

# Übungen zu Systemprogrammierung 1 (SP1)

## Ü 1 – Einführung

**Jens Schedel, Christoph Erhardt, Jürgen Kleinöder**

Lehrstuhl für Informatik 4  
Verteilte Systeme und Betriebssysteme

Friedrich-Alexander-Universität  
Erlangen-Nürnberg

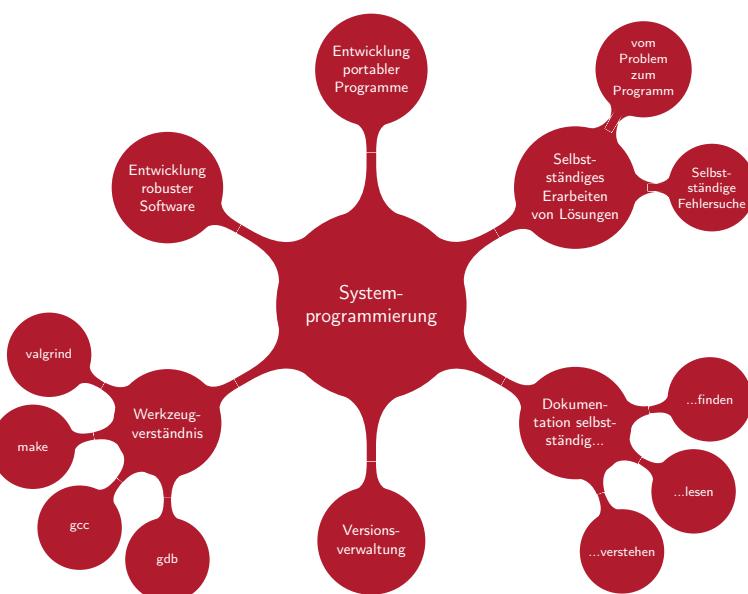
WS 2012/13 – 22. bis 26. Oktober 2012

[http://www4.cs.fau.de/Lehre/WS12/V\\_SP1](http://www4.cs.fau.de/Lehre/WS12/V_SP1)

01-Introduction\_handout



## Lernziele Systemprogrammierung



01-Introduction\_handout

## Agenda

- 1.1 Allgemeines
- 1.2 Organisatorisches
- 1.3 Linux-Kenntnisse
- 1.4 Versionsverwaltung mit SVN
- 1.5 SP-Abgabesystem
- 1.6 Gelerntes anwenden

01-Introduction\_handout



## Aufbau der Übungen

### Tafelübungen und Besprechungen

- Vorstellung nützlicher Werkzeuge
- Vorstellung von Betriebssystemkonzepten
- Einführung in die Verwendung der Schnittstellen
- Erarbeiten eines kleinen Programmes (Aktive Mitarbeit!)
- Besprechung der Abgaben und allgemeiner Fallstricke

### Praktischer Teil – Aufgaben

- Arbeiten mit der Betriebssystemschnittstelle
- Fehlersuche und Fehlerbehebung
- Verwenden der vorgestellten Werkzeuge

01-Introduction\_handout



- 1.1 Allgemeines
- 1.2 Organisatorisches
- 1.3 Linux-Kenntnisse
- 1.4 Versionsverwaltung mit SVN
- 1.5 SP-Abgabesystem
- 1.6 Gelerntes anwenden



## Praktischer Teil – Aufgaben

- Ausgabe neuer Aufgaben in den Tafelübungen
  - Aufgabenstellung meist recht knapp
    - Nicht alles bis in letzte Detail spezifiziert
  - Gegebene Spezifikationen sind dennoch zwingend einzuhalten
- Selbstständiges Bearbeiten der Aufgaben (vorzugsweise im CIP)
  - bei Problemen hilft z.B. ein Besuch in den Rechnerübungen
- Korrektur und Bewertung erfolgt durch den jeweiligen Tafelübungsteiler
  - korrigierte Ausdrucke werden in den Besprechungen ausgegeben
  - teilweise auch elektronisch zur Verfügung gestellt
  - eigenes Ergebnis nach Login im *WAFFEL* einsehbar
- Übungspunkte können das Klausurergebnis verbessern (Bonusnote)
  - Abschreibtests
  - Vorstellen der eigenen Lösungen



- Üblicherweise abwechselnd Tafelübung und Besprechung
  - Ausnahme: Nächster Übungstermin ist wieder Tafelübung

## Feiertag am 1.11.

Teilnehmer der T04 bitte in der Woche vom 29.10. bis 2.11. ausnahmsweise eine der anderen Tafelübungen besuchen.

- Ablauf einer Tafelübung werdet ihr gleich live erleben...



## Praktischer Teil – Bearbeitung der Aufgaben

- einzeln oder in Zweier-Teams je nach Aufgabe
  - bei Teamarbeit müssen beide Partner in der **gleichen** Tafelübung sein
- Bearbeitungszeitraum ist angegeben in Werktagen (bei uns: Montag bis Freitag)
  - Bearbeitungszeitraum beinhaltet den Tag der Tafelübung
  - Feiertage (01.11) und Weihnachtsferien (24.12. bis 05.01.) sind nicht enthalten
  - Abgabetermin kann per Skript erfragt werden
- plant für die Bearbeitung einer Aufgabe ca. 1-2 Tage (in Worten: ein bis zwei **Tage**) ein
  - langer Bearbeitungszeitraum bietet euch Flexibilität bei der Arbeitsverteilung
  - Feedback über wirkliche Bearbeitungszeit erwünscht



- Forum: <https://fsi.cs.fau.de/forum/18>
  - inhaltliche Fragen zum Stoff oder den Aufgaben
  - allgemein alles, was auch für andere Teilnehmer interessant sein könnte
- Mailingliste: [i4sp@informatik.uni-erlangen.de](mailto:i4sp@informatik.uni-erlangen.de)
  - geht an alle Übungsleiter
  - Angelegenheiten, die nur die eigene Person/Gruppe betreffen
- Rechnerübungen
  - Hilfe bei konkreten Problemen (z.B. Quellcode kompiliert nicht)
  - **kein** Händchenhalten während ihr die Tastatur bedient :)
  - angebotene Termine siehe Homepage
- der eigene Übungsleiter
  - Fragen zur Korrektur
  - fälschlicherweise positiver Abschreibtest



## Voraussetzungen

- UNIX-Grundkenntnisse werden vorausgesetzt
  - Übungsleiter sind in den Rechnerübungen bei Bedarf behilflich
- Zur Auffrischung: UNIX-Einführung der FSI  
<http://fsi.cs.fau.de/vorkurs>



## Linux-Install-Party der FSI

- am Donnerstag, den 25.10. um 16 Uhr
- weitere Informationen unter  
[http://fsi.cs.fau.de/dw/fsi/aktionen/linuxinstall\\_ws12](http://fsi.cs.fau.de/dw/fsi/aktionen/linuxinstall_ws12)

- 1.1 Allgemeines
- 1.2 Organisatorisches
- 1.3 Linux-Kenntnisse
- 1.4 Versionsverwaltung mit SVN
- 1.5 SP-Abgabesystem
- 1.6 Gelerntes anwenden



## Dokumentation aus 1. Hand: Manual-Pages

- Aufgeteilt in verschiedene *Sections*
  - 1 Kommandos
  - 2 Systemaufrufe
  - 3 Bibliotheksfunktionen
  - 5 Dateiformate (Spezielle Datenstrukturen etc.)
  - 7 verschiedenes (z.B. Terminaltreiber, IP)
- Angabe normalerweise mit *Section*: `printf(3)`
- Aufruf unter Linux:  
`> # man [section] begriff`  
`> man 3 printf`
- Suche nach *Sections*: `man -f begriff`
- Suche nach Manual-Pages zu einem Stichwort: `man -k stichwort`

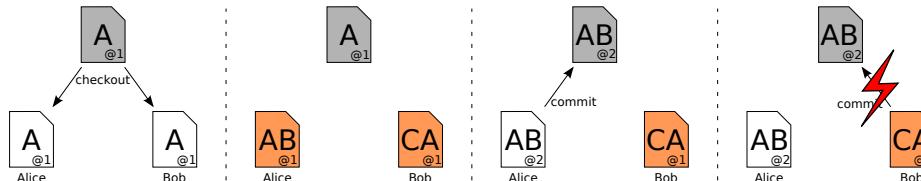


- 1.1 Allgemeines
- 1.2 Organisatorisches
- 1.3 Linux-Kenntnisse
- 1.4 Versionsverwaltung mit SVN**
- 1.5 SP-Abgabesystem
- 1.6 Gelerntes anwenden

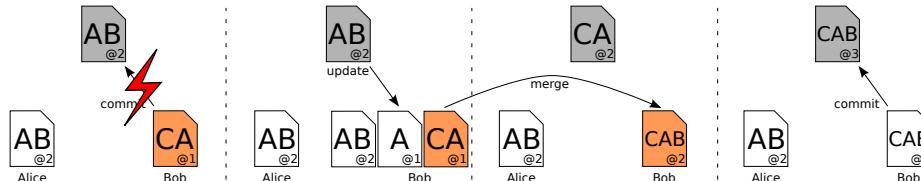


## Warum Versionsverwaltung?

- Versionsnummer zur Erkennung von Modifikationen

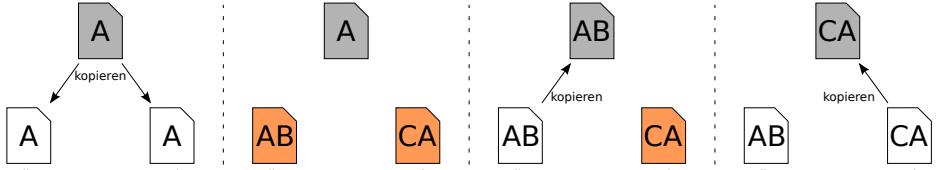


- Entstandener Konflikt muss lokal gelöst werden



## Warum Versionsverwaltung?

- Gemeinsames Bearbeiten einer Datei kann zu Problemen führen:

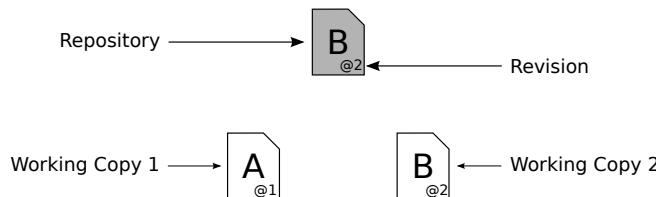


- Modifikationen werden nicht erkannt
- Änderungen von Alice gehen unbemerkt verloren



## Das Versionsverwaltungssystem Subversion (SVN)

- SVN bietet Versionsverwaltung für Dateien und Verzeichnisse
- Speichert Zusatzinformationen zu jeder Änderung
  - Name des Ändernden
  - Zeitpunkt
  - Kommentar
- Ausführliche SVN-Dokumentation im Subversion-Buch  
<http://svnbook.red-bean.com>
- Kommando `svn`
- Graphische Frontends
  - TortoiseSVN (Windows)
  - SCPlugin (Mac OS X)
- SP-Abgabesystem verwendet Subversion



- Repository: zentrales Archiv aller Versionen
  - Zugriff erfolgt beispielsweise per Internet
- Revision (Versionsnummer)
  - Fortlaufend ab Revision 0
- Working Copy (Arbeitskopie)
  - lokale Kopie einer bestimmten Version des Repositories
  - kann versionierte und unversionierte Dateien und Verzeichnisse enthalten
  - es kann mehrere Arbeitskopien zu einem Repository geben (z.B. CIP/daheim)

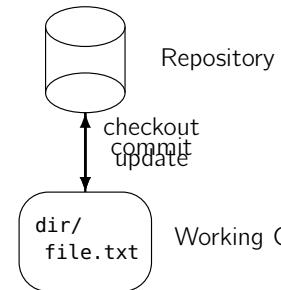
## Exkurs: Kommentar bei commit

- Beim Aufruf von `svn commit` öffnet sich ein Editor zum Eingeben des commit-Kommentars
  - Im CIP wird standardmäßig der Editor `joe` verwendet
    - Zum Speichern und Verlassen `Strg-k x` drücken
    - Hilfemenü öffnet sich mit `Strg-k h`
  - Anderer Editor kann über die Umgebungsvariable `EDITOR` eingestellt werden
 

```
> export EDITOR=nano
```

    - Umgebungsvariable ist nur in dieser Shell-Sitzung gültig
    - Durch Eintragen des Kommandos in die Konfigurationsdatei der eigenen Shell (z.B. `.bashrc`) wird der Standardeditor für jede neue Shell geändert
- Übergabe des Kommentars als Argument von `svn commit`

```
> svn commit -m "Ich schreibe lieber gleich in die Befehlszeile und nicht in den Editor"
```



- `checkout/co`: Anlegen einer neuen Arbeitskopie
- `update/up`: Neuste Revision aus dem Repository holen
  - Bezieht sich auf aktuelles Verzeichnis und alle enthaltenen Verzeichnisse
- `commit/ci`: Einbringen einer neuen Version in das Repository

## Basisoperationen 2

- `add`: Dateien unter Versionskontrolle stellen
  - Bei einer leeren Arbeitskopie müssen entsprechende Dateien oder Verzeichnisse erst eingefügt werden
- `del/remove/rm`: Dateien lokal löschen und nicht länger unter Versionskontrolle halten
- `status/st`: Änderungen der Arbeitskopie anzeigen
 

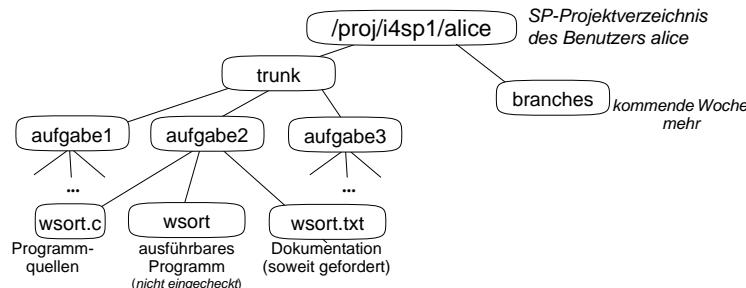
```
> svn status
A  aufgabe1/lilo.txt
M  aufgabe1/lilo.c
?  aufgabe1/lilo
!  aufgabe1/lilo.o
```

  - A** Datei wurde unter Versionskontrolle gestellt
  - M** Dateiinhalt wurde verändert
  - ?** Datei steht nicht unter Versionskontrolle
  - !** Datei steht unter Versionskontrolle, ist aber nicht mehr in der Arbeitskopie vorhanden

- 1.1 Allgemeines
- 1.2 Organisatorisches
- 1.3 Linux-Kenntnisse
- 1.4 Versionsverwaltung mit SVN
- 1.5 SP-Abgabesystem
- 1.6 Gelerntes anwenden



## Aufbau des SP-Repositories



- **trunk** enthält ein Unterverzeichnis **aufgabeX** für jede Aufgabe
- unterhalb von **branches** **nichts** editieren oder von Hand ändern



## SP-Abgabesystem

- Für jeden Teilnehmer wird folgendes bereitgestellt:
  - ein Repository <https://www4.cs.fau.de/i4sp/ws12/sp1/alice>
  - ein Projektverzeichnis `/proj/i4sp1/alice` mit Arbeitskopie
- Die Erzeugung erfolgt in der Nacht nach der *WAFFEL*-Anmeldung

### SVN-Passwort

- Zum Zugriff auf das Repository muss ein Subversion-Passwort gesetzt werden
  - `> /proj/i4sp1/bin/change-password`
  - Das Passwort wird innerhalb der nächsten Stunde aktiv

## Abgabe einer Aufgabe

- Zur Abgabe folgendes Skript aufrufen
  - `> /proj/i4sp1/bin/submit aufgabe0`
  - dieses gibt die aktuellste Version Ihres Repositories ab
  - offene Änderungen müssen vor der Abgabe eingechockt sein!
- mehrmalige Abgabe ist möglich
  - durch erneuten Aufruf des *submit*-Skripts
  - gewertet wird die letzte rechtzeitige Abgabe
- Abgaben nach dem Abgabezeitpunkt sind möglich
  - bei Vorliegen eines triftigen Grundes
  - Wertung nur nach expliziter Rücksprache mit dem Übungsleiter
  - ansonsten wird letzte rechtzeitige Abgabe gewertet

- **Eigener** Abgabetermin kann per Skript erfragt werden  
> `/proj/i4sp1/bin/get-deadline aufgabe1`  
Abgabezeitpunkt fuer Aufgabe 1: lilo: 2012-11-05 17:30:00
- Hilfsskripte sind nur im CIP-Pool verfügbar!

## Aktive Mitarbeit erforderlich!

### „Aufgabenstellung“

- Öffentliche Dateien für Aufgabe 1 ins Projektverzeichnis kopieren
- Vorgabe der Aufgabe 1 abgeben
  - Erforderliche Dateien: `lilo.c`
- „Zuhause“ neue Arbeitskopie anlegen



- 1.1 Allgemeines
- 1.2 Organisatorisches
- 1.3 Linux-Kenntnisse
- 1.4 Versionsverwaltung mit SVN
- 1.5 SP-Abgabesystem
- 1.6 Gelerntes anwenden