Universität der Bundeswehr München

Thema: Versionsverwaltung in der Dateisystemebene

Vortrag der Seminararbeit im Rahmen des Seminars "Langzeitarchivierung"

Seminararbeit eingereicht bei:
Univ. Prof. Dr. Uwe M. Borghoff

Betreuer:
Dipl.-Inform. Sebastian Rönnau

Vortragender:
Arthur Müller

Versionsverwaltung in der Dateisystemebene

1. Einführung

2. Verfahren zur Versionierung

→ Explizite Versionierung, Snapshots, Copy-on-Write

→ Probleme: Transparenz, Kontrolle

3. Versionierung und weitere Langzeitarchivierung

→ Zugriffsrechte, Metadaten, Versionsdarstellung

→ Probleme: Kontrolle, Überlauf, Speicherplatz

4. Zusammenfassung

Einführung

Langzeitarchivierung

~

Datensicherung

~

Versionsverwaltung

Einordnung ← Einführung

Userland-Tools / Repository:	**RCS, CVS, SVN**
Systemaufruf / Dateisystem	
Treiber	
Hardware	

Einordnung ← Einführung

Userland-Tools / Repository:	RCS, CVS, SVN
Systemaufruf / Dateisystem	
Treiber	
Hardware	

Verfahren zur Versionierung

- Explizite Versionierung
- Snapshots
- Copy-on-Write

Explizite Versionierung ← Verfahren zur Versionsverwaltung

Funktionalität wie bei den Userland-Tools z.B. CVS

```
check_in
check_out
commit
update
get_history
```

→ nicht transparent

→ bedarf Kooperation des Anwenders

→ Keine Kontrolle möglich

Snapshots ← Verfahren zur Versionsverwaltung

Def: im Allgemeinen nur mit Leserechten beschränkte Kopien des gesamten Dateisystems

→ nur in Verbindung mit inkrementeller oder differenzieller Sicherung sinnvoll

→ Vollständige Snapshots = Vollsicherung

→ oft mit Copy-on-Write verbunden

Snapshots ← Verfahren zur Versionsverwaltung

Zeitliche Ausführung der Snapshots

1. regelmäßig in vordefinierten Abständen: stündlich, täglich etc

   Problem: die Intervalle können zu groß für die notwendigen Zwecke sein

2. regelmäßig in vom Nutzer vorgegebenen Intervallen
3. manuel durch ein Programm

   => wenig transparent, da Anwenderkooperation erforderlich

4. *On Demand*, z.B. beim Abspeichern einer Datei

   => bietet die beste Transparenz

Snapshots ← Verfahren zur Versionsverwaltung

Snapshot-Größe

1. Das gesamte Dateisystem

   Nachteil:
   - alle Dateien gleichbehandelt
   - großer Speicherplatzverbrauch

   Vorteil:
   - ermöglicht vollständiges Recovery

2. Einzelne Dateien

   Nachteile:
   - muss konfiguriert werden
   - meist zusätzliche Werkzeuge notwendig

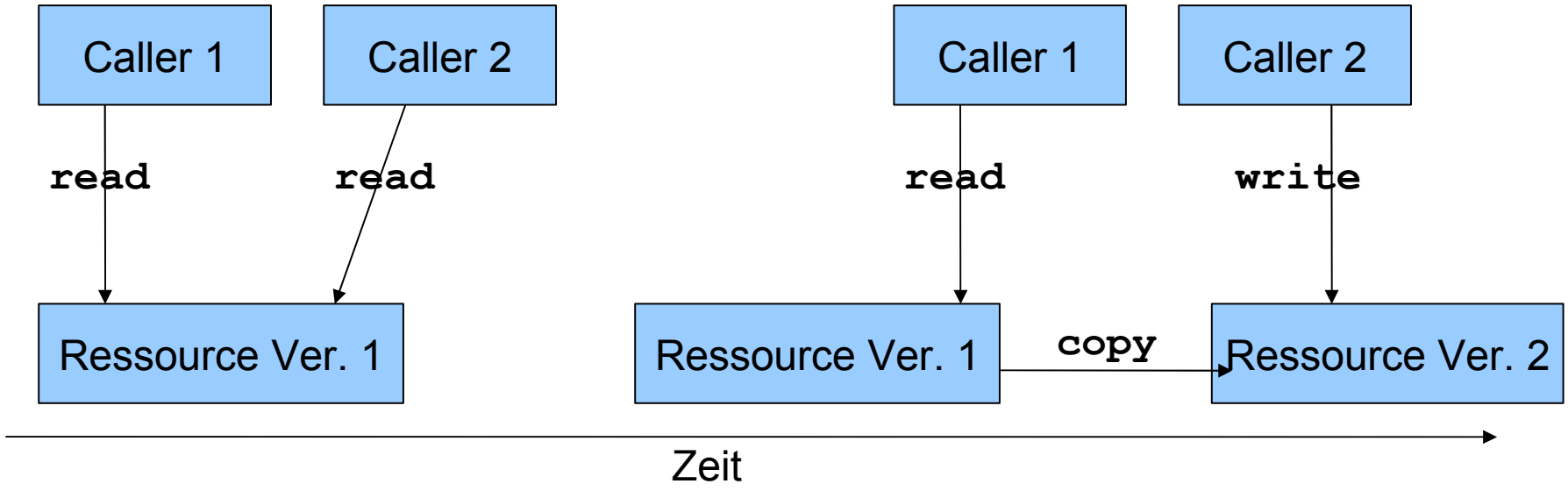
   => Transparenz leidet darunter

Copy-on-Write ← Verfahren zur Versionsverwaltung

Def: Im Allgemeinen
- Optimierungstechnik
- Einsparen des Speichers
- Anwendung bei Dateisystemen

Vorteile:
- Einsparen des Speicherplatzes
- *On Demand*
- Transparenz

Nachteil:
- Jeder Schreibzugriff mit Overhead verbunden



Copy-on-Change ← Copy-on-Write ← Verfahren zur Versionsverwaltung

Copy-on-Write im Allgemeinen:

→ neue Version bei jedem `write`-Aufruf

Copy-on-Change:

→ nur dann, wenn sich wirklich etwas ändert
→ Byte-für-Byte-Vergleich

→ **Vollständige Kopie = Vollsicherung**

→ **nur mit inkrementellen und differentiellen Methoden sinnvoll**

Verwalten der Versionen

Zugriff auf alte Versionen

→ Probleme bei der weiteren Archivierung

Darstellung der Versionen

→ Zugriff auf Archivmedien

Metadaten

→ Verwaltung auf den Archivmedien

Zugriffsrechte ← Verwalten der Versionen

1. Keine *on-line* Zugriffsrechte

- Nur Rollback bei einem System-Crash = Recovery
- Versionsverwaltung zu Sicherheitszwecken

2. Leserechte

- Benutzerzugriff auf Vorversionen
- Löschung gar nicht möglich

→ Sicherheit z.B. vor illegaler Nutzung d. Benutzer

→ Überlauf des Dateisystems !!!

Speicherüberlauf ← Zugriffsrechte ← Verwalten der Versionen

1. Komplett spiegeln

2. Aktuelle Versionen zwischenspeichern

3. Neu formatieren

4. Aktuelle Versionen wieder draufkopieren

→ Aufwendig
→ u.U. Eingriff des technischen Personals notwendig
→ die Arbeit mit Daten muss für längere Zeit unterbrochen werden

Zugriffsrechte ← Verwalten der Versionen

3. Erweiterte Leserechte

- Löschung durch System über vorgegebene/vordefinierte Schranken:
  - Zeit
  - Anzahl d. Versionen
  - Speicherplatz

- Löschung durch Administrator

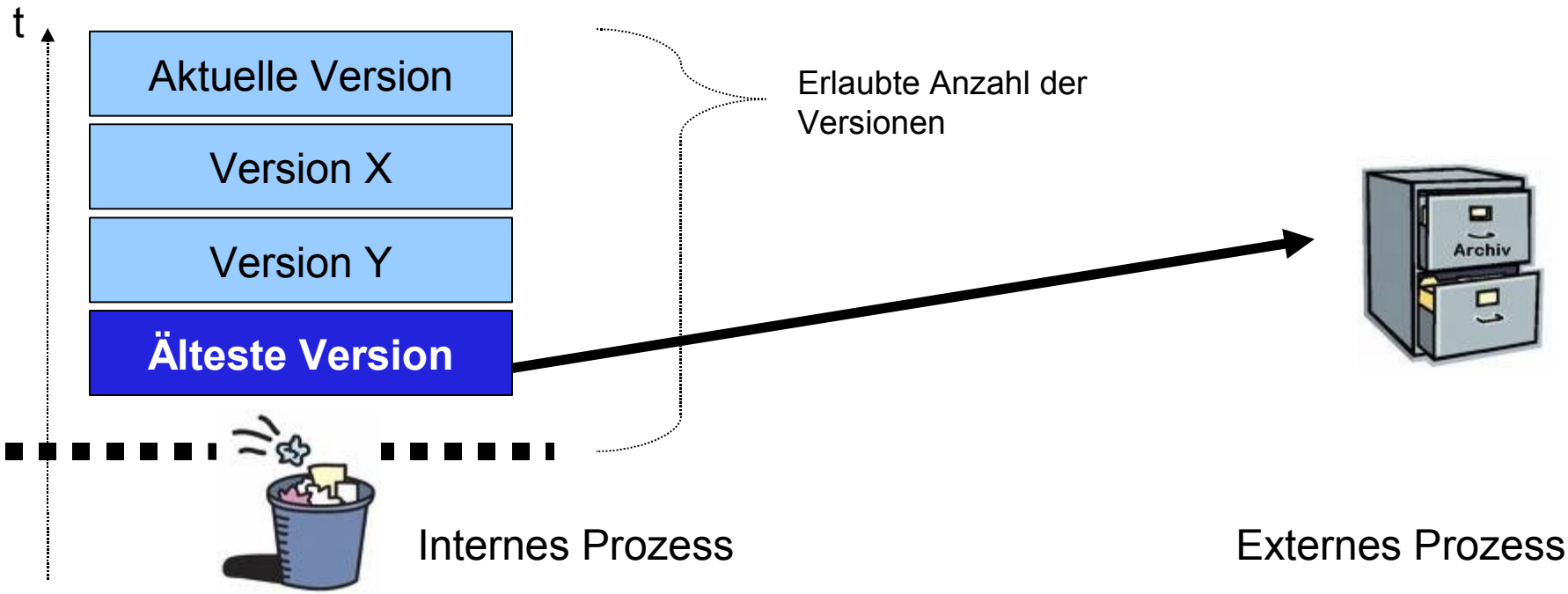
4. Schreibrechte

→ Gefahr durch Verfälschung der Daten
→ Gefahr für die Funktionsfähigkeit der Versionierung

Ständiges Archivieren der alten Versionen

→ Immer die älteste Version

→ Kurz vor der Löschung



Ständiges Archivieren der alten Versionen

← Zugriffsrechte ← Verwalten der Versionen

I.

→ Schnittstelle im Dateisystem zum Anstoßen des Archivprozesses

II.

→ Initiative beim Archivsystem

→ Dateisystem bietet mehr Einstellungen zum kontrollierten Löschen

→ Einstellung der Schranken

Ständiges Archivieren der alten Versionen

← Zugriffsrechte ← Verwalten der Versionen

Inkrementelle Ablage der Versionen: UNDO

→ Bei Ausfall der Platte wird der Archiv nutzlos

Vollständige Datei

Delta 10

Delta 9

Delta 8

Delta 8

Delta 7

Delta 6

Delta 5

...

Ständiges Archivieren der alten Versionen

Inkrementelle Ablage der Versionen: REDO

→ Sehr lange Zugriffszeiten auf die aktuelle Version

Delta 4	Delta 4
Delta 3	Delta 3
Delta 2	Delta 2
Delta 1	Delta 1
Vollständige Datei	**Vollständige Datei**

→ Dateisystem muss Zugriff auf Deltas erlauben

Darstellung der Versionen ← Verwalten der Versionen

1. Versionsnummern

- wie bei Software-Releases mit mehreren Zahlen, z.B. **linux-2.6.23**
- … oder eine fortlaufende Zahl: **1, 2, 3, … , aktuell**

~ Userland-Tools: **RCS, CVS**

→ **Benutzer muss die Version kennen oder mehrere ausprobieren**

2. Point of time

→ **Schnellerer Zugriff bei Kenntnis des Zeitpunkts, v.a. auf Archivmedien**

Metadaten ← Verwalten der Versionen

Wozu METADATEN ???

→ Zusatzdaten
→ Zeiger/Referenzen auf alle Versionen in der Sammlung
→ Angewendete Algorithmen
→ Verwendete Komprimierungsverfahren

etc.

Metadaten ← Verwalten der Versionen

Wozu METADATEN ???

→ Zusatzdaten
→ Zeiger/Referenzen auf alle Versionen in der Sammlung
→ Angewendete Algorithmen
→ Verwendete Komprimierungsverfahren

etc.

Parallele Dateien

→ Zentrale Ablage und Problem bei der Archivierung

Erweiterte Inodes

→ Kompatibilität mit dem Archivmedium

Parallele Dateien ← Metadaten ← Verwalten der Versionen

```
foo
foo;1
foo;2
foo;metadata

bar
bar;1
bar;metadata
```

Daten-Deltas

Zentrale Metadaten

Erweiterte Inodes ← Metadaten ← Verwalten der Versionen

Vorteile:

→ Metadaten dezentralisiert
→ Transparent und geschützt gespeichert

Nachteile und Probleme:

→ Dateisystem muss die Möglichkeit bieten RAW-Inodes auszulesen

→ Archivsystem muss den gleichen gleichen Typ der Inodes unterstützen

- eigentlich das gleiche Dateisystem haben
- Plattendateisysteme NICHT auf Bänder ausgelegt

Zusammenfassung

Optimale Transparenz

→ Copy-on-Write

Optimale Datensicherheit

→ Nur Leserechte
→ Löschung nicht möglich

Oder:
→ Leserechte und Löschung im Hintergrund verbunden mit der weiteren Archivierung

Optimale weitere Archivierung

→ Dezentrale Metadatenverwaltung
→ Inkrementelle Versionierungsverfahren
→ Weitere abgesicherte Schnittstellen zur
Abstimmung Dateisystem ↔ Archivsystem
→ Zugriff über Point of Time

Parallele Dateien ← Metadaten ← Verwalten der Versionen

t

Aktuelle Version	Aktuelle Version	Aktuelle Version
Delta 1	Delta 2	Delta 3
META	META'	Delta 2
	Delta 1	Delta 1
		META''

Archivierungszyklen

		Delta 3
		META''
	Delta 2	Delta 2
	META'	META'
Delta 1	Delta 1	Delta 1
META	META	META

→ Enormer Speicherplatzverbrauch

Nur lineare Speicherung möglich

Parallele Dateien ← Metadaten ← Verwalten der Versionen

t

Aktuelle Version	Aktuelle Version	Aktuelle Version
Delta 1	Delta 2	Delta 3
META	Delta 1	Delta 2
	META'	Delta 1
		META''

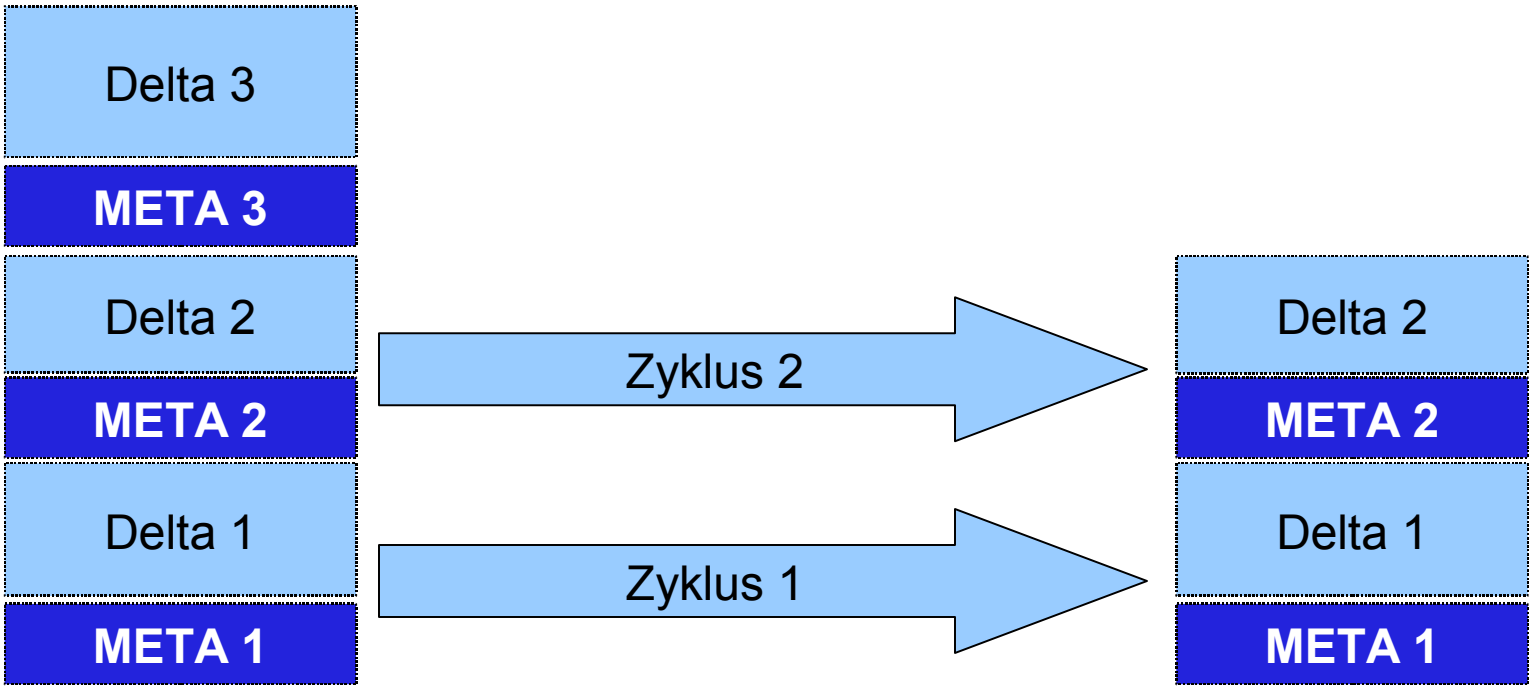
Delta 1	Delta 2	Delta 3
	Delta 1	Delta 2
		Delta 1

META	META'	META''

→ Band zurückspulen und überschreiben
→ Geschwindigkeit und Abnutzung ?

Parallele Dateien ← Metadaten ← Verwalten der Versionen

→ Dezentrfaliesierung der Metadaten in mehreren parallelen Dateien



Erweiterte Inodes ← Metadaten ← Verwalten der Versionen