SMS senden via AT-Befehle

DOKUMENTATION

Wien am 4. Januar 2014
Technische Universität Wien

ausgeführt im Rahmen der Lehrveranstaltung
183.286 – (Mobile) Network Service Applications

Gruppe 35

Stefan Gamerith
0925081
E 066 937
Software Engineering and Internet Computing

Kurzfassung

Im Rahmen der Lehrveranstaltung *(Mobile) Network Service Applications* wurde eine Java Applikation entwickelt welche es ermöglicht, SMS-Nachrichten beliebiger Länge zu versenden. Der Nachrichtentext wird bei der Übertragung mit dem GSM 7-Bit Alphabet [4] enkodiert.

Inhaltsverzeichnis

Tabellenverzeichnis

Abbildungsverzeichnis

1 Einleitung

Aufbauend der Aufgabenstellung *Entwicklung einer Applikation zum Senden von SMS* wurde eine Java-Applikation entwickelt welche es ermöglicht, SMS-Nachrichten beliebiger Länge zu versenden. Das Programm bedient sich sogenannter AT-Befehle [1] zur Kommunikation mit dem Mobiltelefon. Die Datenübertragung geschieht über eine serielle Schnittstelle.

2 Problemstellung/Zielsetzung

Die Java-Applikation kommuniziert mit dem Handy (Nokia 6212) über die serielle Schnittstelle. Intern werden SMS-Nachrichten unter Verwendung sogenannter AT-Befehle versendet.

2.1 AT-Befehle

AT-Befehle werden in 1) *Basic commands*, 2) *Extended commands*, 3) *Parameter type commands* und 4) *Action type commands* unterteilt [1]. Während *Basic commands* auf jedem Modem verfügbar sind, kann die Verfügbarkeit bzw. die genaue Syntax der *Extended commands* mit nachgestelltem Fragezeichen abgefragt werden. Letztere verfügen zusätzlich noch über den Präfix `AT+`. Ein erfolgreich ausgeführter Befehl wird vom Modem mit `OK` quittiert, andernfalls schickt es `ERROR` und bricht die Abarbeitung weiterer Befehle ab.

Tabelle 1 listet alle notwendigen AT-Befehle zum erfolgreichen Versenden von SMS-Nachrichten

AT-Befehl	**Beschreibung**
`ATZ`	Befehl zum Zurücksetzten des Modems.
`AT+CMGF`	Definiert, welches Format für die SMS-Übertragung verwendet wird. (`0` für *PDU-Mode* und `1` für *Text-Mode*)
`AT+CMGS`	Versendet direkt eine SMS. Die genaue Syntax ist abhängig vom Übertragungs-Mode.

Tabelle 1: Menge aller notwendigen AT-Befehle zum Versenden von SMS-Nachrichten

2.2 PDU Mode

Eine wichtige Anforderung für die Implementierung war das Versenden der SMS-Nachrichten im PDU-Mode. Dieser ermöglicht die Übertragung von Binärdaten welche als hexadezimale Zeichenkette angegeben werden können.

Die zwei wichtigsten Nachrichtentypen im PDU-Mode sind SUBMIT-Nachrichten und DELIVER-Nachrichten. Abbildung 1 zeigt schematisch den Aufbau einer SMS-SUBMIT-Nachricht im PDU-Mode [2] (in Byte Blöcken).

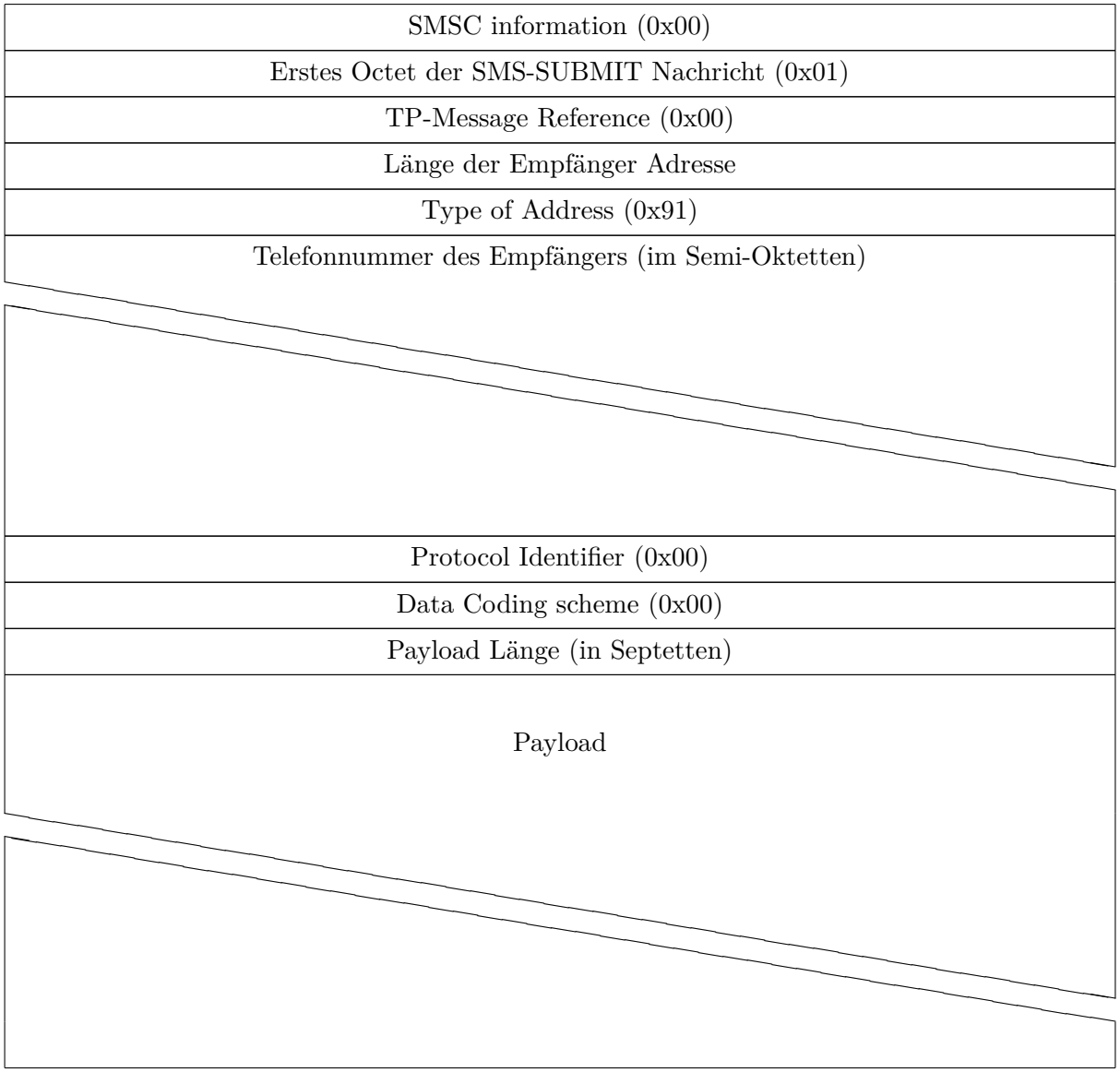

Abbildung 1: Aufbau einer SMS-SUBMIT-Nachricht im PDU-Mode

Um auch Nachrichten mit einer Länge größer als 160 Zeichen versenden zu können, wurde ein eigener Nachrichtentyp spezifiziert. Der Unterschied zum herkömmlichen Nachrichtentyp ist der sogenannte *User Data Header* welcher am Beginn des Payloads zu finden ist. Abbildung 2 zeight schematisch den Aufbau des User Data Headers.

Länge des User Data Headers (Anzahl der Oktetten)
User Data Header Identifier (0x08)
Länge der Referenz Nummer (0x04)
Referenz Nummer (0x0000)
Anzahl der Teile
Teilnummer

Abbildung 2: Schematischer Aufbau des User Data Headers

2.3 Text Kodierung

Eine weitere Anforderung war die Verwendung des GSM 7-Bit Alphabets [4] für die Textkodierung. Diese ermöglicht die Übertragung von 128 unterschiedlichen ASCII-Zeichen. Nach der Umwandlung der Zeichen in Septetten (7-Bit Blöcke) müssen diese noch mit einem geeigneten Algorithmus in Oktetten (8-Bit Blöcke) gepackt werden, da Daten üblicherweise byteweise übertragen werden.

Abbildung 3 veranschaulicht grafisch die Konvertierung von 8 Septetten in 7 Oktetten.

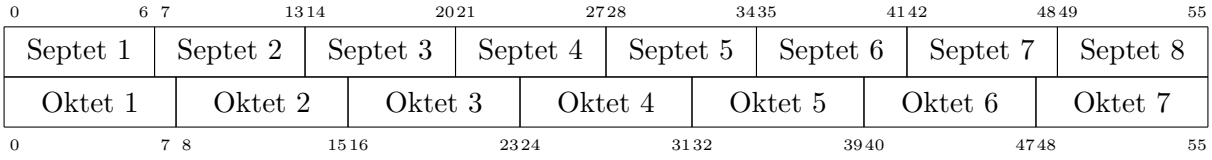

Abbildung 3: Umwandlung von 8 Septetten in 7 Oktetten

Falls die Anzahl der zu übertragenden Septetten nicht einem Vielfachen von 8 entspricht, müssen sogenannte *Paddingbits* noch am Schluss angefügt werden.

3 Methodisches Vorgehen - Fallbeispiel

3.1 Laufzeitanforderungen

Da Nokia nur Windows Treiber zur seriellen Kommunikation mit dem Handy zur Verfügung stellt, fiel die Entscheidung auf Windows XP SP3 als primäre Entwicklungs- und Laufzeitumgebung.

Eine weitere Vorraussetzung für eine einwandfreie Funktionsweise ist die Installation des Nokia 6212 NFC SDKs. Letztere installiert einen COM Port, über jenem die gesamte Kommunikation mit dem Mobiltelefon statt findet. Wichtig ist, das dieser Port AT-Befehle akzeptiert. Dazu öffnet man am besten eine Konsole (zum Beispiel *Hyperterminal*) und tippt `AT` gefolgt von

Enter ein. Anschließend sollte dieser Befehl mit `OK` bestätigt werden. Sollte dies nicht der Fall sein, empfiehlt sich die Neukonfiguration beziehungsweise die Vergabe eines anderen COM-Ports bei einer Neuinstallation des Modems.

Zur seriellen Kommunikation ist weiters die Installation der *Java Comm API* notwendig. Hierzu muss die native Bibliothek *win32com.dll* in den Order `%JAVA_HOME%/jre/bin` und die Datei *javax.comm.properties* in den Order `%JAVA_HOME%/jre/lib` kopiert werden (beide Dateien wurden von der LVA-Leitung zur Verfügung gestellt).

3.2 Build Management

Als Buildmanagement Werkzeug wurde Apache Maven [3] verwendet. Der Befehl `mvn package` erstellt ein ausführbares jar-Archiv im Order *target*. Anschließend genügt das Ausführen des Befehls *java -jar sms-sender-1.0-SNAPSHOT-jar-with-dependencies.jar* um das Programm zu starten.

3.3 Konfigurationsdateien

Die primäre Konfigurationsdatei ist *sendsms.properties*. Abbildung 4 zeigt beispielhaft eine ebensolche Datei.

```
csvfile=sendsms.csv
port=COM4
```

Abbildung 4: Beispiel der Konfigurationsdatei *sendsms.properties*

Daneben muss noch ein Comma Separated File (CSV) existieren (spezifiziert in *sendsms.properties*). Abbildung 5 zeigt beispielhaft die Datei *sendsms.csv*.

```
+436507112256,This is a short text!
+436507112256,This is a not so short text!
```

Abbildung 5: Beispiel der Konfigurationsdatei *sendsms.csv*

4 Fazit

Essentiell für die korrekte Funktionsweise dieses Programms ist eine serielle Verbindung zum Mobiltelefon über jene AT-Befehle abgesetzt werden können. Sollte dies nicht möglich sein, schafft wahrscheinlich ein Blick in den Gerätemanager Abhilfe. Das Handy sollte als Modem im Gerätemanager gelistet sein. Ist dies nicht der Fall, hilft eine Neuinstallation des Handytreibers. Weiters sollte der richtige serielle Port angegeben werden. Dieser kann im Konfigurationsdialog des Modems im Reiter `Advanced` und einem Klick auf `Advanced Port Settings` konfiguriert werden.

Literatur

[1] GSM ETSI. 07.07: Digital cellular telecommunication system (phase 2+) at command set for gsm mobile equipment (me). *http://portal.etsi.org/action/pu/19991214/19991214.htm*, 1997.

[2] GSM ETSI. Digital cellular telecommunications system technical realization of the short message service (sms) point-to-point, technical report. *ETSI 2001, ETSI TS 100 901 V7.5.0*, 1998.

[3] The Apache Software Foundation. `http://maven.apache.org`, 2014. accessed 2-Januar-2014.

[4] International Reference Alphabet (IRA). 7-bit coded character set for information exchange. *ITU-T Recommendation T .50*.