Praktikum:
Entwicklung von Kommunikationssystemen (SoSe 2000)

Prof. Dr. Reinhard Gotzhein, Philipp Schaible

Aufgabe 6 (zwei Wochen)

Ziel dieser Aufgabe ist es, die Client-Komponente zu implementieren und zusammen mit dem Server auf einer Zielplattform (rn5, rn6 Rechner der AG “Rechnernetze“) zum Laufen zu bringen. Anders als der Server wird die Client-Komponente per Hand (in JAVA) implementiert.

Die zuvor erstellte SDL-Spezifikation ist eine Design-Beschreibung und legt das Verhalten der Komponente fest. Die Implementierung ist nur dann eine “gültige Implementierung“, wenn das implementierte Verhalten der Design-Spezifikation entspricht. Dies erreicht man durch eine systematische Implementierung und auch durch das anschließende Testen.

Es gibt mehrere unterschiedliche Methoden, eine Implementierung ausgehend aus einer SDL-Spezifikation zu erstellen. Die wohl bekanntesten sind *Server-Modell* und *Activity-Thread* Implementierungen. Beide werden bei der Implementierung des Clients angewendet.

Bei dem *Server-Modell* werden einzelne SDL-Prozesse als aktive Komponenten implementiert, welche die Aufträge (Signale) aus der ebenso implementierten Warteschlange entnehmen und entsprechend verarbeiten. Vorteil dieser Implementierung besteht darin, daß eine direkte Abbildung (Spezifikation - Implementierung) existiert, was den Nachweis der Korrektheit der Implementierung und eine automatische Codegenerierung einfach macht. Z.B. der von SDT erzeugte C-Code ist eine Server-Implementierung. Nachteil dieser Implementierung besteht darin, daß es u.u. sehr ineffizient ist, denn die Signale werden ständig kopiert (von einer Warteschlange in die andere) und es muß ein Scheduling-Verfahren implementiert werden. Ganz anders die *Activity-Thread* Implementierung. Hier werden die einzelnen SDL-Prozesse überwiegend als passive Komponenten implementiert, welche erst mit dem Signal (z.B. durch einen Methodenaufruf übermittelt) aktiviert werden. Nachteil dieser Implementierung ist eine komplexere Abbildung (Spezifikation - Implementierung) und auch die Tatsache, daß nicht alle SDL-Spezifikationen auf diese Art implementiert werden können.

Die auf diese Implementierungsmodelle zugeschneiderte Architektur der Client-Komponente wird vorgegeben. Die Objekte der GUI-Anwendung, welche ebenso zur Verfügung gestellt werden und die Objekte des Kommunikationssystems werden in demselben Adressraum implementiert, was eine direkte Kommunikation per Methodenaufruf zwischen diesen Komponenten möglich macht.

a) Implementiert zuerst den “öffentlichen” Teil des Clients (d.h. UDP-Kommunikation mit dem Server und die Anbindung an die entsprechende Objekte der GUI-Applikation). Verwendet zur Implementierung das Server-Modell. Die programmiertechnischen Fragen sind der FAQ Liste, welche parallel zu der Aufgabe erstellt wird, zu entnehmen. Die Beschreibung der einzelner Objekte befindet sich unter *rn4:~eksys/pub/app/doc*. Den JAVA-Gerüst selbst findet man unter *rn4:~eksys/pub/app/*. In *rn4:~eksys/pub/sdt/* findet man SDT-Bibliotheken zum Portieren des Servers auf die Rechner rn5 und rn6.

b) Implementiert den “privaten” Teil des Clients (d.h. TCP- Kommunikation unter einander und die Anbindung an die entsprechende Objekte der GUI-Applikation). Verwendet zur Implementierung das Activity-Thread-Modell.

c) Der “private” Teil des Clients soll explizit getestet werden. Wählt ein Verfahren aus der Vorlesung “Rechnernetze“ aus und testet damit die entsprechende Kommunikationskomponente. Begründet die Wahl.

Praktikum:
Entwicklung von Kommunikationssystemen (SoSe 2000)

Prof. Dr. Reinhard Gotzhein, Philipp Schaible

Aufgabe 6 (zwei Wochen)

Ziel dieser Aufgabe ist es, die Client-Komponente zu implementieren und zusammen mit dem Server auf einer Zielplattform (rn5, rn6 Rechner der AG “Rechnernetze“) zum Laufen zu bringen. Anders als der Server wird die Client-Komponente per Hand (in JAVA) implementiert.

Die zuvor erstellte SDL-Spezifikation ist eine Design-Beschreibung und legt das Verhalten der Komponente fest. Die Implementierung ist nur dann eine “gültige Implementierung“, wenn das implementierte Verhalten der Design-Spezifikation entspricht. Dies erreicht man durch eine systematische Implementierung und auch durch das anschließende Testen.

Es gibt mehrere unterschiedliche Methoden, eine Implementierung ausgehend aus einer SDL-Spezifikation zu erstellen. Die wohl bekanntesten sind *Server-Modell* und *Activity-Thread* Implementierungen. Beide werden bei der Implementierung des Clients angewendet.

Bei dem *Server-Modell* werden einzelne SDL-Prozesse als aktive Komponenten implementiert, welche die Aufträge (Signale) aus der ebenso implementierten Warteschlange entnehmen und entsprechend verarbeiten. Vorteil dieser Implementierung besteht darin, daß eine direkte Abbildung (Spezifikation - Implementierung) existiert, was den Nachweis der Korrektheit der Implementierung und eine automatische Codegenerierung einfach macht. Z.B. der von SDT erzeugte C-Code ist eine Server-Implementierung. Nachteil dieser Implementierung besteht darin, daß es u.u. sehr ineffizient ist, denn die Signale werden ständig kopiert (von einer Warteschlange in die andere) und es muß ein Scheduling-Verfahren implementiert werden. Ganz anders die *Activity-Thread* Implementierung. Hier werden die einzelnen SDL-Prozesse überwiegend als passive Komponenten implementiert, welche erst mit dem Signal (z.B. durch einen Methodenaufruf übermittelt) aktiviert werden. Nachteil dieser Implementierung ist eine komplexere Abbildung (Spezifikation - Implementierung) und auch die Tatsache, daß nicht alle SDL-Spezifikationen auf diese Art implementiert werden können.

Die auf diese Implementierungsmodelle zugeschneiderte Architektur der Client-Komponente wird vorgegeben. Die Objekte der GUI-Anwendung, welche ebenso zur Verfügung gestellt werden und die Objekte des Kommunikationssystems werden in demselben Adressraum implementiert, was eine direkte Kommunikation per Methodenaufruf zwischen diesen Komponenten möglich macht.

a) Implementiert zuerst den “öffentlichen” Teil des Clients (d.h. UDP-Kommunikation mit dem Server und die Anbindung an die entsprechende Objekte der GUI-Applikation). Verwendet zur Implementierung das Server-Modell. Die programmiertechnischen Fragen sind der FAQ Liste, welche parallel zu der Aufgabe erstellt wird, zu entnehmen. Die Beschreibung der einzelner Objekte befindet sich unter *rn4:~eksys/pub/app/doc*. Den JAVA-Gerüst selbst findet man unter *rn4:~eksys/pub/app/*. In *rn4:~eksys/pub/sdt/* findet man SDT-Bibliotheken zum Portieren des Servers auf die Rechner rn5 und rn6.

b) Implementiert den “privaten” Teil des Clients (d.h. TCP- Kommunikation unter einander und die Anbindung an die entsprechende Objekte der GUI-Applikation). Verwendet zur Implementierung das Activity-Thread-Modell.

c) Der “private” Teil des Clients soll explizit getestet werden. Wählt ein Verfahren aus der Vorlesung “Rechnernetze“ aus und testet damit die entsprechende Kommunikationskomponente. Begründet die Wahl.