Projektauftrag: Eine Lern-Software für das Labor Netze und Systeme (NuS)

SS 2016 Prof. Dr. K. Hoffmann, Prof. Dr. A. Aßmuth

Problemstellung

Sinn und Zweck der Software / Hintergrund

Die Lehre im Gebiet „Rechnernetze“[1] stellt folgende Themen[2] in den Mittelpunkt:

- Komponenten und Strukturen: Hardwareeinrichtungen und Netzzugriffstechniken, Schichtenmodelle, Netztopologien.
- Dienste und Protokolle: Konfiguration von Diensten und Protokollen, Benutzer- und Ressourcenverwaltung.
- Vermittlung und Verbindung: TCP/IP-Vermittlung, Routing.
- IT-Sicherheit: Programme und Filter, kryptographische Protokolle

Es bedeutet eine wertvolle Unterstützung der Lehr- und Lernaktivitäten, wenn mit Rechnernetzkonfigurationen auf möglichst einfache Weise experimentiert werden kann: eine Software, mit der grundlegende Kommunikationsszenarien in Rechnernetzen simuliert und analysiert werden können, erscheint hierfür wünschenswert.

Im Praktikum zum Modul „Rechnernetze“ gibt es z.B. eine Reihe von Aufgaben, bei denen die Hardwarestruktur eines Rechnernetzes zusammen mit gewissen Anforderungen zur gegenseitigen Erreichbarkeit bzw. Nichterreichbarkeit zwischen Rechnern oder Teilnetzen vorgegeben wird, so kann z.B. gefordert sein, dass:

- jeder Rechner im Subnetz A jeden Rechner im Subnetz B erreichen kann
- bestimmte Rechner im Subnetz C keinen Zugang ins Internet haben oder umgekehrt von dort aus nicht erreichbar sein dürfen.

Zur Lösung derartiger Aufgaben müssen Praktikumsteilnehmer Weiterleitungstabellen geeignet konfigurieren. Sie brauchen in diesem Zusammenhang die Möglichkeit, die Funktion des Rechnernetzes mit ihrer Konfiguration zu erproben und im Falle von Problemen benötigen sie Unterstützung bei der Fehlersuche. Für Praktikumsbetreuer ist es in diesem Zusammenhang wünschenswert, zur Überprüfung der in den einzelnen Aufgaben gestellten Anforderungen Testszenarien definieren und die Lösungen der Praktikumsteilnehmer mit diesen Szenarien „per Knopfdruck“ durchspielen und auswerten zu können.

Im Internet gibt es die Software „Filius“, in der ein Teil der soeben angesprochenen Funktionalität recht gut realisiert ist. Für eine Verwendung in der Lehre an der OTH-Amberg-Weiden bleiben allerdings in zentralen Punkten Wünsche offen; diese werden im nächsten Abschnitt dargelegt.

[1] in den Studiengängen „Angewandte Informatik (AI)“, „Elektro- und Informationstechnik (EI)“ der OTH-Amberg-Weiden gibt es hierzu ein gleichnamiges Modul.

[2] zitiert aus dem gemeinsamen Modulhandbuch für die Studiengänge AI, EI der OTH Amberg-Weiden

Systemüberblick

Funktionalität
Gewünscht ist eine Software, die folgende Hauptanforderungen erfüllt:

- Aufbau der Rechnernetzstruktur aus Hardwareknoten und -verbindungen mit Hilfe eines Editors mit grafischer Benutzeroberfläche (ähnlich wie in Filius)
- Konfiguration eines Rechnernetzes, insbesondere der Weiterleitungstabellen; hierbei soll – anders als in Filius - jeder Rechner im Netz routingfähig[3] und volle Funktionalität beim Subnetting gegeben sein.
- Simulation und Analyse einer Rechnernetzkonfiguration:
    - Möglichkeit der Verfolgung von Paketen und Rückpaketen: Verfolgung des Wegs, den ein bestimmtes Paket durch das Netz nimmt; im Problemfall feststellen, wo es hängen bleibt.
    - Visualisierung des Protokollstapels zu einem ausgewählten Rechnerknoten: Visualisierung der Schichten und deren Aktivität während der Simulation.
- Definition von Testszenarien
- Durchspielen dieser Testszenarien mit Lösungsvorschlägen zu Praktikumsaufgaben „per Knopfdruck“ (wie im vorherigen Abschnitt angesprochen).

Software-Architektur
Obwohl der Software-Entwurf in Filius durchaus recht gut strukturiert wirkt, erscheint es dennoch nicht ganz leicht, z.B. eine neue Schicht in den Protokollstapel einzuziehen, um etwa die Datenübertragung abhörsicher zu machen. Für die zu entwickelnde Software ist daher gewünscht, dass die Architektur des Protokollstapels derartige spätere Erweiterungen auf möglichst einfache Weise unterstützt – „einfach“ heißt konkreter: ohne (oder nur mit minimaler) Änderung an vorhandenem Code, stattdessen nur Neuschreiben von Code für zusätzliche Software-Komponenten. Dazu muss die Architektur sehr klar konzipiert sein: in Abweichung von der etwas „unsauberen TCP/IP-Realität“ (Vgl. S. 79 im Lehrbuch[4] von Tanenbaum und Wetherall) ist hierzu eine deutlichere Orientierung an Konzepten wie „Schicht“, „Dienst“, „Protokoll“ und damit verbundenen Prinzipien wünschenswert, wie z.B.:

- eine Schicht hat eine klar definierte Zuständigkeit, sie erbringt Dienste für die darüber liegende Schicht
- die Art und Weise, wie eine bestimmte Schicht ihre Dienste für die darüber liegende erbringt, soll vollkommen gekapselt sein.

Idealerweise orientiert sich der Software-Entwurf an dem Fünf-Schichten-Modell, das im zuvor erwähnten Buch verwendet wird (siehe dort S. 74).

Implementierung
Das gewünschte Programm soll als Stand-Alone-Anwendung auf den Rechnern des Labors „Netze und Systeme“ lauffähig sein. Dazu ist die Unabhängigkeit von Microsoft-Windows erforderlich. Deswegen wird eine Implementierung in Java oder eine .NET-basierte und auf Mono™ lauffähige Implementierung in C# gewünscht.

Abzuliefernde Arbeitsergebnisse (Produktumfang)

Siehe das Kapitel „Lieferumfang“ im Anforderungsdokument.

[3] In Filius sind hierzu nur sog. Vermittlungsrechner („Router“) fähig, bei allen anderen Rechnern („Notebook“, „Server“) gibt es keine Möglichkeit, eine Weiterleitungstabelle zu konfigurieren, sondern man kann für das Routing einzig und allein die Adresse des Default-Gateways festlegen. In der Lehrveranstaltung „Rechnernetze“ gibt es jedoch Aufgabenbeispiele, die wegen dieser Einschränkung nicht mit Filius bearbeitet werden können.
[4] Siehe Literaturempfehlung zum Modul „Rechnernetze“