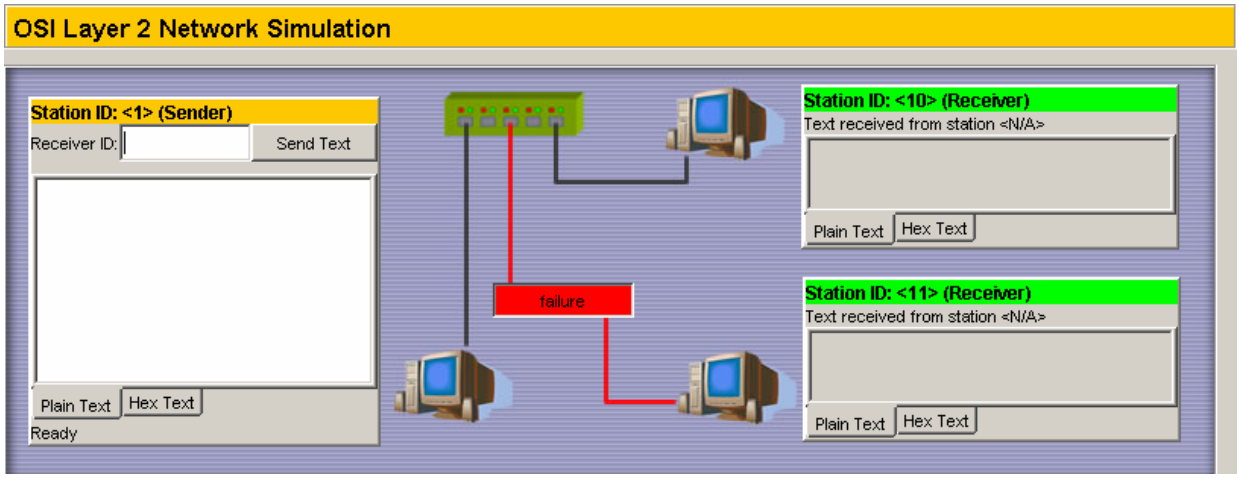
Ü Netzwerke und verteilte Systeme	Übung #3	WS 2004/2005
Name:	Matr-Nr:	
Abgabe: 09.11.2004		Gruppe:

Einführende Informationen zur Übung

Für die folgenden Übungsaufgaben wird ein Java-Framework verwendet, mit welchem eine konkrete Netzwerk-Infrastruktur nachgebildet wird. Das Framework selbst ist nicht an spezielle Protokolle, Geräte oder ähnliches gebunden und ermöglicht die Simulation von Datenübertragung mittels einer ISO/OSI 7 Schichten-Infrastruktur. Auf jeder Schicht können eine Reihe so genannter „Filter“ eingehängt werden, die die Aufgaben der jeweiligen Schicht übernehmen. In dieser Übung liegt der Schwerpunkt auf Layer 2, also dem Data-Link-Layer.

Vorgegebenes Szenario

In dem vorgegebenem Übungs-Szenario existieren drei Rechner, welche mittels Hub (Layer 1) in einem Broadcast-Netz miteinander verbunden sind und somit ein Netzwerk bilden. Die Rechner (stations) haben eine eindeutige Layer 2 Identifikationsnummer. Der Rechner links, auf welchem auf Layer 7 die „Sendeapplikation“ läuft, hat die ID 1; mittels dieser Applikation können gezielt Nachrichten an andere Rechner versendet werden. Auf den beiden anderen Stationen läuft jeweils eine Empfangsapplikation, welche Text anzeigt, sofern dieser bis zur Applikation durchgereicht wird. Die IDs dieser Rechner sind 10 bzw. 11.
Nun ergibt es sich, dass Rechner 11 mittels fehlerhaften Kabels mit dem Hub verbunden ist, und sich auf dem Weg vom Hub zum Rechner Fehler einschleichen können. In der konkreten Implementierung kippt immer genau 1 Bit (das Fehlverhalten des Kabels kann zu Testzwecken mit einem Klick auf den Button „failure“ deaktiviert werden).



EasyL2

Als Übertragungsprotokoll auf Schicht 2 wurde „*EasyL2*“ spezifiziert. Dieses sehr einfache Protokoll enthält die typischen Datenfelder, die auf Layer 2 ausgetauscht werden. Der Aufbau ist wie folgt:

Datenfeld	Größe	Zusätzliche Information
Präambel	1 byte	„Magic number“, muss immer 0x77 sein
Empfänger	2 byte	MSB zuerst übertragen
Sender	2 byte	MSB zuerst übertragen
Größe	1 byte	Größe des gesamten Frames
Daten	0-100 byte	Nutzdaten, max. 100 byte
CRC	4 byte	Prüfsumme über Frame

EasyL2 spezifiziert KEINE Wiederholungsanforderung, wenn beispielsweise vom Netzwerk-Stack erkannt wird, dass der Frame fehlerhaft ist. Derartige Funktionalität muss von darüberliegenden Schichten implementiert werden. D. h., wenn beispielsweise die Länge des Paketes oder CRC nicht übereinstimmt, wird der Frame einfach auf Layer 2 vernichtet und nie in Richtung Layer 3 weitergereicht. Layer 3 Schichten könnten dies beispielsweise mittels Timeouts erkennen und müssen entsprechend reagieren.

Simulation des Netzwerk-Stacks

Ein wesentlicher Bestandteil einer Netzwerk-Simulation ist der virtuelle Netzwerk-Stack von Rechnern. Dieser besteht aus zwei eigenständigen Teilen: einem Stack für ankommende (*IncomingNetworkStack*) und für auszusendende (*OutgoingNetworkStack*) Daten.

IncomingNetworkStack

Vom Kabel (Wire) ankommende Pakete werden beim IncomingNetworkStack eines Rechners angeliefert. Die Verarbeitung beginnt bei Layer 1 (physisches Entgegennehmen von der Leitung) und endet bei Layer 6. Hat ein Paket Layer 6 (erfolgreich) überwunden, wird es an die Applikation ausgeliefert. In jeder Schicht können so genannte *DecodingNetworkFilter* installiert werden. Dabei handelt es sich um Filter, welche eine spezielle, protokollspezifische Aufgabe erledigen. Eine Aufgabe für einen Filter auf Layer 2 wäre z. B., nur Pakete durchzureichen, die wirklich auch an den jeweiligen Rechner adressiert sind. Weiters muss mittels Filter auf Layer 2 der EasyL2 Frame aufgelöst werden. An Layer 3 dürfen nur Layer 3 Nutzdaten weitergereicht werden, da EasyL2 für die oberen Schichten transparent ist.

OutgoingNetworkStack

Möchte eine Applikation Daten an einen anderen Rechner versenden, so müssen diese den OutgoingNetworkStack des Rechners passieren. Die Daten werden sukzessive von einer oberen in eine untere Schicht weitergereicht, bis dass Sie schlussendlich über das Kabel weitertransportiert werden. Dazwischen werden Verarbeitungsschritte der jeweiligen Schicht vorgenommen, welche in *EncodingNetworkFilter* gekapselt sind. In Layer 2, z. B., müssen die Daten u. a. mit dem EasyL2 Protokoll ummantelt werden.

Übungsmodalität

- Download des Frameworks und der Dokumentation von der Übungshomepage **ab 26.10.2004**
- Für diese Übung haben Sie zwei Wochen Zeit. Deshalb ist sie umfangreicher gestaltet und wird stärker gewertet.
- Abgabe des Programmcodes (nur geänderte Klassen; Quellcode + compiliert) zusätzlich via E-Mail an uebung3.netzwerke@putzinger.org (Deadline: 09.11.2004, 15:30 Uhr)
- Fragen und Probleme bzgl. der Übung? Mail an putzinger@fim.uni-linz.ac.at

Tipps und Hinweise

- Zu startende Klasse:
  `at.jku.fim.datalinksimulation.gui.SimulationApplication`
- Die einzelnen Komponenten und Stacks führen Ihr eigenes Log. Sie sind ein erster Anlaufpunkt für die Fehlersuche. Loggen Sie in den Übungsbeispielen ebenfalls alle wichtigen Aktionen, Zustände und Ergebnisse mit. In den Logs können Sie überprüfen, ob Ihre Implementierung auch tatsächlich funktioniert.
- Da Java keinen Datentyp unterstützt, dessen Wertebereich von 0-255 reicht, wird der Typ „int“ anstelle von „byte“ verwendet. Der Wertebereich ist aber trotzdem von 0 bis 255 beschränkt. D. h., für eine 16bit Zahl werden 2 int benötigt, wobei die höherwertigen Bits zuerst übertragen werden.
- Sehen Sie regelmäßig auf der Übungshomepage nach, ob es neue Hinweise oder Versionen des Frameworks gibt.

Beispiel 7

In der Implementierung des EasyL2 Protokolls fehlt die Generierung (beim Sender) bzw. die Überprüfung (beim Empfänger) des CRCs. Diese muss in der Klasse `at.jku.fim.datalinksimulation.layer2.protocols.EasyL2Utils` ergänzt werden:

1. Verdeutlichen Sie in Pseudo-Code, wie man mittels Schieberegister den CRC für eine beliebige Byte-Folge erzeugen kann.
2. Worauf muss bei der Wahl eines CRC-Polynoms geachtet werden und begründen Sie Ihre Antworten?
3. Implementieren Sie die beiden fehlenden Methoden in der genannten Klasse und verwenden Sie hierbei das CRC32-Polynom und ein simuliertes Schieberegister. Verwenden Sie bei der Implementierung keine Lookup-Table! Der CRC wird über den gesamten Frame ohne die letzten 4 Byte (= CRC selbst) gebildet. Zusätzlich müssen noch Nullen im Schieberegister nachgeschoben werden. Überlegen Sie, wozu dies gut sein könnte und wie viele Nullen sinnvoll sind.
4. Wählen Sie ein beliebiges, gänzlich „ungeeignetes“ CRC Polynom und verwenden Sie es für EasyL2. Konstruieren Sie ein Beispiel, wo der Fehler in einem Frame mit Hilfe des CRCs nicht gefunden werden kann (wahrscheinlich wird es hilfreich sein, den „Störalgorithmus“ für das Kabel abzuändern, sodass beispielsweise immer an derselben Stelle der Bitfehler auftritt). Dokumentieren Sie Ihre Arbeit.
5. Recherchieren Sie, in welchen Anwendungsgebieten außer Netzwerksystemen CRC noch verwendet wird.

Beispiel 8

Mittels der Stations-ID „0“ im Empfänger-Teil eines EasyL2-Frames soll ab sofort ein „Broadcast“ durchgeführt werden können. Lokalisieren Sie die dafür verantwortliche Stelle im Code, und ändern Sie diese entsprechend ab. Deaktivieren Sie zum Testen die Leitungsstörung.

Beispiel 9

Überlegen Sie sich, wie die Implementierung einer „Switch“ Komponente für EasyL2 in dem Framework aussehen würde, und halten Sie Ihre Überlegungen schriftlich fest. Bedenken Sie u. a. folgende Fragestellungen:

- Was muss der Software-Switch leisten können? Was passiert beim Eintreffen eines Paketes auf einem „Port“?
- Auf welchem ISO/OSI Layer sind typische Switches angesiedelt? Begründung!
- Welche Fragestellungen oder Probleme treten bei Switches auf, die es auch in der Realität zu lösen gibt?