Technische Universität München
Lehrstuhl Informatik VIII
Prof. Dr.-Ing. Georg Carle
Dipl.-Ing. Stephan Günther, M.Sc.
Nadine Herold, M.Sc.
Dipl.-Inf. Stephan Posselt

Programmieraufgaben zur Vorlesung
Grundlagen Rechnernetze und Verteilte Systeme

Kurzeinführing in Subversion

Was ist Subversion?

Subversion (SVN) ist eines der gängigen Versionsverwaltungssysteme, welche in der Softwareentwicklung eingesetzt werden. Es ermöglicht mehreren Programmierern gemeinsam an einem Softwareprojekt zu arbeiten. Quelltexte werden in einem **Repository** abgelegt, welches auf einem Server bereitliegt. Von dort kann der Inhalt des Repositories ausgecheckt werden, so dass eine lokale Kopie vorliegt. Auf der Kopie kann nun gearbeitet werden. In regelmäßigen Abständen sollte ein **commit** durchgeführt werden, welcher Änderungen auf den Server repliziert. Dabei werden im Repository lediglich die Änderungen gegenüber der letzten Revision gespeichert. Andere Benutzer können diese Änderungen mittels eines **update** herunterladen und so ihre lokale Kopie aktualisieren.
Neben der Verteilungsfunktion bietet SVN insbesondere auch die Möglichkeit, zu früheren Revisionen zurück zu kehren. Sollten sich bestimmte Änderungen als nicht sinnvoll erweisen, kann man mittels **revert** die Änderungen wieder rückgängig machen.

Für ausführliche Informationen über Subversionen sind die folgenden Links hilfreich:

- http://de.wikipedia.org/wiki/Apache_Subversion
- http://subversion.apache.org/faq.html
- http://svnbook.red-bean.com/

Warum brauchen wir SVN?

Es gibt mehrere Gründe, weswegen wir für die Programmieraufgaben SVN verwenden:

1. Jeder Informatiker sollte mit wenigstens einem Versionsverwaltungssystem umgehen können. Falls Sie es nicht schon längst können, sollten Sie es besser früher als später lernen.

2. Sollten Sie in einem Team arbeiten, werden Sie SVN schnell zu schätzen lernen. Es erleichtert nicht nur die Zusammenarbeit, sondern bietet Ihnen auch ein automatisches Backup Ihrer Programme.

3. Wir brauchen ein System, wie Sie Ihre Programme abgeben können. Anstatt uns Ihre Porgramme via Email zu schicken oder als Tarball über ein Webformular hochzuladen, müssen Sie nur sicherstellen, dass Ihre aktuelle Version im SVN liegt.

Mercurial / Git ist doch viel besser als SVN!

Stimmt. SVN ist aber dennoch verbreiteter – insbesondere im industriellen Umfeld. Und wer Mercurial oder Git bedienen kann, wird sich für die Programmieraufgaben sicher auch mit SVN arrangieren können.

Wie bekomme ich SVN?

- Linux-Nutzer installieren `svn` mit der Paketverwaltung ihrer Wahl.
- Nutzer von OS X erhalten SVN entweder über MacPorts bzw. Fink oder können sich wahlweise auch vorkompilierte Versionen aus dem Netz laden.
- Windows-Nutzer wollen sicherlich einen grafischen Client. Empfehlenswert ist hier TortoiseSVN, welcher sich direkt in das Kontextmenü des Windows-Explorers integriert.

Wie funktioniert SVN?

Die folgende Anleitung ist nur für die Kommandozeilen-Clients unter Linux und OS X gültig. Für den grafischen Client unter Windows finden sich im Netz viele Tutorials. Der prinzipielle Ablauf ist aber auch unter Windows derselbe.

Auschecken (checkout)

Zunächst müssen Sie Ihre Gruppenverzeichnis auschecken (Passwort ist das Ihres MyTUM-Accounts):

```
svn co --username <LRZ-Kennung> https://projects.net.in.tum.de/svn-tum/grnvss13 <Zielverzeichnis>
```

Möglicherweise erhalten Sie eine Fehlermeldung, dass die Zugriffsrechte auf den Private-Key inkorrekt sind. Das ziehlt darauf ab, dass nur Sie und sonst niemand Schreibrechte darauf haben sollten. In diesem Fall (es wird nur Linux und OS X betreffen, unter Windows ist das eh egal), ändern Sie bitte die Schreibrechte auf den Private-Key mittels folgenden Kommandos:

```
chmod 600 <Pfad zum Private-Key>
```

Dateien zum Repository hinzufügen (add)

Wenn Sie neue Quelltextdateien anlegen, müssen Sie diese zum Repository hinzufügen. Erst wenn Sie die jeweiligen Dateien hinzugefügt haben, werden sie von SVN erfasst:

```
svn add <Dateiname>
```

Sie können auf diese Weise auch ganze Verzeichnisse hinzufügen. Allerdings sollten Sie stets darauf achten, dass Sie ausschließlich Textdateien und keine kompilierten Programme oder andere Binärdateien einchecken. Der Grund dafür besteht darin, dass SVN stets nur die Änderungen gegenüber der letzten Version speichert. Typischerweise ändert sich von einer Version auf die nächste nur ein kleiner Teil einer (Quell-)Textdatei. Kompilierte Dateien hingegen können sich vollständig ändern. Abgesehen davon ist das Einchecken der kompilierten Programme vollkommen überflüssig, da die Binaries jederzeit aus den Quellen neu übersetzt werden können.
Manchmal verliert man den Überblick, welche Dateien man bereits hinzugefügt hat und welche man ggf. vergessen hat. Dies lässt sich leicht mit folgendem Befehl prüfen:

```
svn status
```

SVN listet alle Dateien unterhalb des aktuellen Verzeichnisses auf. Dateien mit einem führenden `?` werden von SVN nicht verwaltet. Ein führendes `M` hingegen bedeutet, dass diese Datei lokale Änderungen beinhaltet, die noch nicht mit dem Repository abgeglichen wurden.

Dateien / Änderungen einchecken (commit)

Sie sollten **regelmäßig** den aktuellen Stand Ihres Programms einchecken („committen“). Dadurch werden Ihre Änderungen in das Repository repliziert. Erst danach können Ihre Teammitglieder mittels eines **update** ihre eigene lokale Kopie aktualisieren.

```
svn commit -m ‘‘<Beschreibung>’’
```

Sie sollten unbedingt bei jedem Commit eine kurze Beschreibung der Änderungen hinzufügen. Dies ermöglicht es einerseits Ihren Teamkollegen schnell zu erkennen, was Sie geändert haben. Andererseits ist es auch sehr hilfreich, wenn Sie später zu einer früheren Version zurückkehren möchten.

Dateien / Änderungen aktualisieren (update)

Bevor Sie Änderungen an Ihrer lokalen Kopie vornehmen, sollten Sie stets ein Update durchführen. Damit stellen Sie sicher, dass Ihre lokale Kopie auf dem aktuellen Stand ist. Wechseln Sie dazu in Ihr Teamverzeichnis und führen Sie den folgenden Befehl aus:

```
svn update
```

Weiterführende Dokumentation

Für weitere Details über SVN, insbesondere das **revert** auf ältere Versionen sowie **Branches**, werfen Sie bitte einen Blick auf umfangreiche aber gut gegliederte offizielle SVN-Dokumentation. Diese finden Sie unter http://svnbook.red-bean.com/en/1.5/index.html.

Die Verzeichnisstruktur

Abbildung 1 zeigt exemplarisch die Verzeichnisstruktur des SVNs. Nach dem Auschecken erhalten Sie zwei Verzeichnisse:

- Im Ordner `./pub` finden Sie die Vorlesungsunterlagen, die auch auf der Homepage zum Download bereitstehen.
- Ihrem Teamordner `./team<xyz>`. Da Sie nur auf Ihren eigenen Teamordner Leserechte haben und deswegen auch nur diesen einen Teamordner auschecken können, erfahren Sie auf diese Weise Ihre Teamnummer.

Innerhalb Ihres Teamordners finden Sie eine Reihe von Unterordnern und Dateien:

- Einen Ordner für jedes Assignment. Innerhalb dieser Ordner können Sie bei Bedarf weitere Unterordner anlegen. Dies sind auch die einzigen Ordner, in denen Sie Schreibrechte haben. Nach der Deadline der jeweiligen Programmieraufgabe, werden die Schreibrechte im betreffenden Ordner entfernt, so dass keine nachträglichen Änderungen mehr möglich sind.
- Im Ordner `./team<xyz>/ssh` finden Sie RSA-Schlüssel zum Zugang zu Ihrer GRNVS VM (siehe Einührung zu den VMs). Bewahren Sie den Private Key `svm<xyz>.i<n>.as.net.in.tum.de-id_rsa` sorgfältig auf und geben Sie ihn nicht weiter!
- In der Datei `./team<xyz>/grades.txt` werden die Ergebnisse der Programmieraufgaben bekannt gegeben.

```
.
|-- pub
|   |-- problem01.pdf
|   |-- problem02.pdf
|   |-- slides_chap0.pdf
|   |-- slides_chap1.pdf
|   `-- solution01.pdf
|-- team000
|   |-- assignment01
|   |-- assignment02
|   |-- assignment03
|   |-- grades.txt
|   `-- ssh
|       |-- svm<xyz>.i1.as.net.in.tum.de-id_rsa
|       `-- svm<xyz>.i1.as.net.in.tum.de-id_rsa.pub
```

Abbildung 1: SVN Verzeichnisstruktur

Übungsaufgabe

Bitte stellen Sie zunächst sicher, dass Sie und Ihre Teammitglieder Zugriff auf das SVN-Repository haben. Bei Problemen wenden Sie sich bitte an guenther@tum.de. Machen Sie sich bitte mit SVN vertraut.
In den nächsten Assignments werden Sie lauffähigen Programm-Code schreiben und in das Repository einchecken. Für dieses Assignment sollen Sie lediglich eine einzelne Textdatei einchecken.

1. Erstellen Sie im Repository im Ordner `team<abc>/assignment01` eine Datei mit dem Namen `note.txt`.

2. Als Inhalt schreiben Sie in die Datei „Ich liebe GRNVS!“ und was Ihnen sonst noch erwähnenswert erscheint.

3. Committen Sie ihre Änderungen. Ihr Team-Partner sollte die Datei nach einem erfolgreichen `update` in seinem Dateisystem vorfinden.

Dieses Assignment bleibt noch unbewertet. Wir gehen bei den zukünftigen Aufgaben allerdings davon aus, dass Sie SVN verwenden können. Deswegen empfehlen wir jedem, auch diese Aufgabe im vorgesehenen zeitlichen Rahmen zu bearbeiten.