Technische Universität München
Lehrstuhl Informatik VIII
Prof. Dr.-Ing. Georg Carle
Dr. Alexander Klein
Stephan M. Günther, M.Sc.

Tutorial – Discrete Event Simulation
Introduction to Subversion, WS11/12

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 funktioniert SVN?

Die nachfolgende Beschreibung bezieht sich auf den Kommandozeilen-Client. Es gibt auch grafische Frontends, die man aber nur unter Windows verwenden will.

Installation

- Debian/GNU-Linux:
  `sudo apt-get update`
  `sudo apt-get install subversion`
- OS X:
  `sudo port install subversion`
  MacPorts nicht installiert? Dann ist jetzt ein guter Zeitpunkt dafür. MacPorts für alle aktuellen OS X Versionen gibt es auf http://www.macports.org/install.php. Alternativ gibt es für OS X auch diverse Binaries.
- Windows XP/Vista/7:
  TortoiseSVN ist ein kostenfreier grafischer Client, der sich ins Rechtsklick-Menü einnistet. Den Download gibt es auf http://tortoisesvn.net/downloads.html.

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/des <Zielverzeichnis>
```

Anschließend sollten Sie die beiden folgenden Unterverzeichnisse erhalten haben:

- `team<nummer>`: Dies ist Ihr Gruppenverzeichnis, in dem Sie Ihre Programme entwickeln sollten. Darunter befinden sich für jede Programmieraufgabe je ein Unterverzeichnis sowie eine grading.txt. Die jeweilige Programmieraufgabe muss bis zum Abgabetermin (s. Übungsblätter) vollständig in das entsprechende Unterverzeichnis eingecheckt werden. Nach Abschluss der Korrektur (ca. eine Woche nach der Abgabe) finden Sie die Bewertung der Aufgabe sowie Anmerkungen in der jeweiligen grading.txt. Beachten Sie bitte, dass Sie nur in den Unterverzeichnissen `assignment<nummer>` Schreibrechte besitzen und diese nach der Abgabefrist auf read-only geändert werden.
- `pub`: Über dieses Verzeichnis werden wir Ihnen die Programmieraufgaben bereitstellen. Kopieren Sie sich zu gegebener Zeit die jeweilige Programmieraufgabe in Ihr Teamverzeichnis. Weitere Details folgen auf den Angaben zu den Programmieraufgaben.

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 info
```

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.

Wie gehts jetzt weiter?

Bitte stellen Sie zunächst sicher, dass Sie und Ihre Teammitglieder Zugriff auf das SVN-Repository haben. Bei Problemen wenden Sie sich bitte an die Übungsleitung (guenther@in.tum.de). Machen Sie sich bitte mit SVN vertraut – Sie können innerhalb eines der bereits existierenden Unterverzeichnisse ein weiteres Verzeichnis für Tests anlegen.