Fakultät für Informatik

MNM TEAM
Munich Network Management Team

Institut für Informatik

LMU

Systempraktikum im Wintersemester 2009/2010 (LMU):
Zum Selbststudium – Foliensatz 0

Systemprogrammierung: Erste Schritte (P)
Das Manpage-System (P)
Versionsverwaltung mit Subversion (P)

Dr. Thomas Schaaf, Dr. Nils gentschen Felde

Themenübersicht

MNM TEAM

- Erstes C-Programm
- Grundlegende Syntax von C-Programmen
- C-Programme übersetzen und ausführen
- Einfache formatierte Ausgabe
- Man-Pages als wichtiges Hilfsmittel der C-Programmierung
- Versionsverwaltung mit Subversion (SVN)
- Die wichtigsten SVN-Befehle

Willkommen zum Systempraktikum

MNM TEAM

- Dieser Foliensatz soll Sie auf das Systempraktikum im Wintersemester 2009/2010 vorbereiten.
- Bitte laden Sie sich die beiden zugehörigen Beispielprogramme von der Praktikumswebseite herunter, und testen Sie die Übersetzung und Ausführung am eigenen Rechner oder im CIP-Pool (bzw. remote auf einem CIP-Rechner).
- Experimentieren Sie ggf. mit Änderungen an diesen Programmen.
- Üben Sie den Umgang mit dem Manpage-System.
- Der Zeitaufwand zum Durcharbeiten dieses Foliensatzes einschließlich dem praktischem Nachvollziehen am Rechner sollte etwa 60 Minuten betragen.
- Beachten Sie bitte auch das Aufgabenblatt 0 "Einstieg in Linux", das auf der Praktikumswebseite bereitgestellt wird.

Erstes C-Programm

MNM TEAM

- Bevor ein C-Programm ausgeführt werden kann, muss der Quelltext in ein ausführbares Programm übersetzt werden. Die Komponente, die diese Übersetzung vornimmt, ist selbst ein (Hilfs-)Programm und heißt Compiler.
- Wir betrachten den Quelltext des folgenden C-Programms (first.c):

```
#include <stdio.h>

int main(int argc, char *argv[]) {

    //Ein Kommentar
    printf("Hallo Welt!\n");
    return 0;
}
```

- Jedes C-Programm benötigt eine `main()`-Funktion. Diese wird beim Start des Programms automatisch aufgerufen (analog zur main-Methode in einer Java-Klasse).

Erstes C-Programm

MNM TEAM

- Die `main()`-Funktion in first.c enthält nach dem Kommentar zwei C-Anweisungen:
  - Zunächst wird die Zeichenkette "Irgendein Text" gefolgt von einem Zeilenumbruch (`\n`) ausgegeben (`printf()`).
  - Anschließend wird 0 als Rückgabewert der Funktion zurückgegeben (`return 0`) und mit dem Verlassen der `main()`-Funktion auch das Programm beendet.
- Funktionsrümpfe werden durch geschweifte Klammern umschlossen `{ ... }`.
- Befehle werden durch `;` voneinander getrennt.
- Dass die `main()`-Funktion einen ganzzahligen Rückgabewert (Integer) hat, wird durch `int` in der Funktionssignatur angezeigt.
- Der Rückgabewert 0 wird standardmäßig verwendet, um anzuzeigen, dass keine Fehler aufgetreten sind. Das ist aber nur eine Konvention. Prinzipiell könnte man auch einen beliebigen anderen Integer-Wert zurückgeben lassen.

Übersetzung und Ausführung

MNM TEAM

- Speichern Sie das Programm first.c zum Beispiel in Ihrem Home-Verzeichnis.
- Um den Quelltext zu übersetzen und das Programm somit ausführbar zu machen, öffnen Sie eine Konsole (Shell), wechseln in das entsprechende Verzeichnis und geben das folgende Kommando ein:

```
gcc -o first first.c
```

- Anschließend starten Sie das Programm durch Eingabe von:

```
./first
```

Einfache formatierte Ausgabe

MNM TEAM

- Im Programm first.c wurde die Funktion `printf()` verwendet, die eine Zeichenkette auf der Konsole ausgibt. Diese Funktion hat selbst einen Rückgabewert vom Typ Integer (ganze Zahl), der angibt, wie viele Zeichen ausgegeben wurden.
- Das folgende Programm (formatoutput.c) ist eine modifizierte Version des ersten Beispielprogramms:

```
#include <stdio.h>

int main(int argc, char *argv[]) {

    int length;
    length = printf("Zeichenkette");
    printf("\nLänge: %i Zeichen\n", length);
    return 0;
}
```

- Neue Konstrukte:
  - Variablen-Definition: Es wird eine Integer-Variable mit dem Namen `length` definiert.
  - Zuweisung: Der Variablen wird als Wert der Rückgabewert des Aufrufs von `printf()` zugewiesen.

Man-Pages als wichtiges Hilfsmittel

MNM TEAM

- Eine Man-Page (Abkürzung für "Manual", Bedienungsanleitung) gibt dem Programmierer alle Informationen zu einem Shellbefehl, einem Systemaufruf oder einer Bibliotheksfunktion, wie zum Beispiel (im Falle einer Funktion):
  - Was macht die Funktion? (Beschreibung)
  - Wie kann ich sie aufrufen? (Funktionssignatur)
  - Was bedeuten die Parameter?
  - Welche Semantik hat der Rückgabewert?
  - Welche Header-Datei muss inkludiert werden, damit die Funktion genutzt werden kann?
  - ...
- Im Vergleich zu anderen (externen) Quellen hat das Man-Page-System einige Vorteile:
  - Die Seiten sind sofort verfügbar, da lokal gespeichert
  - Man-Pages haben ein weitgehend einheitliches Layout
  - Eine Man-Page enthält i.d.R. **alle** verfügbaren Informationen zu einem Befehl oder einer Funktion

Man-Pages als wichtiges Hilfsmittel

MNM TEAM

- Es ist wichtig, zwischen folgenden Arten von Befehlen und Aufrufen zu unterscheiden:
  - [1] Ausführbare Programme und Shellbefehle
    - Beispiele: gcc, ls, cd, ps, man
  - [2] Systemaufrufe (Kernel-Funktionen)
    - Beispiele: open, close, read, write
  - [3] Bibliotheksfunktionen
    - Beispiele: printf, scanf
  - [C] Kern-Bestandteile aus dem Befehlsvorrat der Sprache C
    - Beispiele: if ... else, while, return
- C-Programme bestehen oft aus einer Mischung der verschiedenen Befehlsarten – insb. [C], [2] und [3].
- Mit anderen Worten: Das Grundvokabular [C] wird durch [2] und [3] erheblich erweitert.
- Hinweis: Bibliotheksfunktionen können auf Systemaufrufen (System Calls) aufbauen. Mehr dazu in der ersten Vorlesung.

Man-Pages als wichtiges Hilfsmittel

MNM TEAM

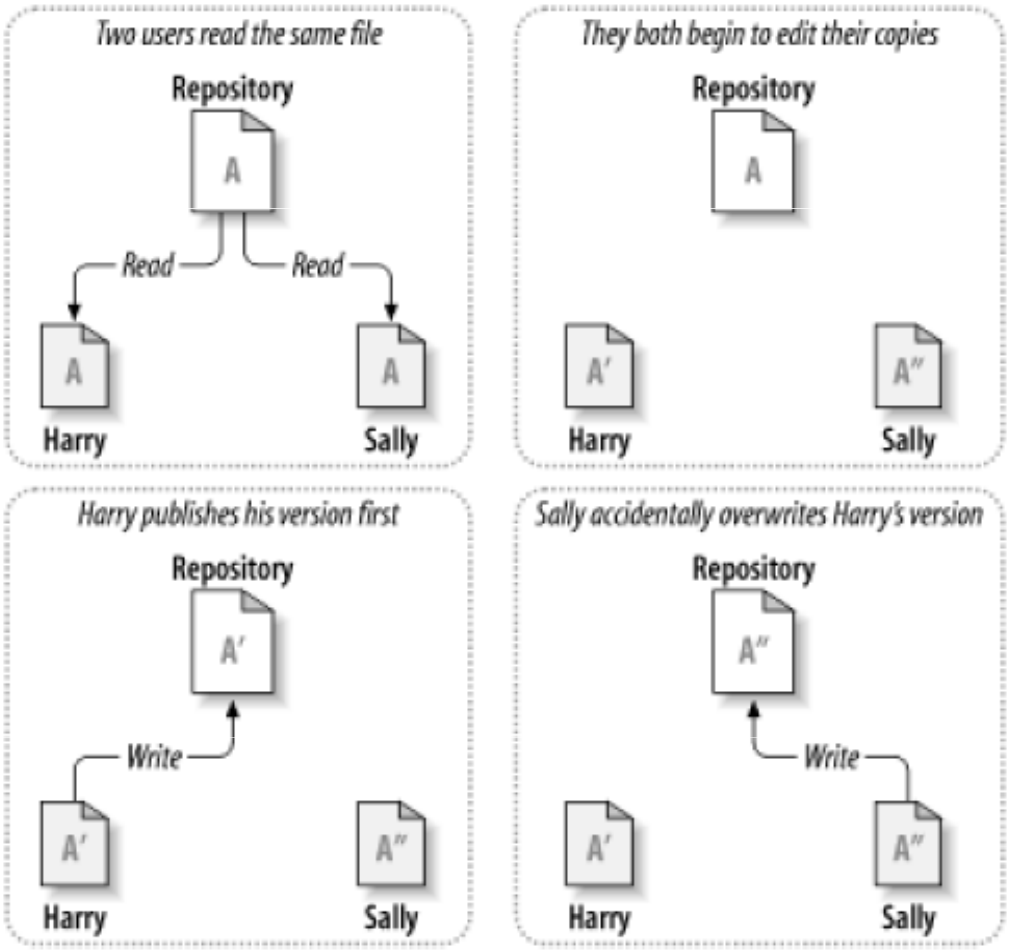
- Wie finde ich die gewünschte Man-Page?
  - Das Man-Page-System ist in mehrere Abschnitte/Sektionen unterteilt, die man sich wie Kapitel eines Buches vorstellen kann.
  - In welcher Sektion man suchen muss, hängt davon ab, ob man Informationen zu einem [1] Shellbefehl, einem [2] Systemaufruf oder einer [3] Bibliotheksfunktion benötigt (siehe vorherige Folie).
- Beispiele
  - Gesucht: Informationen zum Befehl `ls`
    - Shell-Befehl → Sektion 1
    - Aufruf der Man-Page: `man 1 ls` (identisch mit `man ls`)
  - Gesucht: Informationen zur Funktion `printf()`
    - Bibliotheksfunktion → Sektion 3
    - Aufruf der Man-Page: `man 3 printf`
- Man-Pages verstehen
  - Anfangs wirken Man-Pages nicht gerade selbsterklärend.
  - Im Rahmen des Praktikums werden Sie lernen, Man-Pages effektiv als Hilfsmittel beim Programmieren einzusetzen.
  - Übrigens: Zum Befehl `man` gibt es selbst eine Man-Page → `man man`

Versionsverwaltung mit Subversion

MNM TEAM

- Versionsverwaltungssystem – wozu?
  - Verwaltung unterschiedlicher Versionen eines Software-Projekts
  - Ermöglichung des parallelen Arbeitens mehrerer Programmierer
  - Konfliktbehandlung und -vermeidung
  - Erhaltung eines konsistenten Gesamtzustands
- Grundlegende Konzepte
  - Zentrales Repository: Alle Informationen zu Dateien und Verzeichnissen eines Projekts an einem einzigen, zentralen Speicherort → Im Repository wird niemals direkt gearbeitet.
  - Lokale Arbeitskopie: Kopie des Projekts zur Bearbeitung durch einen bestimmten Benutzer → Es können mehrere Arbeitskopien existieren, in denen unabhängig gearbeitet/editiert werden kann.
- Mehr zu Subversion
  - SVN-Buch: http://svnbook.red-bean.com/nightly/en/svn-book.pdf
  - SVN am CIP: http://www.rz.ifi.lmu.de/Dienste/Subversion
  - Wikipedia: http://de.wikipedia.org/wiki/Subversion_%28Software%29
- Im Folgenden: Auszüge aus dem SVN-Buch

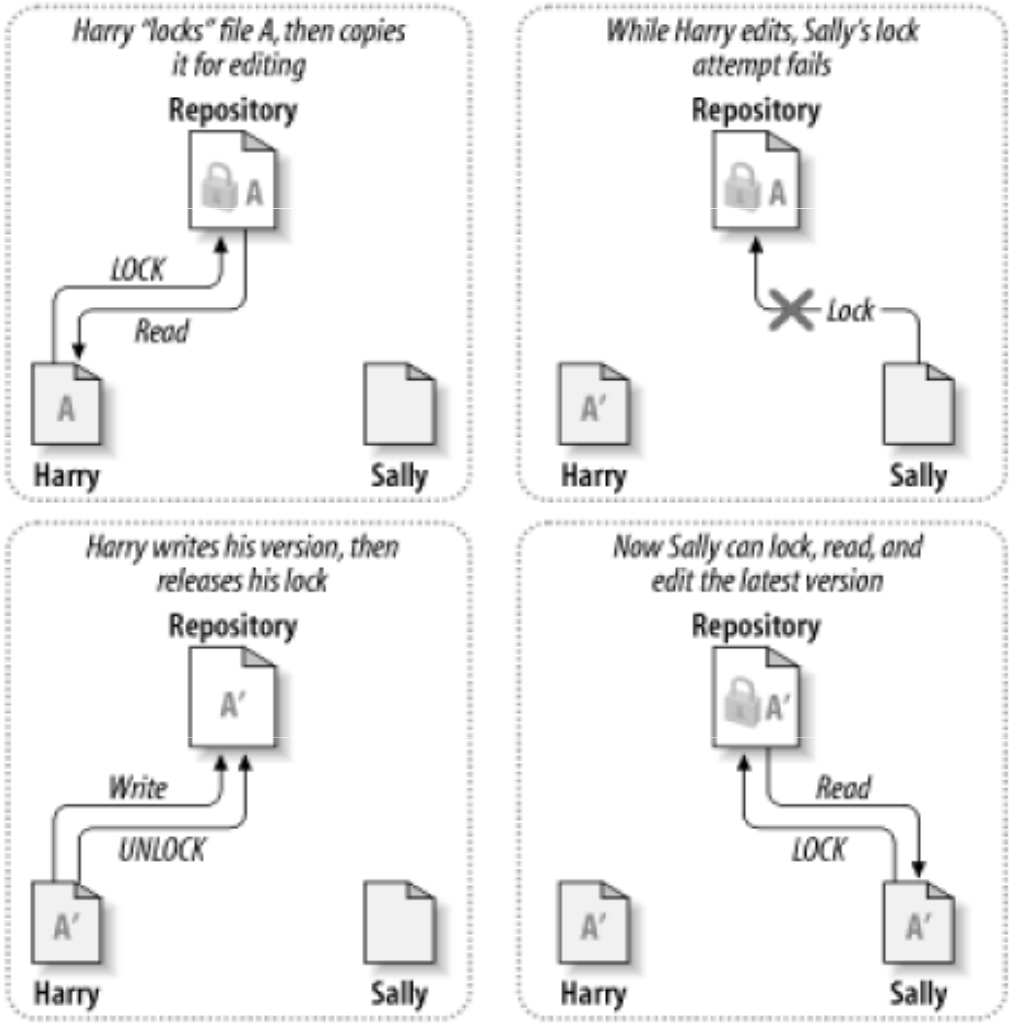
- Folgendes Problem gilt es zu vermeiden



Versionsverwaltung mit Subversion

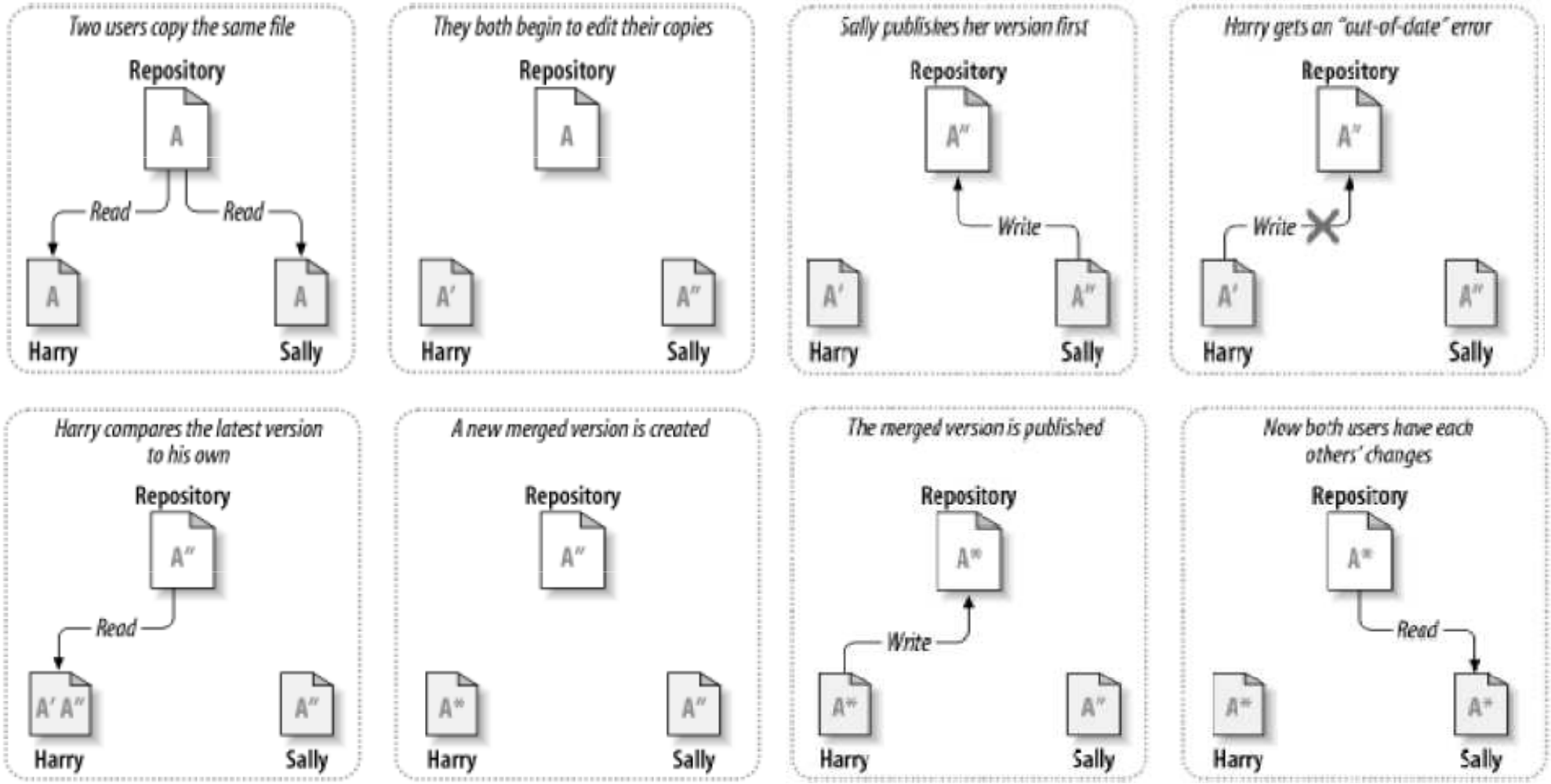
MNM TEAM

- Ansatz 1: Lock-Modify-Unlock



- Probleme des Ansatzes:
  - -Keine Parallelität
  - -Harry könnte das UNLOCK vergessen
  - -Granularität eines Locks schwer zu bestimmen – zum Beispiel:
    - Lock auf Verzeichnis-Ebene → Sally kann Datei B nicht editieren, obwohl Harry nur an A arbeitet (da gesamtes Verzeichnis von Harry gelockt)
    - Lock auf Datei-Ebene → Sally kann B editieren, aber was wenn A und B in Abhängigkeit stehen und Harrys Änderungen an A inkompatibel zu Sallys Änderungen an B sind?

- Ansatz 2: Copy-Modify-Merge (wird von SVN verfolgt)



- Die SVN-Befehle
  - `svnadmin <unterbefehl>`: Administration eines SVN-Repositorys, z.B. Anlegen eines neuen Repositorys mit

    `svn create`
  - `svn <unterbefehl>`: Arbeiten mit einem SVN-Repository, z.B. Auschecken einer Arbeitskopie aus einem Projektarchiv mit

    `svn checkout`
  - Hilfe zu den SVN-Befehlen:

    `svn help`

    `svnadmin help`
  - Hilfe zu einem Unterbefehl:

    `svn help <unterbefehl>`

    `svnadmin help <unterbefehl>`

Versionsverwaltung mit Subversion

MNM TEAM

- Die fünf wichtigsten SVN-(Unter-)Befehle
  - `svnadmin create`

    Anlegen eines neuen SVN-Repositorys, z.B. zu Beginn eines Software-Projekts
  - `svn import`

    Datei oder Verzeichnis(baum) zum Projekt hinzufügen, d.h. bisher nicht versionierte Dateien werden ins Projektarchiv übertragen und ab sofort von SVN mitverwaltet
  - `svn checkout`

    Auschecken einer aktuellen Arbeitskopie zum lokalen Editieren
  - `svn update`

    Abgleich und Aktualisierung der Arbeitskopie mit den seit dem Auschecken erfolgten Änderungen am Projektarchiv
  - `svn commit`

    Eigene Änderungen an der Arbeitskopie ins Projektarchiv übertragen