Verbesserung von Saros in unzuverlässigen Netzwerken mit hoher Latenz

Sebastian Bauch

Freie Universität Berlin
Institut für Informatik
AG Softwaretechnik

3. März 2011

1. Einleitung
2. Netzwerkebene
   - Ausgangssituation
   - Arbeiten an der Netzwerkebene
   - Zusammenfassung der Arbeit an der Netzwerkebene
3. Sandor - Testumgebung
   - Ausgangssituation
   - Arbeiten an der Testumgebung
   - Zusammenfassung der Arbeit an der Testumgebung
4. Zusammenfassung

Einleitung

- Saros ist eine Eclipse-Erweiterung zur Unterstützung verteilter Paar- und Side-by-Side-Programmierung
- Merkmale von Saros sind:
  - Schreiber-, Verfolgerrollen
  - Screensharing, VoIp
  - Versand von Daten an während einer Sitzung.
- Saros wird als Open-Source entwickelt und beruht auf mehrere Abschlußarbeiten.

Einleitung

Die Arbeit im Projekt teilte sich in folgende zwei Schwerpunkte

Netzwerkebene

- Analyse von bestehenden Netzproblemen.
- Überarbeitung der Netzwerkebene.
- Behebung von Defekten

Testumgebung

- Analyse von Defekten
- Überarbeitung der bestehenden Testfällen
- Übersetzung von Testfällen aus Testlink.

Ausgangssituation - Netzwerkebene

- Die Implementation der Netzwerkebene beruht seit der Entstehung von Saros auf der Bibliothek *Smack*
- Mein Kenntnisstand nach dem Lesen relevanter Arbeiten und ersten Testläufe war:

Kenntnis nach dem Lesen

- Ermittlung der IP-Adressen mit Hilfe eines STUN-Servers
- Jingle (TCP/UDP)
- SOCKS5
- In-Band Bytestreams

Kenntnis nach den Testläufen

- STUN funtkierte nicht
- Jingle(TCP) nur in Netzwerken ohne NAT
- Keine SOCKS5 Verbindungen
- In-Band Bytestream funktionierte immer.

Verbindungsarten während einer Sitzung

Verbindungsart	**Häufigkeit**	**NAT**
Jingle-TCP	oft	nein
Jingle-UDP	selten	ja
SOCKS5	nie	nein
In-Band Bytestream	oft	ja

Arbeiten an der Netzwerkebene

- Auf der Grundlage der Testläufe wurde eine Überarbeitung der Netzwerkebene beschlossen.
- Meine weitere Arbeit gliederte sich folgend:
  - Analyse der Defekte - betreffend Jingle
  - Reduzierung und Überarbeitung der Jingle-Klassen.
    - Entfernung von Jingle-UDP
    - Anpassung von Jingle-Klassen an die neuen SMACK-Schnittstellen.
- Behebung von Defekten in den *JUnit4*-Testfällen
- Behebung von Defekten, die in der Veröffentlichungswoche aufgetreten sind.

Zusammenfassung der Arbeit an der Netzwerkebene

- Die Jingle-Klassen - zur Zeit deaktiviert - sind überarbeitet und lesbarer geworden.
- Eine Aktivierung der Klassen ist möglich, wenn eine Lösung für das Ermitteln der IP-Adressen gefunden ist.
- Als Ergebnis der Bearbeitung verwendet Saros für die Verbindung SOCKS5 und In-Band Bytestream
- Offene Aufgaben sind:
  - Eine Trennung zwischen einer Projek-Datenverbindung und einer Kommunikationsverbindung.
    - Projek-Datenverbindung - Projektbezogene Daten und Aktivitätsnachrichten
    - Kommunikationsverbindung - Sprach- und Videoübertragung

Ausgangssituation - Testumgebung

- Die Testumgebung wurde von Sandor Szücs entworfen.
- Ziel: Automatisierung und Vereinfachung des Testprozesses
- Die Automatisierung basiert auf dem SWTBot, RMI-Schnittstellen und JUnit4-Testfällen.
- Die Testumgebung lag zu Beginn der Arbeit als Prototyp vor und auf lief auf einem Forschungsrechner auf einem VM-Server.
- Umzug der Testumgebung auf einen VM-Server im Institutscluster
- Nach dem Umzug funktionierten die alten Testfälle nicht mehr.

Arbeiten an der Testumgebung

- Die Schwerpunkte lagen in Fehleranalyse und Erstellung von neuen Testfällen.
- Die Schwierigkeit der Fehleranalyse lag in der Reproduzierbarkeit von Testergebnissen.
- Die häufigsten Probleme waren:
  - Fehler bei der Erfüllung von Bedinungungen
  - Betriebssystemabhängigkeit einiger SWTBot Methoden.
- Die wesentlichen Schwierigkeiten bei der Erstellung von Testfällen waren:
  - Fehlende Unterstützung von *native Dialogs*.
  - Die Ansteuerung von Eclipse-Fensterlementen.

Zusammenfassung der Arbeit an der Testumgebung

- Analyse von Defekten beim SWTBot
- Unterstützung bei der Weiterentwicklung der Testumgebung.
- Erstellung und Bearbeitung von Testfällen.
- Offene Aufgaben sind:
  - Das Aufsetzen weiterer VM - zur Simulation grösserer Entwicklergruppen.
  - Erstellen von Skripten zur Simulation von Netzwerken hinter einem NAT.
  - Erstellen von Skripten zur Sammlung und Darstellung von einzelnen Monitoraufnahmen.

Zusammenfassung der Gesamtarbeit

- Mit dieser Arbeit wurde erreicht:
  - Analyse und Behebung von Defekten der Netzwerkschicht.
  - Eine Überarbeitung der Netzwerkschicht.
  - Mithilfe bei der Weiterentwicklung der Testumgebung.
  - Weiterentwicklung und Neuentwurf von Testfällen für die Testumgebung.
- Aufgetretene Probleme waren:
  - Die Analyse und das Verstehen des Quelltextes, bedingt durch fehlende Dokumentation.
  - Die Analyse und Behebung von betriebssystemabhängigen Defekten an der Netzwerkschicht.
  - Die Behebung von Defekten in der Testumgebung.

Vielen Dank für Ihre Aufmerksamkeit.