Client/Server-Anwendung für ein Sensoren-Netzwerk

Für ein Forschungsprojekt sollen Umweltdaten über viele gleichartige Sensoren gesammelt und über eine Funkverbindung an einen Server gesendet werden. Dort sollen die Daten gesammelt und ausgewertet werden. Eine besondere Schwierigkeit ergibt sich aus der relativen Unzuverlässigkeit der Funkverbindung: Nicht alle Daten kommen vollständig und/oder korrekt beim Empfänger – also beim Server – an.

Zunächst geht es darum einen Entwurf für ein Anwendungsprotokoll für solch ein Sensor-Netzwerk zu bewerten (Aufgabe 1). Anschließend soll ein entsprechender Server implementiert werden (Aufgabe 2).

Aufgabe 1 (30%):

Die einzelnen Sensorstationen sammeln Daten (Beispiele in Klammern) zu Temperatur (13,7 °C), Luftdruck (1012 hPa), Luftfeuchtigkeit (63%), Windgeschwindigkeit (12 km/h) und Windrichtung (NW). Außerdem sendet jede Station eine Stationskennung, um die Daten den einzelnen Stationen zuordnen zu können. Die eigentlichen Sensordaten und die Stationskennung werden dabei auf der Sender-Seite noch mit einer Prüfsumme versehen. Mit Hilfe dieser Prüfsumme kann auf der Empfängerseite überprüft werden, ob der Datensatz vollständig und korrekt übertragen wurde:

<ID#Temperatur#Luftdruck#Luftfeuchtigkeit#Windgeschwindigkeit#Windrichtung>Prüfsumme?

Also für die oben gegebenen Beispielwerte und die Stationskennung 53:

`<53#13.7#1012#63#12#NW>1167?`

Das Prüfsummenverfahren wird in Aufgabe 2 genauer beschrieben, muss hier (in Aufgabe 1) aber nicht weiter betrachtet oder verstanden werden!

Ein Kollege von Ihnen weist darauf hin, dass die Übertragungsgeschwindigkeit auf der Funkstrecke so gering ist, dass man unvollständig empfangene Datenpakete nicht noch einmal schicken kann. Er behauptet, dass es wegen der unzuverlässigen Funkverbindung sinnvoller wäre, das Protokoll so zu ändern, dass jeder einzelne Messwert (also etwa die Temperatur) zusammen mit der Stationskennung und einer Prüfsumme verschickt wird.

(a) Nehmen Sie zu der oben genannten Behauptung des Kollegen begründet Stellung. Gehen Sie hierbei auf folgende Aspekte ein:
- Welche Vorteile und welche Nachteile würden sich aus so einer Trennung ergeben?
- Spielt es dabei eine Rolle wie groß die Wahrscheinlichkeit für unvollständige und/oder unkorrekte Übermittlung der Daten über die Funkstrecke ist?

(b) Nennen und erläutern Sie auf der Grundlage des Schichtenmodells, Gemeinsamkeiten und Unterschiede zwischen den Protokolltypen UDP und TCP.

(c) Geben Sie an, welche der beiden Protokollvarianten UDP oder TCP die Verwendung von Prüfsummen als Teil des Anwendungsprotokolls unnötig machen würde. Begründen Sie Ihre Auswahl und gehen Sie dabei auf Vor- und Nachteile der Protokollvarianten ein.

Aufgabe 2 (70%):

Es soll mit Java ein Server entwickelt werden, der Temperatur- und Luftdruckwerte von mehreren Clients (Sensor-Stationen) empfangen und anzeigen kann. Im Eclipse-Workspace finden Sie zu diesem Zweck einen fertigen Client in Form einer ausführbaren JAR-Datei `SensorStation.jar` . Diesen Client können Sie mehrfach starten und dadurch gleichzeitig mehrere Clients Daten an Ihren Server schicken lassen. Beim Starten des Clients müssen Sie jeweils eine Stationskennung festlegen. Es liegt in Ihrer Verantwortung, niemals zwei Clients mit der gleichen Stationskennung zu starten!

Die Clients schicken alle fünf Sekunden ihre Temperatur- und Luftdruckdaten im Format

<ID#Temperatur#Luftdruck>Prüfsumme?

Implementieren Sie den Server entsprechend der folgenden Teilaufgaben:

(a) Der von Ihnen zu implementierende Server soll auf Port `11111` die Daten von den Clients empfangen. Wenn sich ein neuer Client verbindet, soll aus dem ersten empfangenen Datensatz die Stationskennung (`ID`) ermittelt und abgespeichert werden.

Hinweis: Wenn es Ihnen nicht gelingt, die Stationskennung zu ermitteln, dann können Sie ersatzweise (Punktabzug) die Stationskennung `77` annehmen und verwenden.

Der Server soll die empfangenen Datensätze in der `JList`-Komponente wie in Abbildung 1 gezeigt darstellen. Neu empfangene Datensätze werden dabei ans Ende der Liste angehängt. Beachten Sie, dass die `JList`-Komponente zu diesem Zweck in eine `JScrollPane` eingebettet sein sollte (siehe Abbildung 3).

Die Clients (`SensorStation.jar`) senden bei uns ihre Daten natürlich nicht per Funk. Um die unzuverlässigere Funkverbindung zu simulieren, kann man im Client die Checkbox „Unzuverlässige Funkverbindung simulieren“ aktivieren. Solange dieses Häkchen nicht gesetzt ist kommen die Daten der Clients immer korrekt bei Ihrem Server an.

Wenn dieses Häkchen gesetzt ist, wird der Client die Daten nicht mehr verlässlich korrekt senden. Im Protokollfenster des Clients sind Datensätze, die auf diese Weise gesendet wurden mit `**` am Ende des Datensatzes markiert. Zu Ihrer Hilfe sind diejenigen Datensätze, die tatsächlich fehlerhaft übertragen wurden mit `**!!**` markiert. Siehe Abbildung 2. **Achtung**: Die Markierungen `**` sowie `**!!**` werden **nicht** an den Server gesendet! Es sind ausschließlich Hinweise im Protokollfenster des Clients!

Für den nächsten Aufgabenteil soll aber genau diese Unzuverlässigkeit aktiviert werden. Dazu müssen Sie den Haken in der Checkbox des Clients setzen.

(b) *Hinweis*: Sollten Sie mit der Umsetzung dieser Teilaufgabe Schwierigkeiten haben, können und sollten Sie dennoch die leichtere Teilaufgabe (c) bearbeiten!

Die empfangenen Daten müssen nun zunächst mit Hilfe der Prüfsumme auf Korrektheit überprüft werden. Die Prüfsumme ist eine ganze Zahl zwischen dem Größer-Zeichen (`>`) und dem abschließenden Fragezeichen (`?`). Sie wird auf der Client-Seite wie folgt berechnet:
Über alle einzelne Zeichen (Datentyp `char`) des zu sendenden Strings bis einschließlich des Größer-Zeichens wird eine Summe gebildet. Dabei wird sich zu Nutze gemacht, dass jeder `char`-Wert auch als `int`-Wert interpretierbar ist (**siehe Anhang**). Die so insgesamt gebildete Summe ist unsere Prüfsumme und wird durch das abschließende Fragezeichen (`?`) begrenzt.

Ihr Server muss jetzt nach dem selben Verfahren alle empfangenen Zeichen – beginnend mit dem führenden Kleiner-Zeichen (<) und bis einschließlich des Größer-Zeichens (`>`) – wie oben beschrieben aufsummieren. Nur dann, wenn die so gefundene Prüfsumme identisch mit der empfangenen Prüfsumme ist, sind die empfangenen Daten korrekt. Wenn nicht, muss dieser Datensatz ignoriert werden!

Wenn ein Datensatz mit unkorrekter Prüfsumme empfangen wurde, soll statt des Datensatzes in der `JList`-Komponente ein Eintrag in der folgenden Form eingefügt werden:

```
Prüfsummenfehler für Sensorstation 53: 1
```

Die erste Zahl steht dabei für die Stationskennung und die zweite Zahl (als Vorbereitung für Teilaufgabe c) für die Anzahl von aufeinander folgenden Datensätzen dieser Sensorstation, bei denen es einen Prüfsummenfehler gab. Siehe Abbildung 3.

Der Zähler für die Anzahl von registrierten Prüfsummenfehlern wird wieder auf 0 gesetzt, wenn beim Zählerstand 1 der nächste Datensatz korrekt (ohne Prüfsummenfehler) empfangen wird.

(c) Bislang handelt es sich bei der Kommunikation zwischen Client und Server um eine reine Einweg-Kommunikation: Daten werden bisher ausschließlich vom Client zum Server geschickt. Das soll im letzten Schritt geändert werden!

Der Server soll Kommunikationsprobleme mit einzelnen Clients erkennen und diese zu einem Reset veranlassen können: Wenn der Server zweimal hintereinander vom gleichen Client einen Datensatz mit

einer falschen Prüfsumme erhalten hat (siehe Hinweis unten), sendet er an den Client den Befehl

```
R!
```

Der Client antwortet darauf nicht, aber schaltet sich zurück in den Zustand ohne simulierte Funkverbindung. Darum müssen Sie sich natürlich nicht kümmern. Sie sehen den Erfolg des Reset Befehls aber daran, dass der entsprechende Haken an der Checkbox des Clients verschwindet.

Hinweis: Wenn Sie die Teilaufgabe (b) nicht lösen konnten, dann können Sie ersatzweise (Punktabzug) nach 100 empfangenen Zeichen den Reset Befehl an den Client schicken.

Anhang

Testdaten für die Prüfsummenberechnung:

String „A“ –> 65
String „AB“ –> 131
String „ABC“ –> 198

Erläuterung:

`char 'A'` entspricht `int 65`
`char 'B'` entspricht `int 66`
`char 'C'` entspricht `int 67`

Die Prüfsumme über den String berechnet sich als Summe der den einzelnen Zeichen entsprechenden Integer-Werte.

Die Prüfsumme `692` in der ersten `EMPFANGEN` Zeile in Abbildung 1 ist die Summe über die Integer-Werte der einzelnen Zeichen des Strings `"53#10,6#1008"`

Abbildung 1: Empfang von korrekten Daten

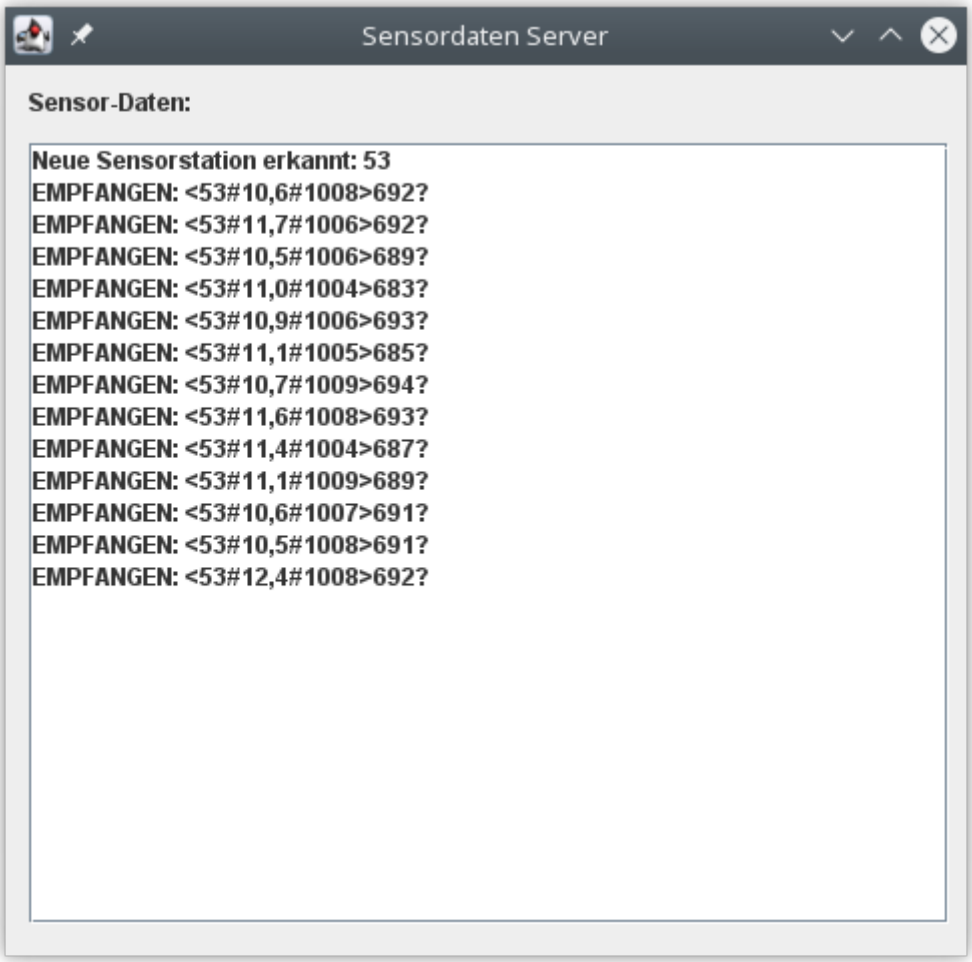
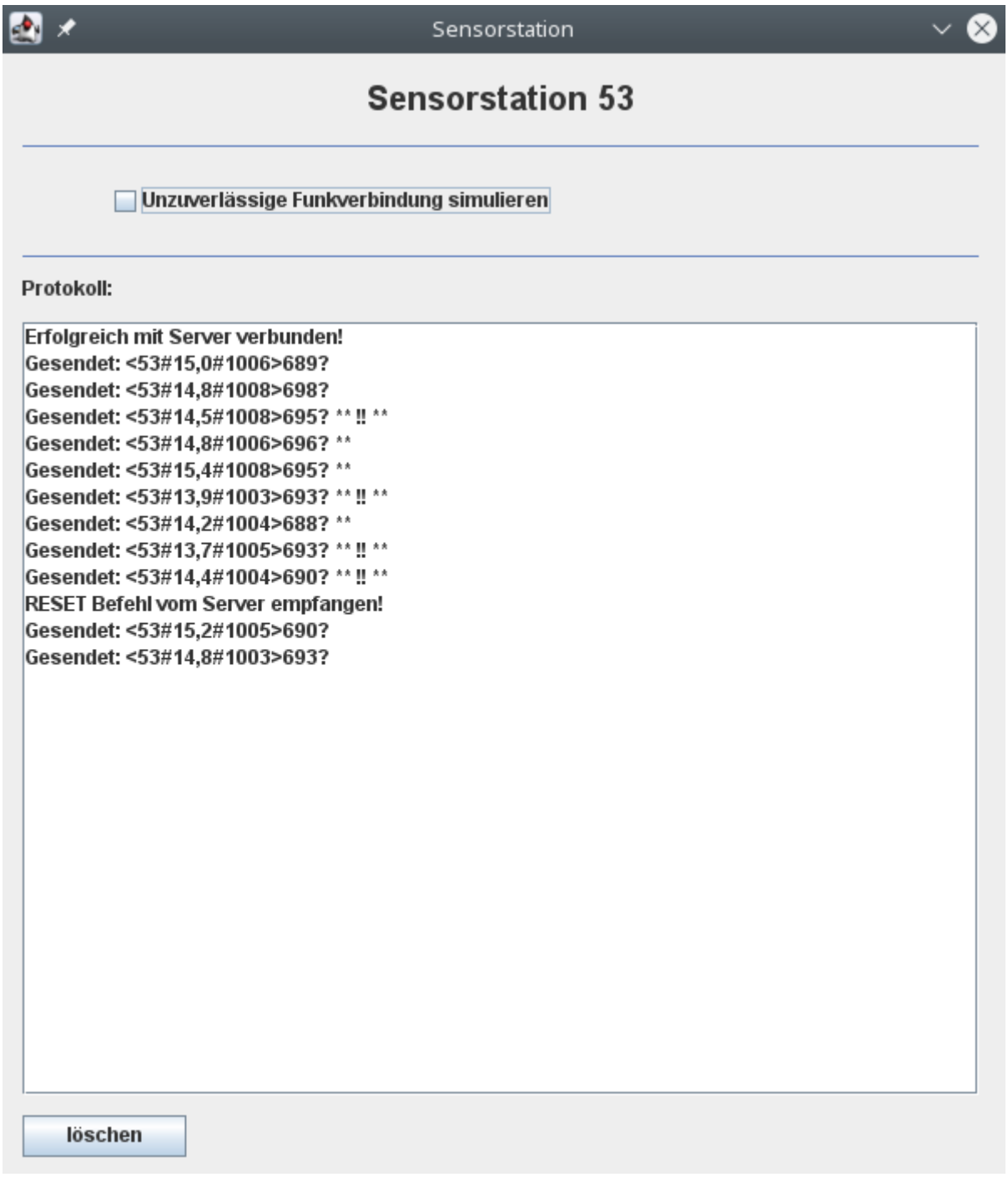
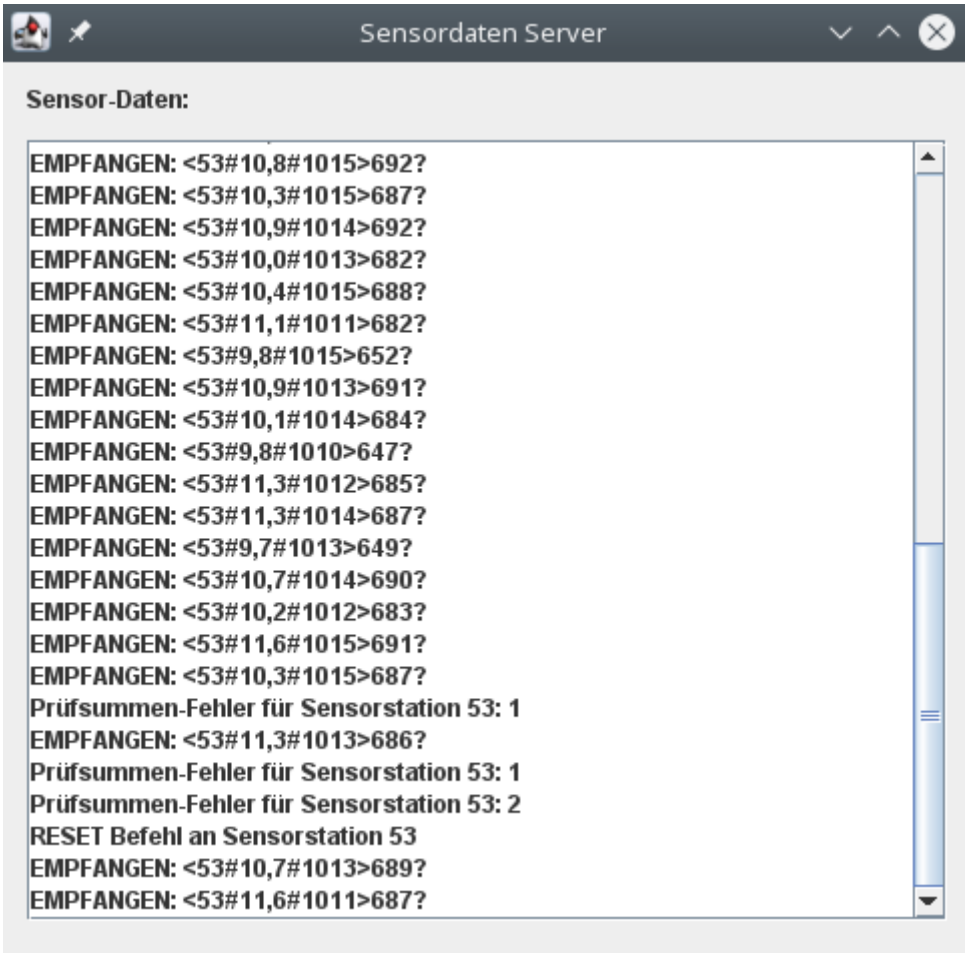


Abbildung 2: Client sendet fehlerhafte Datensätzen



Das Häkchen an der Checkbox ist nicht mehr gesetzt, weil der Client inzwischen einen RESET Befehl vom Server erhalten hat und deshalb die Daten nun wieder ohne Fehler überträgt. Man kann sehen, dass der dritte bis neunte Datensatz mit simulierter Unzuverlässigkeit versendet wurde (markiert durch **). Dabei traten tatsächlich vier Fehler auf (markiert durch **!!**)

Abbildung 3: Der Server empfängt (und erkennt) fehlerhafte Datensätzen



Nachdem der Server zunächst längere Zeit fehlerfrei Daten empfangen hat wurde beim Client die Simulation der unzuverlässigen Funkverbindung aktiviert. Man kann sehen, dass nach einem ersten fehlerhaft empfangenen Datensatz der nächste Datensatz wieder fehlerfrei ankam (der Fehlerzähler wurde deshalb auch wieder zurückgesetzt): **Unzuverlässige Datenverbindung bedeutet eben nicht zwangsläufig, dass es zu Fehlern kommt. Aber es kann zu Fehlern kommen!**
Direkt im Anschluss kommt es aber zu zwei aufeinanderfolgenden fehlerhaften Datensätzen. Der Server hat deshalb den RESET Befehl an den Client gesendet.