Theorieteil

1 Verteiltes Programmieren mit Subversion

1.1 Allgemeines

Subversion ist ein *Version Control System*, ein Programm zur Versionsverwaltung von Dateien, hauptsächlich Software-Quellcode. Subversion ist eine Weiterentwicklung des CVS (das noch heute viel eingesetzt wird, z.B bei SourceForge), welches seinerseits eine Weiterentwicklung des RCS (Revision Control System) ist. Subversion (SVN) erlaubt mehreren Developers gemeinsam an einem Projekt zu arbeiten, ohne sich gegenseitig Code zu löschen.

1.2 Funktionsweise

Subversion vereinfacht die Verwaltung von Quellcode dadurch, dass es alle Dateien eines Software-Projektes an einer zentralen Stelle, einem so genannten *Repository*, speichert. Dabei können jederzeit einzelne Dateien verändert werden, es bleiben jedoch alle früheren Versionen erhalten, einsehbar und wiederherstellbar, auch können die Unterschiede zwischen bestimmten Versionen verglichen werden.
Ein Entwickler holt sich die Daten vom Server (meist über http/https) und arbeitet zu Hause an einer lokalen Kopie. Sobald er seine Arbeit veröffentlichen will, wird er die Daten (die Änderungen) wieder auf den Server laden (*commit*). Der Server wird dann seine eigene Kopie updaten und schauen, dass alle anderen Entwickler an eine aktuelle Version gelangen können. Probleme können sich ergeben, wenn mehrere Mitarbeiter gleichzeitig an der selben stelle in einer Datei Änderungen vornehmen. In diesem Fall muss der Entwickler manuell die verschiedenen Änderungen zusammenfügen (*merge*).

1.3 Anwendungslexikon

- Daten herunterladen: Dieser Befehl lädt die neueste Version von `[repository]` an die angegebene `[destination]`.

```
svn checkout [repository] [destination]
svn co [repository] [destination]
```

- Dateien hinzufügen: So kannst du Daten oder Verzeichnisse (`[file]`) hinzufügen. Alle Daten die zum vollständigen Projekt gehören sollten auch hinzugefügt werden.

```
svn add [file]
```

- Dateien löschen: Um Daten im Repository zu löschen, musst du diesen Befehl verwenden.

```
svn rm [file]
```

- Änderungen anzeigen: So kannst du deine lokalen Änderungen einsehen.

```
svn diff
```

- Neue Version herunterladen: Synchronisiert das lokale Repository mit dem Server und lädt alle Änderungen herunter.

```
svn up
```

- Daten hochladen: Dieser Befehl ladet deine lokalen Änderungen auf den Subversion Server. Der Verständlichkeit zuliebe noch ein schöner Kommentar `[message]`.

```
svn checkin -m "message"
svn ci -m "message"
```

- Weitere Informationen: Falls du mehr über einen gewissen `[command]` erfahren willst[1].

```
svn help [command]
```

1.4 Subversion in Orxonox

Orxonox verwendet eine spezielle Syntax der Log-Nachrichten beim commiten von Änderungen. Diese Notation erlaubt dem Betrachter einer Log-Nachricht jederzeit die Tätigkeit und den Arbeitsort festzustellen:

```
svn ci -m "orxonox/[folder]: [message]"
```

Wobei `[folder]` den Ort bezeichnet, an dem du gearbeitet hast, als z.B. in *trunk* oder in *branches/student1*. Die Nachricht `[message]` sollte möglichst kurz und einfach beschreiben, was verändert worden ist und weswegen (dies ist insbesondre wichtig, wenn man Code von anderen Leuten verändert).
Auch wenn es zwischendurch muehsahm erscheinen mag: es ist sehr wichtig, dass du **immer eine Log-Nachricht spezifizierst** beim commiten, da die anderen Entwickler den Inhalt des commits selber herausfinden müssen.

[1] Alternative Informationsquelle: `http://svnbook.red-bean.com/`

Praktikum

2 Orxonox Installieren, Kompilieren und Starten

In dieser ersten Lektion wirst du lernen, wie man Orxonox auf den Tardis Rechnern der ETH zum laufen bringt. Dazu wirst du die gebräuchlichen Open Source Tools *subversion* verwenden um die Daten zu *fetchen* und die Automake/Autoconf scripts um das Projekt zu komilieren.
Im zweiten Teil wirst du bekannte Tools ausprobieren, die zur Kommunikation zwischen den Entwicklern dient.

Source Code runterladen: Zuerst müssen wir aber ein Verzeichnis erstellen und wechseln in dieses:

```
mkdir orxonox
cd orxonox
```

Nun können wir den Source auschecken. Dazu wirst du deinen Benutzernamen mit dem dazugehörigen Passwort brauchen, dass wir dir auf einem separaten Blatt gegeben haben (das Blatt mit Passwort bitte nach Eingabe und Auswendiglernen aufessen :D). Orxonox braucht zur Ausführung zwei *Repositories*, eines für den Source (`../orxonox/`) und eins für Daten wie Grafiken, Level und Models (`../data/`). Die *Repositiries* werden sowohl via ssl verbindung (https) als auch ungeschützt angeboten (http). Deine Änderungen im Orxonox Source Code kannst du aber nur über die verschlüsselte Verbindung übertragen. Heute verwenden wir noch den anonymen http Zugang:

```
svn co http://svn.orxonox.net/orxonox/trunk trunk
svn co http://svn.orxonox.net/data/trunk data
```

Später, wenn wir dann den Source Code verändern werden wirst du den die verschlüsselte Verbindung verwenden müssen. Dafür wirst du noch einen Benutzernamen/Passwort erhalten.

```
## Verschl\"usselte Verbindung
svn co https://svn.orxonox.net/orxonox/trunk trunk
svn co https://svn.orxonox.net/data/trunk data
```

Source Code Compilieren Versuche nun Orxonox zu kompilieren. Dazu verwendest du die standard GNU Compiler Collection (`GCC`) wie g++ und `make`.

1. Generation der Automake Files: Damit du den source configurieren und maken kannst, musst du zuerst folgendes eingeben:

   ```
   ./autogen.sh
   ```

2. Spezielles configure flag für tardis: Damit der Orxonox code auf den tardisen funktioniert musst du dem configure command noch das flag `--with-tardis` anfügen. Das ist aber **nur** für die Computer des ITET nötig:

```
./configure --with-tardis
```

3. Kompilieren: Um den source zu kompilieren musst du nur noch

```
make
```

ausführen. Falls du den source auf einem Computer mit mehreren Prozessoren kompilierst, kannst du den make Prozess auch parallelisieren (z.B. im ETL F11):

```
make -j3
```

Damit werden drei parallele make Prozesse gestartet.

4. Ein kleiner Bug: Damit du den code nun ausführen kannst, musst du in deinem home Verzeichnis noch einen neuen Ordner erstellen (/verb".orxonox"). In diesem Ordner wird Orxonox seine Settings speichern.

```
## wechseln ins home
cd
## Verzeichnis erstellen
mkdir .orxonox
```

5. Ausführen: Um das executable auszuführen musst du nur noch

```
./src/orxonox
```

eingeben. Orxonox wird einen Konfigurationsdialog starten. **Wichtig:** das Data-Directory musst du im Dialog auf jeden Fall bestimmen, in dieser Installation heisst es `~/orxonox/data`.

3 Kommunikation

Mailinglisten: Trage dich in die folgenden Mailinglisen ein. Orxonox hat drei verschiedene Listen: `announce` (Ankündigungen von Releases oder Events), `dev` (Entwickler unter sich) und `commit` (die commit messages). Geht dazu auf die folgenden Webpages, um euch anzumelden:

```
https://mail.datacore.ch/mailman/listinfo/orxonox-announce
https://mail.datacore.ch/mailman/listinfo/orxonox-dev
https://mail.datacore.ch/mailman/listinfo/orxonox-commit
```

Wir empfehlen dir sehr für die drei Mailinglisten einen Filter in deinem Mailclient zu definieren (z.B. kannst du für jede Liste ein Verzeichnis erstellen). Dies hilft dir den Überblick zu wahren. Vor allem die orxonox-commit Liste hat zwischendurch bis zu 20 Mails pro Tag.

IRC: Der *Internet Relay Chat* ist gut geeignet für kleinere (real-time) Diskussionen. Wir werden viel im *Orxonox IRC* anzutreffen sein, damit wir euch allen bei euren Problemen so schnell wie möglich helfen können. Auf den Tardis sind auch IRC Clients installiert, starte `xchat` und *joine* den Channel orxonox.

```
xchat irc://irc.datacore.ch
    ->   /join #orxonox
```

Forum: Wichtige Diskussionen werden auch im Forum geführt. Verzweig deinen Webbrowser auf die Seite:

```
https://forum.orxonox.net
```

Kreiere dir einen Benutzer-Login. Es ist wichtig, dass du das Forum möglich viel besuchst, damit das Forum möglichst interaktiv ist.

WIKI: Damit Informationen auch für die Leute zugänglich gemacht werden können, die nicht auf einer Mailingliste eingeschriben sind oder sich im IRC aufhalten, werden wichtige Informationen direkt ins Wiki geschrieben. Das Wiki ist der wichtigste Datenpool von Orxonox.

```
https://dev.orxonox.net/
```

Das Login fürs Wiki ist das gleiche wie du auch für das Subversion Repository erhalten wirst. Sobald du die Login-Daten von uns erhalten hast, kannst du ins Wiki einloggen. Suche den Bereich People unter Misc (unten auf der Seite), dort findest du deinen Namen, in einer komischen grauen Farbe und einem Fragezeichen am Ende (undefinierte Links nehmen im Trac-Wiki diese graue Farbe an). Wenn du auf deinen Namen klickst, dann kommst du auf eine neue leere Seite. Schreibe hier etwas kleines über dich, damit du die Wiki Syntax kennenlernst[2].

4 Selbstständiges Arbeiten zu Hause

1. Versuche Orxonox nun mal zu Hause zum laufen zu bringen, damit du auch von dort arbeiten kannst. Wir empfehlen dir Orxonox unter einer Linux Distribution aufzusetzen und zu entwicklen, da dies am einfachsten geht. Orxonox funktioniert aber auch unter Windows und MacOSX (V10.4).
   Falls du bei der Installation auf Probleme stossen solltest, komm doch einfach in den IRC Chat oder beschreibe das Problem im Forum.

2. Schau dir unter `https://dev.orxonox.net/wiki/PPS_TopicsSS2007` die Projekt Themen an und überlege dir, woran du gerne arbeiten würdest.

3. Installiere das 3D-Modeling Programm `Blender` auf deinem Computer zu Hause (auf den Tardisen ist Blender schon installiert). Du findest binaries oder source auf:

   ```
   http://www.blender.org
   ```

4. Wir werden dir ein 3D-Model per Mail schicken, dass du mit dem Blender öffnen solltest

[2]Wiki Syntax auch einsehbar unter: https://dev.orxonox.net/wiki/WikiFormatting