HTWK Leipzig
Fachbereich IMN
Sommersemester 2014

Traincard

Beleg im Smartcard Programmierung
bei Prof. Dr. rer. nat. Uwe Petermann

Kurt Junghanns, B.Sc.
Marcel Kirbst, B.Sc.
Michael Reher, B.Sc.

11. Juli 2014

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

Diese Arbeit befasst sich mit der Vorstellung der Belegarbeit im Fach Smartcard Programmierung mit dem Thema “Traincard”. Ziel des Belegs ist die prototypische Implementierung eines Trainingssystems auf Basis JCOP-fähiger Smartcards.

Im ersten Kapitel Anwendungsfall wird der Anwendungsfall für die prototypische Implementierung beschrieben, sowohl unter Anwendungsfall aus Sicht des Trainierenden aus Sicht des Trainierenden, wie auch unter Anwendungsfall aus Sicht des Trainers aus Sicht des Trainers. Im Kapitel Grundlagen werden die grundlegenden Technologien und Standards vorgestellt. Kapitel Umsetzung erläutert detailliert die Systemumgebung und beschreibt den On-Card und den Off-Card Teil der Implementierung. Abschließend wird im Kaptiel Zusammenfassung und Ausblick eine Einschätzung beim Abschluss des Belegs gegeben.

2 Anwendungsfall

In vielen Fitnessstudios werden heute noch bei der Neuanmeldung Trainingspläne auf Papier ausgegeben. Der Trainierende führt den Trainingsplan über die Trainingsperiode ständig bei sich und verzeichnet den Trainingsfortschritt in diesem Dokument. Im Verlauf der Trainingsperiode kann der Trainer beurteilen wie effektiv das Training beim Trainierenden ist.

Die Verwendung altmodischer Trainingspläne auf Papier hat jedoch einige Nachteile. Beispielsweise wird das Dokument vom Trainierenden manchmal vergessen, die betreffenden Trainingsfortschritte müssen also nachgetragen werden. Weiterhin sind Traingspläne in Papierform nicht besondern resistent gegen Abnutzung und verschleißen mit fortdauernder Verwendung. Um den genannten Nachteilen zu begegnen soll im Rahmen dieser Belegarbeit der Einsatz von Trainingsplänen auf Basis so genannter Smartcards abgebildet werden. Smartcards können sehr leicht in einer Brieftasche mitgeführt werden und sind weniger anfällig für Abnutzung. Weiterhin bieten sich viele weitere Vorteile wie ... (Datenlogging für das Studio, Kopieren bzw. verteilter Zugriff auf die Daten ....)

2.1 Anwendungsfall aus Sicht des Trainierenden

Aus Sicht des Trainierenden bietet die Verwendung der Smartcard einige der folgenden Vorteile:

- leichte Transportierbarkeit
- robuster als Trainingspläne aus Papier

2.2 Anwendungsfall aus Sicht des Trainers

Trainer profitieren beim Einsatz von Smartcards unter anderem von folgenden Vorteilen:

- leichte und lesbare Auswertung der vorgegebenen Trainingspläne
- zusätzliche Metainformationen wie beispielsweise zu welchem Zeitpunkt welcher Datensatz geschrieben wurde
- statistische Auswertungen lassen sich sehr leicht erstellen

3 Grundlagen

In diesem Kapitel wird auf die Grundlagen der in diesem Beleg verwendeten Technologien eingegangen.

3.1 Grundlagen Smartcard

Die in diesem Beleg verwendete Smartcard basiert auf der Java Card Technologie. Die Java Card Technologie bietet eine Teilmenge der Java Programmiersprache, sowie eine hinsichtlich der Anforderungen an Smartcards optimierte Laufzeitumgebung.
Die Verwendung von Java-basierten Smartcards bietet einige Vorteile, von denen nachfolgend eine Auswahl beispielhaft genannt sei: [2]

plattformunabhängig: Java Card Applets, die der Java Card API entsprechen, lassen sich plattformunabhängig und herstellerübergreifend nutzen.

Multiapplikationsfähig: es können mehrere Applikationen gleichzeitig auf einer Smartcard ausgeführt werden

hohe Flexibilität: die Verwendung der objektorientierten Programmiersprache Java erlaubt die Erstellung komplexer Anwendungen für die Smartcard.

Post-Aktualisierbarkeit: die Möglichkeit, nachträglich Code auf der Smartcard zu modifizieren und auszutauschen erhöht die Flexibilität weiter

Standardkonformität: die Java Smartcards entsprechen dem ISO7816 Standard [3]

Eine Java Smartcard besteht im Wesentlichen aus den Bestandteilen Kommunikationsschnittstelle, Speicher und einem Prozessor zur Durchführung von Berechnungen. Bei Verwendung der Smartcard wird diese in ein Lesegerät eingelegt. Das Lesegerät wird in einschlägiger Literatur auch als Card Acceptance Device, abgekürzt CAD, bezeichnet. Der Speicher auf Java Smartcards besteht aus zwei Typen, RAM und EEPROM. Der RAM-Speicher ist flüchtig und kann beliebig oft beschrieben werden. Der EEPROM-Speicher ist nichtflüchtig und kann nur endlich oft beschrieben werden, je nach Hersteller und Modell bis zu 100.000 mal pro Speicherzelle.

3.2 JCOP

Die Java Card Open Plattform, abgekürzt JCOP, ist ein Smartcard Betriebssystem das iniitial von IBM entwickelt wurde, inzwischen jedoch von NXP Semiconductors

betreut wird. Der Titel leitet sich von den Namen der zu Grunde liegenden Spezifikationen für Java Card und Global Platform (früher bekannt unter Open Plattform) ab.

Bei der Softwareentwicklung kann auf reale sowie auf simulierte JCOP-Karten zurückgegriffen werden. Der Zugriff auf eine reale JCOP-Karte erfolgt mittels eines Terminal-Kartenlesers sowie spezieller Treiber. Da während der Erarbeitung des Belegs keine Hardware zur Verfügung stand, wurde die Implementierung an einer simulierten JCOP-Karte getestet. Auf die Entwicklungsumgebung wird im Kapitel Umsetzung näher eingegangen.

3.3 APDU

Die Kommunikation zwischen dem Kartenlesegerät und der Smartcard erfolgt reaktiv und paketweise. Reaktiv bedeutet, dass die Smartcard keine Kommunikation initiiert sondern nur auf Anfragen vom Lesegerät reagiert. Der bei der Kommunikation zwischen Smartcard und Lesegerät verwendete Kommunikationsmechanismus wird als Application Protocol Data Units, abgekürzt APDU, bezeichnet. Spezifiziert ist dies ebenfalls in ISO7816 Teil 4.[3]

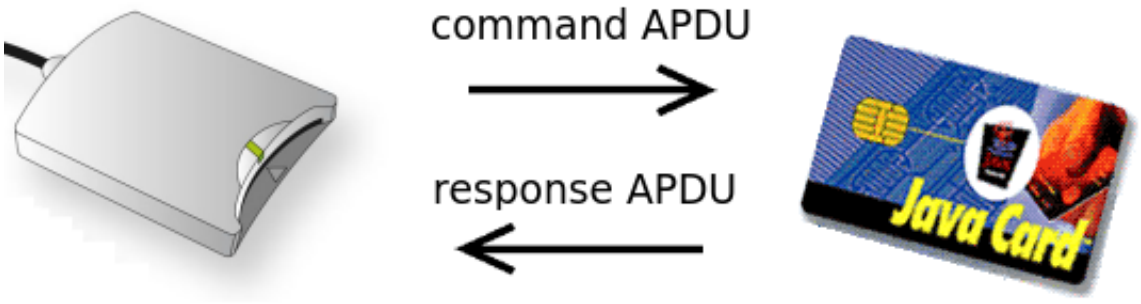

Abbildung 1: APDU-Kommunikation schematisch dargestellt, Quelle: Autoren

Die Kommunikation wird über ein so genanntes command APDU Paket initiiert, welches aus den folgenden Feldern besteht:

CLA: class,gibt die Klasse an, spezifiziert ob es sich um ein ISO7816-4 konformes Kommando handelt

INS: gibt die Instruktion an

P1: zusätzlicher Parameter

P2: zusätzlicher Parameter

Weiterhin können je nach Kommandotyp noch die folgenden, optionalen Felder an das command APDU Paket angehängt werden:

Lc: Length, gibt die Länge der Kommandodaten

Data: gibt die Kommandodaten an

Le: gibt die Länge der erwarteten Antwort an

Als Antwort erhält das Lesegerät von der Smartcard ein so genanntes response APDU Paket. Dieses Antwortpaket kann ein Datenfeld enthalten, dies ist jedoch nicht obligatorisch. Das response APU Paket ist wie folgt aufgebaut.

Data: optionales Datenfeld

Sw1: Statusword, erstes Byte

Sw2: Statusword, zweites Byte

4 Umsetzung

Dieses Kapitel legt die Umsetzung dar.

4.1 Kommunikation

Die Anwendung besteht aus einem On-Card und einem Off-Card Teil.
Entsprechend der JCOP Umgebung, ist der On-Card Teil durch ein Applet realisiert, welches auf der Smartcard installiert und gestartet wird. Diese Smartcard kann eine physische oder eine emulierte zum Einsatz kommen. Im Rahmen dieses Projektes wird die Smartcard per JCOP Eclipse Umgebung emuliert und verwendet. Dabei ist in der gestarteten JCOP Shell der Befehl */close* auszuführen, wodurch die Smartcard für externe Zugriffe auf der lokalen IP des Emulator-Rechners auf dem Port 8090 erreichbar ist.
Der Off-Card Teil ist ebenfalls in Java geschrieben und kommuniziert mit Hilfe des OpenCard Frameworks mit der Smartcard.

Die Kommunikation zwischen On-Card und Off-Card auf verschiedenen Rechnern benötigt entsprechende Regeln der Firewalls der Rechner, lokale Ausführung beider Teile auf einem Rechner hingegen keine.
Der Inhalt der APDUs orientiert sich stark an der Norm ISO 7816-4. Der konkrete Aufbau ist in Abbildung 2 dargestellt.
Die Abkürzungen stehen dabei für folgendes:

CLA ein Byte, welches immer mit dem Wert 0 belegt wird, da die APDU nicht vollständig ISO 7816-4 konform ist

INS ein Byte, welches die Instruktion angibt

NOA ein Byte, welches die Anzahl an gesamt zu übertragenden APDUs zur vollen Abarbeitung der Instruktion angibt

LEN ein Byte, welches die Anzahl der folgenden Daten-Bytes aufzeigt

DATA beliebige Anzahl von Bytes, welche die Daten enthalten[1]

SW1 ein Byte, welches den vorderen Teil der Statusrückgabe darstellt

SW2 ein Byte, welches den zweiten Teil der Statusrückgabe enthält

[1] Anzahl der Daten-Bytes sollte LEN entsprechen

Die Statusrückgabe wird automatisch durch die JCOP Umgebung an die APDU angehangen. Die Interpretation der Statusrückgabe ist anhand ISO 7816-4 durchzuführen.

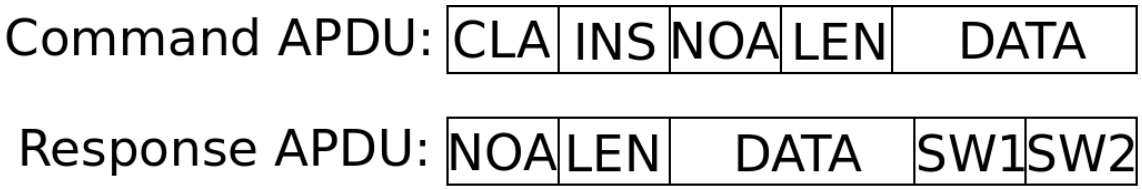

Abbildung 2: Selbstdefinierter Dateninhalt der APDUs

APDUs enthalten bis zu 256 Bytes, davon bis zu 252 Bytes für Daten. Es sind ganze Trainingspläne und andere Objekte im Byte Format zwischen On-Card und Off-Card Teil zu übertragen. Dazu existiert pro Datenmodell in der dazugehörigen Klasse je eine Funktion zur Umwandlung von Instanz zu Byte Array und zur Erzeugung einer Instanz aus einem Byte Array. Diese Umwandlung wurde eigens implementiert und orientiert sich am Aufbau von Ethernet Paketen. Ein Paket oder Byte Array besteht dabei immer aus einem Byte für einen eindeutigen Identifikator des Modells, zwei Bytes für die Anzahl der folgenden Bytes sowie Daten Bytes. Eine Instanz des Modells MyDate mit den Attributen Jahr[2], Monat und Tag wird in folgendes Byte Repräsentation umgewandelt: 01 00 03 0e 07 0b

Bei der Realisierung des Trainingssystemes werden keine Sicherheitsrelevanten Daten zwischen On-Card und Off-Card Teil übertragen. Auf Grund dessen wurde auf eine verschlüsselte Kommunikation mit asymmetrischer oder symmetrischer Verschlüsselung verzichtet. Einzig Rollen-spezifische Schreibvorgänge sind in beiden Card Teilen durch je ein Passwort geschützt. Der Trainer und der Sportler besitzen jeweils ein Passwort, welches mit der Java Bibliothek java.security.MessageDigest und derdarin enthaltenen Hashfunktion SHA-256 zu einem 32 Byte langem Wert gehasht. Dieser Hash ist auf der Smartcard gespeichert und muss bei der Anmeldung zum anschließenden Vergleich des Trainers oder Sportlers übertragen werden. Der jeweilige Benutzer gibt in der grafischen Oberfläche sein Passwort im Klartext ein worauf die Programmlogik dieses hasht und es an die Smartcard überträgt. Initial ist die Karte mit je einem Hash für Trainer und Sportler zu füllen. Im Zuge der Vereinfachung sind diese Werte bereits statisch gefüllt. Um den Passworthash des Trainers zu setzen ist das in Listing 1 dargestellte Kommando an die Smartcard zu senden.

[2] Bei einem Jahr werden nur die letzten zwei Ziffern hexadezimal abgespeichert.

```
/send 00 05 01 21 02
```

Listing 1: Setzen des Passwortes eines Sportlers in Form eines Kommandos

4.2 Datenstruktur

4.3 On-Card Teil

Der On-Card Teil der Implementierung enthält sämtliche Trainingspläne und weitere Daten.

4.4 Off-Card Teil

Der Inhalt dieses Abschnittes bildet die Umsetzung des Off-Card-Teils. Dabei wird auf die graphische Oberfläche, die Logik zur Steuerung und zur Kommunikation mit der Smartcard eingegangen.

Wenn man die Off-Card-Anwendung startet, gelangt man in das Hauptmenü der Anwednung, welches in der Abbildung dargestellt ist. Dieses Menü enthält fünf Buttons, welche jeweils zu einer Funktion des Programmes führt. Den gesamten Trainingsplan, den Tagesplan und den Trainingsfortschritt kann man sich ohne Eingabe eines Passwortes anschauen. Um die Trainingsergebnisse einzutragen oder den Trainingsplan zu ändern, ist ein Passwort nötig. Jeder der Funktionen, sowie die Passwortabfrage, wird im folgenden genauer erläutert.

Trainingsplan

Über den Button Trainingsplan gelangt man in die nächste Ansicht, welche den gesamten aktuellen Trainingsplan darstellt, siehe Abbildung. Um den Trainingsplan zu laden, ruft das Programm die Funktion CardInterface.getWorkoutplan() aus dem CardInterface auf, welche als Rückgabewert den aktuellen Triningsplan liefert. Dieser wird im Anschluss ausgelesen und in die Tabelle geschrieben. Wenn der Trainingsplan leer ist, ist folglich auch die Tabelle leer. Über den Zurück Button gelangt man, wie auch in allen anderen Funktion, zurück in das Hauptmenü.

Tagesplan

Die Ansicht des Tagesplan ist ähnlich zur Anzeige des gesamten Trainingsplans, mit einer Einschränkung. Über die Combobox Day kann der gewünschte tag ausge-

wählt werden. Nach dieser Auswahl wird in der Tabelle, welche den Trainingsplan anzeigt, nur diejenigen Übungen eingetragen und dargestellt, welche an dem jeweiligen Tag ,laut Trainingsplan, stattfinden sollen. Dazu besitzt jede Stage in einem Workoutplan-Object ein Attribut day. Diese Ansicht ist in Abbildung zu sehen.

Trainingsfortschritt

Login: Sportler

Login: Trainer

5 Zusammenfassung und Ausblick

Der beschriebene Anwendungsfall wurde im Rahmen dieses Belegs analysiert und eine Lösung implementiert.

6 Quellenverzeichnis

Quellenverzeichnis

[1] Prof. Dr. rer. nat. Petermann, Uwe: *Lehrmaterial zur Veranstaltung Smartcard-Programmierung im SS2014 an der HTWK-Leipzig.* https://bildungsportal.sachsen.de/opal/auth/RepositoryEntry/437649412/CourseNode/87246817112798
intern abrufbar am 10.Juli 2014

[2] Sun Microsystems, Inc.: *Java Card Applet Developer's Guide* http://www.oracle.com/technetwork/java/javacard/downloads/index.html
abrufbar am 01.Juli 2014

[3] International Organization for Standardization: *ISO 7816 - Identification cards – Integrated circuit cards* kostenpflichtig abrufbar unter http://www.iso.org/