
    - ... etc.
  - Paralleles, *unkoordiniertes Ändern* durch mehrere Personen (Schreiber – Leser-Problem!)
  - *Undokumetierte Schnellreparaturen* an in Betrieb befindlicher Software
- ↳ Hohe Kosten
- Probleme wachsen *überproportional* mit der Komplexität
- Das *Gegenmittel* heißt Software-Konfigurations-Management

Definitionen

- Software-Konfigurations-Management
  (software configuration management)
  Die Gesamtheit aller Verfahren + Maßnahmen zur eindeutigen *Kennzeichnung* der Konfiguration eines Software-Systems mit dem Zweck, den *Aufbau* und alle *Änderungen* dieser Konfiguration systematisch zu *überwachen*, die *Konsistenz* des Software-Systems *sicherzustellen* und die Möglichkeit der *Rückverfolgung* anzubieten.
- Software-Konfiguration
  Eine Menge zusammenpassender Software-Einheiten
- Software-Einheit
  (software configuration item)
  Der *kleinste*, im Rahmen des Konfigurations-Managements als *atomar* behandelte *Baustein* einer Konfiguration.
  - Als Ganzes *registriert, freigegeben* oder *geändert*
  - Zum Beispiel Programm-Module und Dokumente

Kennzeichnung von Software-Einheiten

- Software-Einheiten haben eine eindeutige Kennzeichnung
- Besteht aus einem *Namen* und einer *Versionsnummer*
- Kann weitere Informationen enthalten, zum Beispiel Name des Systems oder Teilsystems
- Die Identität einer Software-Einheit ist feststellbar, z.B. mit *Prüfsummen*



Registrierung + Verwaltung

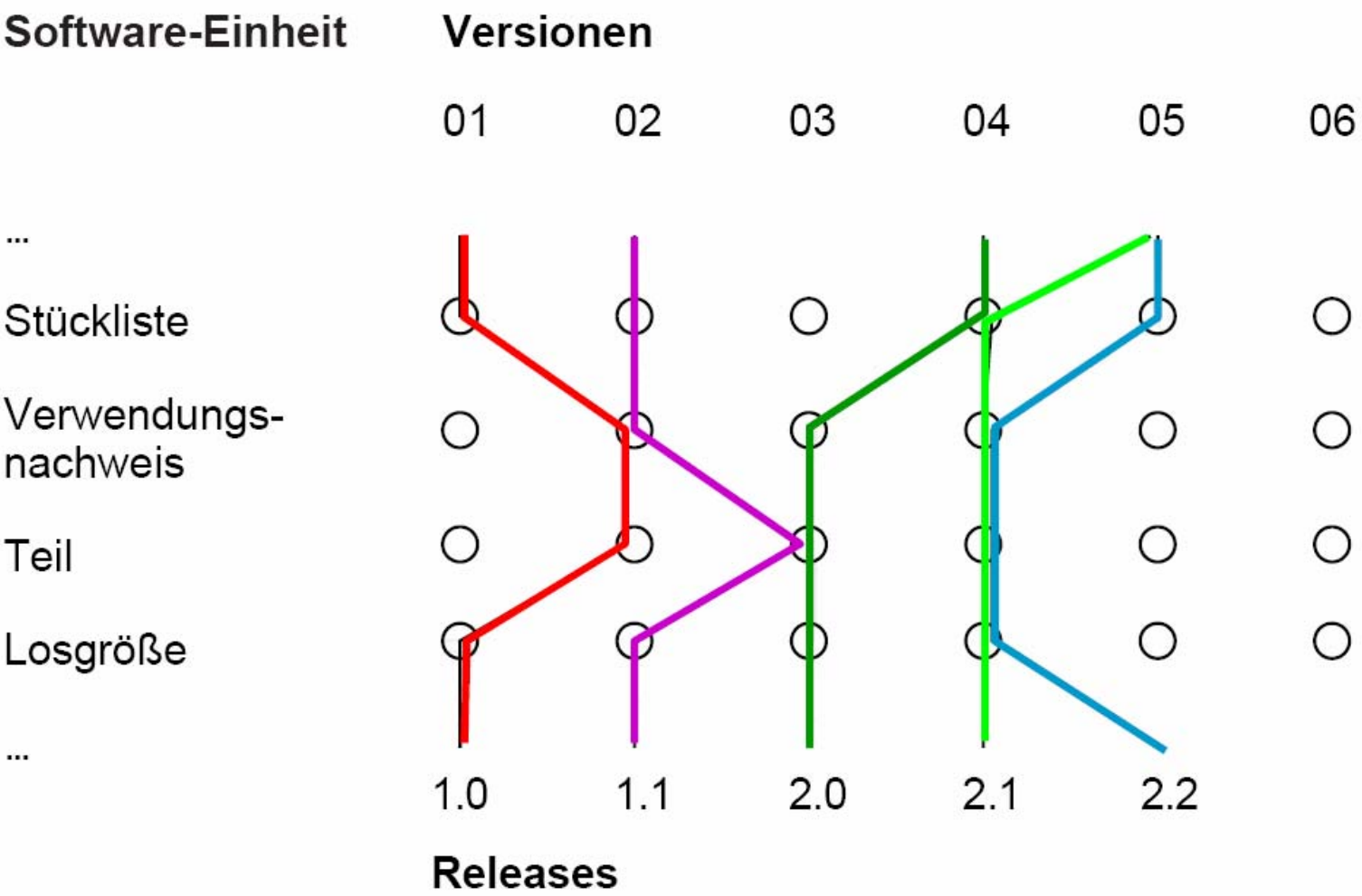
- Registrierung und Verwaltung der Software-Einheiten durch Software-Bibliothekar
- Pro Einheit mehrere Versionen möglich
- Im einfachsten Fall – aufsteigende Versionsnummern
- Im allgemeinen Fall – Revisionen (aufsteigend) und Varianten (parallel)
- Aktionen zur Verwaltung
  - „Check-In" einer Einheit – Übernahme in die Verwaltung („Electronic Vault")
  - „Check-Out" einer Einheit – Bezug zur Änderung
  - „Code-Freeze" – Festschreibung einer konsistenten Version

Nummer	Name	Typ	Ver	Prüfsumme	Status
...					
LOG 0021	Materialwesen	EntwDok	02	0873451-2	freigegeben
LOG 0027	Stückliste	Prog	03	0372538-1	freigegeben
LOG 0028	Verwendungsnachweis	Prog	02	0576927-6	in Prüfung
...					

Konfiguration + Release

- Release
  Eine *konsistente* Menge von Software-Einheiten, die *gemeinsam* zur Benutzung *freigegeben* werden
- Dient vor allem
  - Zur *Auslieferung* von Software-Produkten an *Kunden*
  - Zur periodischen *Lieferung* von *Nachträgen* + *Verbesserungen*
- Beantwortet u.a. folgende *Fragen*
  - Welche Software-Einheiten gehören zu einer Konfiguration?
  - Wie hängen die Einheiten voneinander ab?
  - Wie wird ein auslieferbares System generiert?
  - Welche Einheiten gehörten zur Version, die am12. Oktober 1999 ausgeliefert wurde?
  - Welche Änderungen wurden bei einer Einheit aus dieser Version bis jetzt gemacht? Änderungsanträge + Änderungsfreigaben.
  - ... u.v.m.

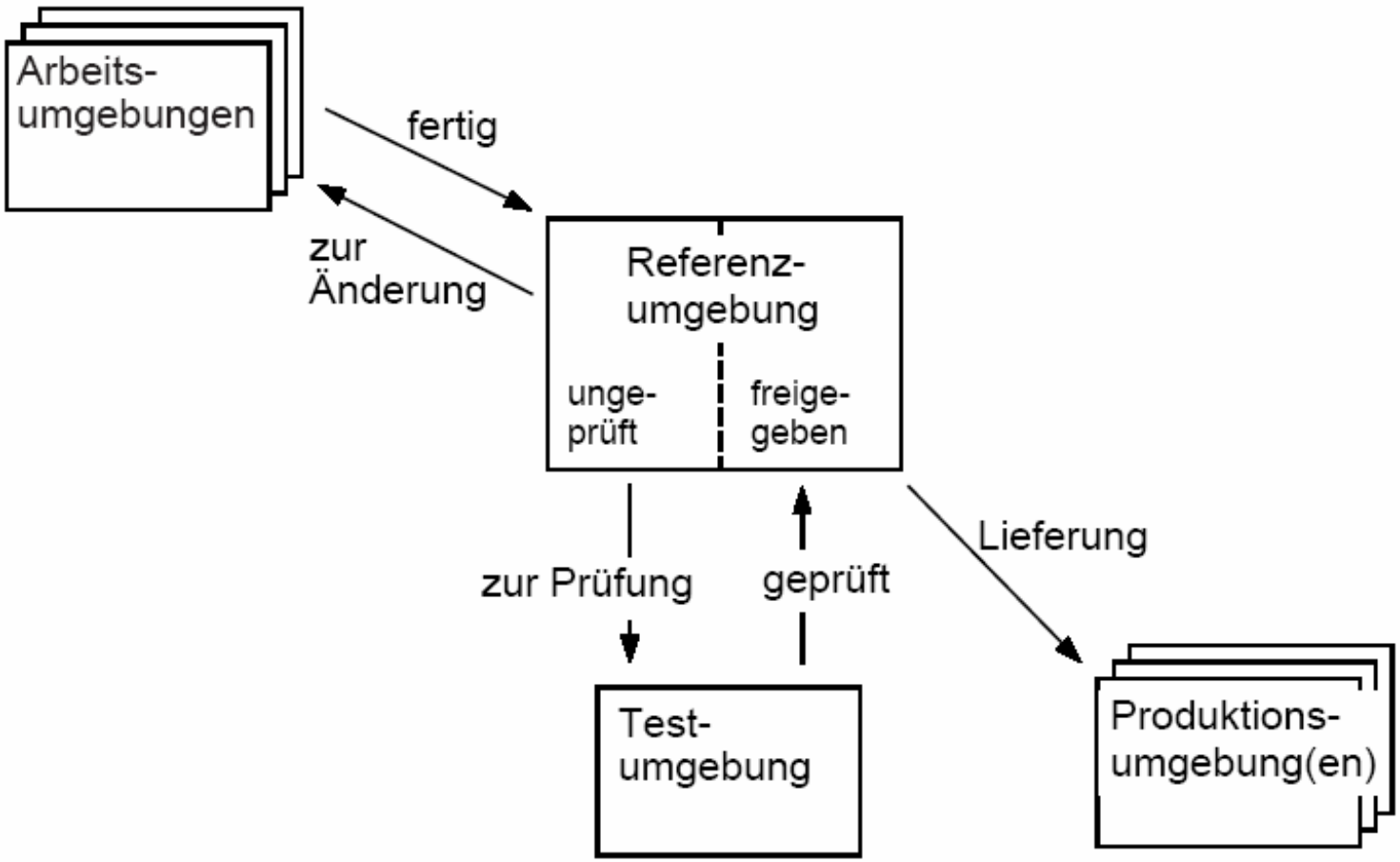
Eine Folge von Releases



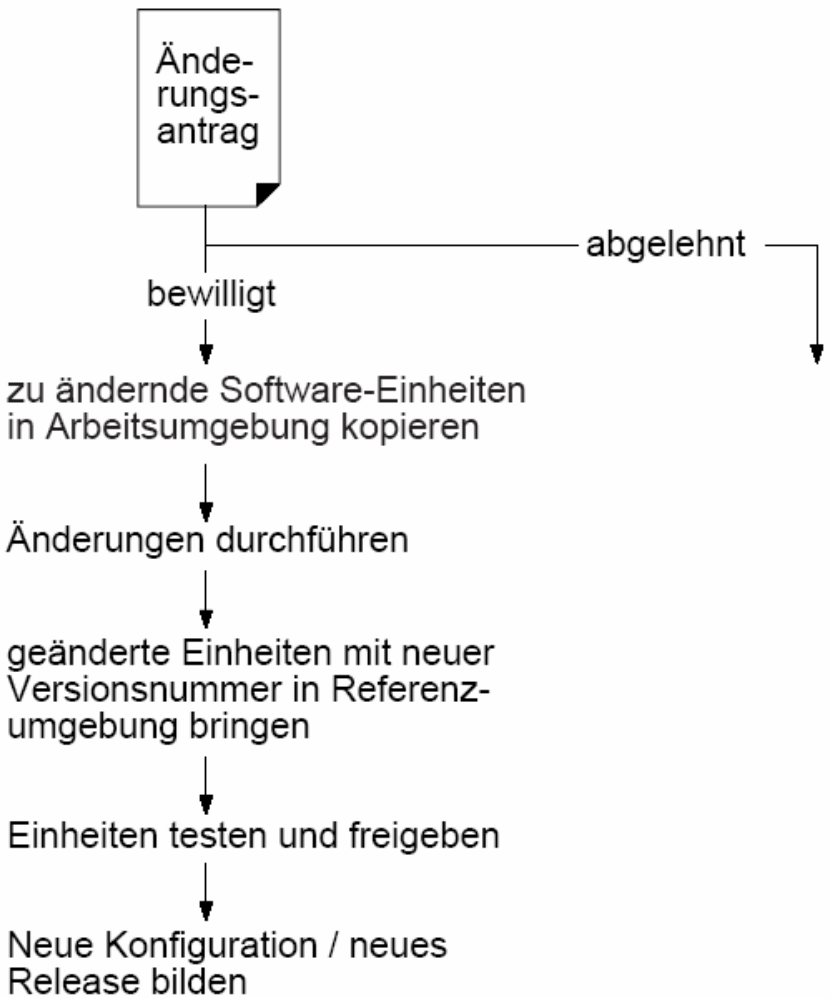
Änderungs- und Freigabewesen

- Getrennte Umgebungen für
  - Entwicklung (*Arbeitsumgebung*)
  - Verwaltung (*Referenzumgebung*)
  - Test (*Testumgebung*)
  - Operativen Einsatz (*Produktionsumgebung*)
- Freie Änderungen nur in der *Arbeitsumgebung*
- *Strikt festgelegte Änderungsprozesse* für Software-Einheiten in der *Referenzumgebung*
- *Änderungen* in der *Produktionsumgebung* sind verboten
- Änderungsstand einer Software-Einheit jederzeit *nachweisbar*
- Definierter *Prozess* für den Ablauf von Änderungen, deren Freigabe und Übernahme in Referenzumgebung

Konfigurations-Management Umgebungen



Prozess für den Ablauf einer Änderung



Problem- / Fehlererfassung + Dokumentation

- Systematische Behandlung von Kundenproblemen
- Grundlage: organisiertes Problem-Meldewesen
- Problem-/Fehlermelde-Formular
- Geordneter Bearbeitungsablauf (Problembearbeitungsprozess)
  - Registrierung eingegangener Meldungen
  - Analyse der Meldungen
  - Klassifizierung der Meldung (A / B / C)
  - Klasse A + B: „work-around" möglich?
  - Vorläufige Antwort (innerhalb von 24 Stunden)
  - Problembehebung + Dokumentation
  - Abschließende Antwort
  - Abschluss + Ablage der Problemmeldung
  - Auslieferung eines „Patches" oder eines neuen „Releases"

Problem- / Fehlermeldung – 1te

Problemmeldung	**Nr.**

Verfasser

Name _____ Datum _____

Firma _____ Telefon / Fax / E-mail _____

Adresse _____

Betrifft

- ❑ Produkt _____
- ❑ Leistung _____
- ❑ anderes _____

Verwendete Hardware _____

Betriebssystem _____

Problem ist

	ja	nein
reproduzierbar	❑	❑
umgehbar	❑	❑

Problem betrifft

- ❑ Programme
- ❑ Unterlagen
- ❑ Leistungen

Antwort erwartet bis

_____

Problem- / Fehlermeldung – 2te

Problembeschreibung ❑ Problembeschreibung in Beilage

Zu treffende Maßnahmen

Klassifizierung der Maßnahmen

Fehlerbehebung ❑
Anpassung ❑
Erweiterung ❑
Beratung/Info ❑
Schulung ❑

Verantwortlicher Sachbearbeiter

Name _____ Datum _____ Visum _____

Zwischenbescheid an Kunde

(erforderlich, wenn Meldung nicht bis zum vom Kunden erwarteten Termin erledigt werden kann)

Datum _____ Visum _____

Problem erledigt und Kunde informiert _____

Name _____ Datum _____ Visum _____

Zusammenfassung

- Konfigurations-Management ist das konsistente Management der einzelnen Software-Einheiten
- Jede Einheit ist eindeutig gekennzeichnet und über dem gesamten Lebenszyklus hinweg nachvollziehbar
- Konfigurations-Management beantwortet Fragen wie
  - Welche Software-Einheiten gehören zu einer Konfiguration?
  - Wie hängen die Einheiten voneinander ab?
  - Wie wird ein auslieferbares System generiert?
  - Welche Einheiten gehörten zur Version, die am12. Oktober 1999 ausgeliefert wurde?
  - Welche Änderungen wurden bei einer Einheit aus dieser Version bis jetzt gemacht? Änderungsanträge + Änderungsfreigaben.
- Strikt reglementiertes Änderungswesen stellt durch vorgegebene Prozesse die Wartbarkeit der Software sicher, so dass der Änderungsstand jederzeit nachweisbar ist.

Literatur – Software Engineering

- Skript Informatik II — Prof. Dr. Kühn / Fb W FH Karlsruhe
  http://www.home.fh-karlsruhe.de/~kuin0001/inhalt.htm
- Skript Software Engineering — Prof. Dr. Martin Glinz Universität Zürich
  http://www.ifi.unizh.ch/groups/req/courses/se_I/
- Skript Software Engineering II — Bernd Kahlbrandt FH Hamburg
  http://www.informatik.fh-hamburg.de/~khb/st4se2/
- Software Engineering
  Ian Sommerville — (ISBN3-82737-001-9)
- Software Engineering
  - Grundkurs für Praktiker –
  Roger S. Pressman — (ISBN 3-89028-163-X)
- Software Entwurf
  - Methoden und Werkzeuge –
  A. Schulz — (ISBN 3-486-21608-2)
- Software Engineering und Prototyping
  Thorsten Spitta — (ISBN 3-540-17542-3)
- CASE
  Helmut Balzert — (ISBN 3-411-03224-3)
- Software-Qualitätssicherung
  Ernest Wallmüller — (ISBN 3-446-15846-4)

Software Engineering

Informatik II.

11. Software-Entwicklung

– Produktivitätsfaktoren –