Technische Universität Braunschweig
Institut für Betriebssysteme und Rechnerverbund
Kommunikation und Multimedia
Prof. Dr. L. Wolf

Praktikum Kommunikationssysteme im SS06

– Aufgabenbeschreibung –

Betreuer: Zefir Kurtisi, Habib-ur-Rehman, Alexej Beresnev

Aufgabe 1: IBR-Chat (IBRC)

Termine

Konzept	27.04.06
Implementation	25.05.06

1 Aufgabenstellung

Es ist ein einfaches Chat-Programm namens IBRC *(IBR C*hat) zu entwickeln, welches sich an das bekannte IRC *(I*nternet *R*elay *C*hat) anlehnt. Das IBRC ist allerdings deutlich einfacher gestaltet als das originale IRC und weist ihm gegenüber einige Modifikationen auf.

IRC stellt ein System zur Durchführung von Textkonferenzen dar und bietet dem Benutzer die Möglichkeit, sich mit anderen Personen via Rechner zu "unterhalten", d. h. geschriebenen Text auszutauschen. Hierzu existieren verschiedene "Diskussionsrunden" (Kanäle genannt), in denen Teilnehmer über bestimmte Themen ihrer Meinung freien Lauf lassen können. Die Anzahl der Teilnehmer pro Diskussionsrunde wird nur durch die verfügbaren Ressourcen begrenzt.
IRC ist Client-Server-basiert, d. h. das Programm besteht aus zwei Teilen, einem Client und einem Server. Es wird im Detail in [1] beschrieben.

2 Grundlagen

Im Folgenden werden kurz die grundlegenden Konzepte des IBRC erläutert, bevor die zu bearbeitende Aufgaben im Detail vorgestellt werden.

2.1 Clients

Als Clients werden diejenigen Rechner bezeichnet, an denen ein Teilnehmer der Diskussionsrunde sitzt, d. h. sie bilden die eigentliche Talkrunde. Jeder Client unterhält eine Verbindung zu genau einem Server. Zu diesem werden alle vom Client eingegebenen Daten gesendet. Der Server muss dann über deren weitere Verwendung bzw. deren Verarbeitung entscheiden. Identifiziert werden Clients durch den Namen des entsprechenden Rechners (Hostname). Die Teilnehmer werden normalerweise nicht durch ihre wahren Namen identifiziert, sondern durch sogenannte *nicknames* (Spitznamen), welche sie sich frei aussuchen können. Diese Spitznamen dürfen max. neun Zeichen lang sein und müssen aus Buchstaben oder Ziffern bestehen.

2.2 Server

Server erfüllen im wesentlichen zwei Aufgaben. Zum einen nehmen sie alle Daten von den angeschlossenen Clients entgegen, um diese weiterzuleiten oder zu bearbeiten. Die Teilnehmer (in Form von Clients) kommunizieren also niemals direkt miteinander, sondern immer über einen oder mehrere Server.
Zum anderen muss jeder Server Informationen über die Clients im Netzwerk verwalten. Dies ist u. a. notwendig, um die Kontrolle über alle existierenden Teilnehmer bzw. Kanäle zu behalten (zur Beschreibung der Kanäle s. nächster Abschnitt). Es muss beispielsweise verhindert werden, dass ein Teilnehmer mehr als einem Kanal gleichzeitig angehört oder dass zwei Kanäle dieselbe Bezeichnung erhalten.
Weitere Einzelheiten zum Ablauf der Kommunkation werden im Abschnitt 4 gegeben.

2.3 Kanäle

Unter einem Kanal kann man sich eine Gruppe von Teilnehmern vorstellen, die untereinander Nachrichten austauschen. Die Nachrichten eines Teilnehmers werden an alle Teilnehmer, die diesem Kanal angehören, verschickt. Ein Teilnehmer kann nicht mehreren Kanälen gleichzeitig angehören. Wenn sich der allererste Teilnehmer einem Kanal anschließt, d.h. wenn der Kanal noch nicht existiert, so wird dieser erzeugt. Wichtig ist, dass sich der Zustand eines Kanals dynamisch ändern kann, etwa wenn neue Teilnehmer hinzukommen. Diese Änderung muss vom entsprechenden Server bemerkt und die notwendigen Informationen an alle weiteren Server im Netz gesendet werden.

Kanäle werden (etwas abweichend vom originalen IRC) durch Kurzbezeichnungen identifiziert, z.B. "Fussball-WM". Diese Bezeichner sind max. 12 Zeichen lang und müssen eindeutig sein, d.h. es darf keine zwei Kanäle mit derselben Bezeichnung geben. Die Bezeichner werden bei allen Befehlen verwendet, um einen Kanal eindeutig zu identifizieren. Um beispielsweise dem Kanal "Fussball-WM" beizutreten, muss ein Teilnehmer den Befehl `JOIN Fussball-WM` eingeben.
Darüber hinaus existiert für jeden Kanal eine ausführlichere Beschreibung (Thema) des diskutierten Inhalts. Das Thema des Kanals "Fussball-WM" könnte z.B. lauten: "Wissenswertes zu und rund ums Runde".
Das Thema kann ausschließlich von demjenigen Teilnehmer, der den Kanal erzeugt hat, bestimmt werden, wobei auch nachträgliche Änderungen möglich sind. Dieses Recht auf Themenänderung ist nicht auf andere Teilnehmer übertragbar.

3 Nachrichten und Befehle

Bei den Eingaben, die der Benutzer des IBRC vornimmt, ist prinzipiell zwischen Nachrichten und Befehlen zu unterscheiden. Nachrichten entsprechen geschriebenen Worten, die an alle Teilnehmer des zugehörigen Kanals gesendet werden. Sie sind zeilenbasiert, d.h. nach Betätigen der Return-Taste wird die aktuelle Zeile an den angeschlossenen Server gesendet, der sie an die entsprechenden Clients oder ggf. an weitere Server weiterleitet.
Befehle dienen zur Steuerung der Kommunikation für die Teilnehmer, z.B. um einem Kanal beizutreten, Informationen über andere Teilnehmer einzuholen etc. Sie werden durch einen vorangestellten Schrägstrich (`/`) gekennzeichnet, wobei die Groß-/Kleinschreibung keine Rolle spielt.
Alle Befehle haben die Form `<Befehl> [ <Parameter1> [ <Parameter2> ] ].`
Einen Überblick über alle Befehle des IBRC gibt Tabelle 1.

Befehl	Beschreibung
JOIN	Einem Kanal beitreten
LEAVE	Aktuellen Kanal verlassen
NICK	Eigenen Nickname ändern
LIST	Alle existierenden Kanäle auflisten
GETTOPIC	Beschreibung eines Kanals ausgeben
SETTOPIC	Beschreibung des aktuellen Kanals setzen
PRIVMSG	(Private) Nachricht an einzelnen Teilnehmer senden
QUIT	IBRC-Tool verlassen

Tab. 1: IBRC–Befehle

4 Kommunikation

Im IBRC sind zwei Teilnehmer (Clients) niemals direkt miteinander verbunden, sondern immer über einen oder mehrere Server. Ein Client ist immer mit genau einem Server verbunden. Die Server haben dafür Sorge zu tragen, dass eine Nachricht, welche in einem Kanal versendet wird, an alle Teilnehmer dieses Kanals weitergeleitet wird. Wichtig ist hierbei, dass vom Client die Nachricht nur *einmal* an den angeschlossenen Server geschickt wird (auch wenn an diesem noch weitere, dem Kanal angehörende, Clients angeschlossen sind). Der Server entscheidet, an welche Clients und ggf. an welche weiteren Server die Nachricht weiterzuleiten ist. Es ist selbstverständlich, dass die Nachricht nur an diejenigen Server gesendet wird, die die Nachricht benötigen, um sie ihrerseits an weitere, dem entsprechenden Kanal angehörende, Clients zu senden.

Darüber hinaus obliegt den Servern die Bearbeitung der Befehle, die die Teilnehmer eingeben können. Hierzu ist eine Speicherung der Daten über alle existierenden Teilnehmer, Kanäle etc. notwendig. Wichtig ist, dass *alle* Server im Netz ihre Daten synchronisieren, d.h. dass alle Server stets über die gleichen Daten verfügen. Es ist klar, dass hierzu eine Kommunikation der Server untereinander notwendig ist. Im originalen IRC werden hierfür eigene Server-Befehle definiert. Für dieses Praktikum ist nichts derartiges vorgeschrieben, es bleibt den Teilnehmern überlassen, wie sie die Serverkommunikation realisieren.

4.1 Topologie des Chat–Netzwerks

Das aus Clients und Servern bestehende IBRC-Netzwerk soll eine Baumstruktur aufweisen und ist somit zyklenfrei. Die genaue Topologie kann von den Teilnehmern frei gewählt werden. Es muss allerdings möglich sein, in das Netzwerk mindestens fünf Server und 10 Clients integrieren zu können. Die geforderte Zyklenfreiheit kann dadurch sichergestellt werden, dass sich neue Server nur an *einen* bereits bestehenden Server anschließen dürfen.
Wenn die Servertopologie feststeht, dürfen sich Clients an beliebige Server anschließen, wobei ein Client natürlich nur an *einen* Server angeschlossen sein darf. Nachdem der erste Client in das Netzwerk integriert wurde, ist die Topologie für die Server fest, d.h. es dürfen keine Verbindungen zwischen Servern neu gebildet oder entfernt werden. Das Einfügen neuer Server ist ebenfalls nicht mehr erlaubt. Es kann davon ausgegangen werden, dass niemals mehr als 10 Server und 20 Clients vorhanden sind (d.h. es können Arrays für die Speicherung verwendet werden).

4.2 Kommunikationsformen

Im IBRC (und ebenfalls im IRC) existieren zwei verschiedene Kommunikationsarten: *one-to-many* und *one-to-one*.

one-to-many: Diese Kommunikationsform entspricht der normalen kanalbasierten Form, d.h. die Nachrichten werden an alle Teilnehmer eines Kanals gesendet.

one-to-one: Es findet eine Kommunikation innerhalb eines Kanals zwischen genau zwei Teilnehmern statt. Alle anderen bemerken von dieser "verdeckten" Kommunikation nichts. Sie kann benutzt werden, um Rückfragen zu halten, über andere Teilnehmer zu lästern etc.
Für diese Art der Kommunikation existert der `PRIVMSG`-Befehl.

5 Aufgabe

5.1 Protokollspezifikation

Vor der Programmierung soll jede Gruppe ein Konzept erstellen, welches das Protokoll für die Kommunikation sowie einen groben Überblick über den späteren Programmablauf und der benötigten Daten enthalten soll. Ein vollständiges und gut durchdachtes Konzept ist essentiell für eine erfolgreiche Implementierung, dafür ist eine Woche Bearbeitungszeit vorgesehen. Es muss schriftlich ausgearbeitet den Betreuern vorgelegt und anschließend diskutiert werden.

Nach bisherigen Erfahrungen ist es sinnvoll, mindestens folgende Punkte im Konzept zu berücksichtigen:

Protokoll: Form (Klartext oder binär), Arten (Client-Server und Server-Server oder einheitlich), Trennzeichen

Datenstrukturen: welche Datensätze werden wo benötigt, wann und wie werden Datensätze aktualisiert

Verbindungen: welche Verbindungen werden wie aufgebaut, wie und wann werden sie wieder abgebaut

Routing: welche Nachrichten werden wann wohin gesendet, welche Daten werden für das Routing benötigt, wie wird das Routen realisiert

Fehlerbehandlung: wer Prüft wann auf Gültigkeit, was passiert im Fehlerfall, wie wird "Gleichzeitigkeit" behandelt, wie sollen Fehler kommuniziert werden

5.2 Programmgerüst

Für den IBRC wird kein Programmgerüst vorgegeben, da die Funktionalität vollständig im Konsolefenster nachgewiesen werden kann. Eine Entwicklung mit Hilfe von `Makefiles` empfiehlt sich hier, allerdings ist jeder Gruppe überlassen, andere Entwicklungsumgebungen zu benutzen (`XCode`, `Eclipse` etc.). Genauso kann jede Gruppe mit Interesse und Zeit eine grafische Benutzerschnittstelle implementieren. Für die Abnahme der Aufgabe ist jedoch allein der Nachweis der geforderten Funktionsweise entscheidend.

5.3 Implementation

In dieser Aufgabe soll nun ein kleines IBRC–Netzwerk mit mehreren Servern implementiert werden. Das Netz muss mindestens fünf Server und 10 Clients verwalten können. Die geforderte Topologie wurde bereits in Abschnitt 4.1 beschrieben.

Die Server nehmen Daten von den angeschlossenen Clients entgegen und zwar mit Hilfe des zuvor spezifizierten Protokolls. Handelt es sich um Nachrichten, so müssen diese an den oder die entsprechenden Clients gesendet werden. Ist ein Client nicht direkt vom Server erreichbar, muss die

Nachricht "geroutet" werden, d.h. an denjenigen Server geschickt werden, von dem aus der Client erreichbar ist. Da die verwendete Topologie zyklenfrei ist, sind diese Routingentscheidungen eindeutig. Es ist darauf zu achten, dass nur diejenigen Server, die die Nachricht wirklich benötigen, sie auch erhalten. Ein "Broadcast" der Nachrichten von jedem Server an alle weiteren, um sich das Routing zu sparen, ist nicht zulässig. Als Transportprotokoll ist TCP zu benutzen.

Als Server bzw. Clients können beliebige Rechner aus dem CIP-Pool benutzt werden. Zur Kontrolle, welche Nachricht ein Server erhält und an wen er sie weiterleitet, ist auf jedem Server eine einfache Ausgabe zu implementieren, die zu jeder empfangenen Nachricht die Hostnamen des Absenders sowie des Adressaten enthält.

5.4 Zu implementierende Befehle

Alle in der nachfolgenden Beschreibung aufgeführten Befehle sind zu implementieren. Es sei noch einmal daran erinnert, dass Befehlen bei der Eingabe ein Schrägstrich voranzustellen ist. Leerzeichen an Anfang, in der Mitte (zwischen einzelnen Argumenten) oder am Ende eines Kommandos sollen ignoriert werden.

JOIN $< Kanal >$
: Mit Hilfe dieses Befehls tritt ein Teilnehmer einem Kanal bei. *Kanal* ist die Kurzbezeichnung des Kanals (z.B. "Fussball-WM"). Existiert der Kanal noch nicht, so wird er neu erzeugt. Ist der Teilnehmer bereits einem anderen Kanal angeschlossen, so wird der Befehl ignoriert und es erfolgt die Ausgabe einer Fehlermeldung. Der Teilnehmer kann nur chatten, wenn er einem Kanal angeschlossen ist.

LEAVE
: Hiermit wird der aktuelle Kanal verlassen. Gehört der Teilnehmer keinem Kanal an, so erfolgt die Ausgabe einer Fehlermeldung. Falls der Teilnehmer das letzte Mitglied des Kanals ist, dann wird der Kanal gelöscht.

NICK $< Name >$
: Setzt den Spitzamen (nickname) des Benutzers, unter dem dieser in Erscheinung tritt. Diese Namen sind frei wählbar, allerdings dürfen keine zwei Teilnehmer den gleichen Namen erhalten. Nicknames sind max. neun Zeichen lang und müssen aus Buchstaben oder Ziffern bestehen. Fehlerhafte Eingaben werden mit einer Fehlermeldung quittiert.

LIST
: Listet alle im IBRC existierenden Kanäle mit ihren Kurzbezeichnungen auf.

GETTOPIC $< Kanal >$
: Gibt die ausführliche Beschreibung (Thema) zum angegebenen Kanal aus. Existiert der Kanal nicht, wird eine Fehlermeldung ausgegeben.

SETTOPIC $< Thema >$
: Setzt das Thema des aktuellen Kanals. Dieser Befehl kann nur von dem Teilnehmer, der diesen Kanal erzeugt hat, ausgeführt werden. Der Versuch anderer Teilnehmer, diesen Befehl auszuführen, wird mit einer entsprechenden Fehlermeldung quittiert.

PRIVMSG $< Name > < Nachricht >$
: Sendet eine private Nachricht an den unter *Name* angegebenen Teilnehmer. Dieser Teilnehmer muss demselben Kanal angehören, dem auch der Absender angehört. Ist dies nicht der Fall oder existiert der angegebene Teilnehmer nicht, so erfolgt die Ausgabe einer Fehlermeldung.

QUIT

Falls ein Teilnehmer der Meinung ist, genügend gequatscht zu haben, kann er mit diesem Befehl den IBRC–Client verlassen. Gehört er noch einem Kanal an, wird seine Mitgliedschaft in diesem automatisch beendet.

5.5 Abnahme

Die Aufgabe gilt als abgenommen, wenn das Konzept eingereicht und die darauf basierende Implementation die genannte Funktionsfähigkeit nachweist. Der Quellcode wird vorab zur Prüfung per Email an die Betreuer gesendet. Während der betreuten Zeiten wird dann das Programm auf den Rechnern im G40 demonstriert und besprochen. Bei dieser Abnahme müssen alle Gruppenmitglieder anwesend sein, um den eigenen Programmteil erläutern und Fragen zur Implementation beantworten zu können.

Literatur

[1] J. Oikarinen and D. Reed. Internet Relay Chat Protocol. RFC 1459, Merit Network Inc., NASA, May 1993.