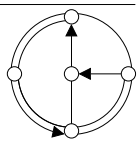
ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich



WS 2002/03 Prof. R. Wattenhofer / Keno Albrecht

Vernetzte Systeme
Übung 4

Ausgabe: **13. November 2002**
Abgabe: **25. November 2002**

Bitte schreiben Sie immer Ihre(n) Namen auf die Lösungsblätter.

Vorbemerkung zu den praktischen Übungen

In den praktischen Übungen werden Sie die Gelegenheit haben, einige der in der Vorlesung besprochenen Aspekte selbstständig auszuprobieren und umzusetzen.

Zur Implementierung sollten Sie eine der beiden Programmiersprachen Oberon oder Java verwenden:[1]

- Sie finden Oberon auf den Rechnern in den Räumen IFW-C31, HRS-F8/12 und HG-E27 (nur oben). Starten Sie den Rechner neu, und wählen Sie im Bootmanager den Eintrag für Oberon. Zusätzlich bietet der VIS eine Oberon-CD an, die Sie über uns beziehen können. Leider funktioniert die jetzige Version nur mit Netzwerkkarten der Firma 3COM und mit Chipsätzen von Realtek.
- Auf den meisten ETH-Rechnern (etwa *slab* oder *rif/raf*) sollte standardmässig ein Java-Compiler installiert sein. Bitte verwenden Sie die Version 1.2 oder höher.

Die Aufgabenstellungen geben Ihnen meist das Grundgerüst einer Klasse vor. Die dort enthaltenen leeren Prozedurrümpfe sollen von Ihnen ausgefüllt werden, um das jeweilige Programm zum Laufen zu bringen. Studieren und verstehen Sie aber unbedingt den kompletten Quellcode. Auch die von uns vorgegebenen Teile sind wichtig für das Verständnis und prüfungsrelevant! Als Abgabe schicken Sie bitte alle Dateien an Ihren Übungsleiter. Vermerken Sie in der Email alle Namen der beteiligten Studierenden (max. 3)!

1 Registrierung beim Server (24 Punkte)

In dieser Aufgabe werden Sie den ersten Teil eines einfachen Instant Messengers (IM) schreiben. Der Zweck eines IMs besteht darin, Textnachrichten direkt (Peer-to-Peer) zwischen verschiedenen Benutzern auszutauschen. Im Gegensatz zu einer Email, die an eine feste Adresse (etwa `ihr.name@student.ethz.ch`) versendet wird, können sich die Adressen eines IM-Clients allerdings ändern, da der Computer z.B. bei der Einwahl über ein Modem eine dynamische IP-Adresse des jeweiligen Internetdienstleisters zugeteilt bekommt. Doch woher weiss ein IM-Client, wohin eine Nachricht gesendet werden soll? Dazu beinhaltet ein IM-System typischerweise neben den Clients auch einen Server, bei dem sich die aktiven Clients mit ihrer aktuellen IP-Adresse registrieren müssen. Möchte ein Benutzer eine Nachricht an einen Bekannten senden, so muss erst die derzeitige Adresse des Empfängers beim Server ermittelt werden.

In dieser Aufgabe geht es darum, die Registrierung des Clients beim Server zu schreiben und die Liste aller aktiven Benutzer auszugeben. In den folgenden Übungen werden Sie den IM um

[1] Es besteht grundsätzlich auch die Möglichkeit, eine andere Programmiersprache einzusetzen. Beachten Sie aber bitte, dass wir unter Umständen nicht in der Lage sind, Ihnen bei allfälligen Problemen zur Seite zu stehen. Klären Sie den Einsatz in jedem Fall mit Ihrem Übungsleiter ab.

weitere Funktionen erweitern. Wir haben zwei IM-Server zum Testen aufgesetzt: Der eine läuft auf `dcg.ethz.ch` (Java) und der andere auf `bluebottle.ethz.ch` (Oberon). Der Einsatz von Java und Oberon soll Sie darauf aufmerksam machen, dass Ihre Entwicklung unabhängig von der Wahl der Programmiersprache und der Ausführungsumgebung ist, da wir ein fest definiertes Protokoll verwenden. Testen Sie Ihre Lösung auf jeden Fall mit beiden Servern!

Laden Sie von unserer Webseite das Grundgerüst für den **RegistrationClient** entsprechend Ihrer gewählten Programmiersprache herunter, und implementieren Sie die gekennzeichneten Stellen. Der Ablauf des Programms sieht vor, dass Sie eine Nachricht an den Server schicken, um sich dort zu registrieren. Anschliessend erhalten Sie eine Liste aller vor kurzem registrierten Benutzer zurück, die Sie ausgeben sollen. Drucken Sie die Liste bitte auch aus, und reichen Sie diese bei Ihrem Übungsleiter ein (max. 20 Einträge genügen).

Da wir den IM in Zukunft noch erweitern werden, ist es notwendig, bereits jetzt ein einfaches Nachrichtenformat einzuführen. Dieses besteht aus einem einzigen Byte, einer Typenkennung, das jeder Nachricht vorausgesendet wird. Anhand dieser kann die Gegenstelle erkennen, um was für eine Nachricht es sich handelt, und entsprechend darauf reagieren. Für diese Übung benötigen wir zwei Nachrichtentypen:

- `REGISTER` hat den Wert 0x01 und wird zur Registrierung an den Server geschickt.
- `USERLIST` hat den Wert 0x03 und wird an den Client geschickt, der darauf eine Liste aller aktiven Clients erhält.

Lösen Sie nun die folgenden Teilaufgaben mit Hilfe der aufgeführten Hinweise und der Kommentare im Quellcode:

a) (4 Punkte) Implementieren Sie die Prozedur `write16`.

Datenströme arbeiten auf unterster Ebene character- bzw. byteorientiert. In dieser Teilaufgabe sollen Sie eine 16-Bit-Zahl in zwei 8-Bit-Zahlen aufteilen. Beachten Sie dabei nur die unteren (less significant) beiden Bytes und keine Vorzeichen. Übertragen Sie erst das höherwertige (more significant) der beiden Bytes und danach das niederwertige. Haben Sie also zum Beispiel eine 32-Bit-Zahl in Hexadezimalschreibweise als 0xAABBCCDD gegeben, so extrahieren Sie die Bytes 0xCC und 0xDD und übertragen zuerst 0xCC und dann 0xDD.

b) (2 Punkte) Implementieren Sie die Prozedur `read16`.

Stellen Sie die 16-Bit-Zahl, die mit `write16` in zwei 8-Bit-Zahlen aufgeteilt wurde, wieder her.

c) (8 Punkte) Implementieren Sie die Prozedur `registerUser`.

Schicken Sie eine Nachricht des Typs `REGISTER` an den Server. Anschliessend schicken Sie unter Verwendung der bereits implementierten `write16`-Prozedur Ihren als Kommandozeilen-Parameter übergebenen Port. Danach folgt die Länge Ihres Namens als ein Byte (die Länge darf also 255 Bytes nicht überschreiten), und zum Schluss folgt Ihr Name selbst. Die Nachricht sieht also wie folgt aus:

Nachrichtentyp	Nachrichtenformat (Anzahl der Bytes)
REGISTER	Typ(1) — Port(2) — Länge des Namens(1) — Name(max. 255)

d) (10 Punkte) Implementieren Sie die Prozedur `receiveUserList`.

Sie empfangen eine Nachricht des Typs `USERLIST` vom Server. Als nächstes folgt die Anzahl der aktiven Benutzer und dann **für jeden Benutzer** zuerst die Länge seines Namens als ein Byte, der Name selbst und schliesslich die IP-Adresse und der Port, über welche dieser anzusprechen ist. Beachten Sie, dass die IP-Adresse wie folgt übertragen wird: Lautet die Adresse a.b.c.d, so wird zuerst a, gefolgt von b, c, d gesendet. Denken Sie auch daran, dass IP-Adressen Zahlen aus dem Bereich 0-255 enthalten, ein Byte aber u.U. (Java) vorzeichenbehaftet ist! Geben Sie die Benutzerliste aus, und reichen Sie einen Ausdruck von max. 20 Einträgen bei Ihrem Übungsleiter ein. Kompakt sieht das Nachrichtenformat wie folgt aus:

Nachrichtentyp	Nachrichtenformat (Anzahl der Bytes)
USERLIST	Typ(1) — Anzahl Benutzer(2) — { Länge des Namens(1) — Name (max. 255) — IP(4) — Port(2) }*