Open Card Application Framework - ein objektorientiertes Framework zur visuellen Erstellung von Smartcard-Anwendungen

Thomas Stober[1], Thomas Schäck[1], Lothar Merk[1]

[1] IBM Deutschland Entwicklung GmbH, Global Smart Card Solutions,
Schönaicher Straße 220, 71032 Böblingen, Deutschland
tstober@de.ibm.com, schaeck@de.ibm.com, lmerk@de.ibm.com

Abstract. In diesem Artikel wird eine Architektur für die visuelle Entwicklung von Smartcard Lösungen vorgestellt. Der Einsatz von Java und der Java Beans Technologie ermöglicht es, Smartcard Anwendungen intuitiv und schnell in einer visuellen Entwicklungsumgebung zu erstellen. Komplexe Lösungen können auf einfache Weise aus flexiblen Komponenten zusammengefügt werden, ohne daß der Anwendungsprogrammierer über detailliertes Smartcard Know-how verfügen muß. Eine klare Aufteilung in Benutzerschnittstelle, Anwendungslogik und kartenbezogenen Teil von Anwendungen, die auf diesem Framework basieren, gewährleistet die Austauschbarkeit von hersteller- und kartenbezogenen Komponenten. Entwickelte Lösungen sind in der Lage, mit verschiedensten Typen von Smartcards zu arbeiten - von den heute verbreiteten dateiorientierten Smartcards bis hin zu Java Cards.

1 Einleitung

Seit 1984 in Frankreich eine Chipkarte als Telefonkarte eingesetzt wurde, hält eine rasante Entwicklung des Chipkartengeschäftes an - sowohl aus wirtschaftlicher als auch aus technologischer Sicht. Bis zum Jahr 2000 wird ein jährliches Wachstum von 25 Prozent prognostiziert. Allein die Geldkarte wurde auf über 40 Millionen EC-Karten verteilt. Aus einfachen Speicherchips sind moderne Chips mit eigenem Mikroprozessor und eigenem Betriebssystem geworden, die über bis zu 32 Kilobyte freien Speicher für Anwendungen verfügen. Zuverlässige Authentisierung, elektronische Unterschrift und Kryptographie sind Aufgabenfelder in denen Smartcards herkömmlichen Technologien wie beispielsweise Magnetstreifenkarten deutlich überlegen sind. Durch den integrierten Prozessor können Smartcards unabhängig und unbeeinflußt von der Außenwelt selbständig Operationen durchführen: beispielsweise das Kennwort eines Benutzers überprüfen, bevor der Zugriff auf gespeicherte Daten der Karte freigegeben wird oder kryptographische Algorithmen ausführen, so daß sich sensible Daten - wie z.B. einer geheimer Schlüssel - zu keinem Zeitpunkt außerhalb der Karte befinden müssen. Dadurch lassen sich

beispielsweise Zahlungen ohne die teure Online-Verbindungen zu Hostrechnern realisieren. Dies kann auf einer Smartcard lokal geschehen, was insbesondere im Bereich des Electronic Commerce eine wesentliche Rolle spielt.
Ein anderer Wachstumsmarkt für Smartcards ist die Konsumelektronik: Handys und Settopboxen arbeiten schon heute mit Smartcards. Auch eine neue Generation von Computern, die auf dem ersten Blick nicht unbedingt als solche erkennbar sind, werden Smartcard Technologien nutzen - vielleicht ein ins Handy integrierter Internetbrowser.

1.1 OpenCard Framework

Smartcard Betriebssystem und Terminal-API sind meist proprietäre Entwicklungen der Karten- bzw. Terminalhersteller. Dies hat zur Folge, daß Smartcard -nwendungen auf spezifische Kartentypen abgestimmt werden müssen, was die Einführung neuer heterogener Lösungen unter Beteiligung vielfältiger Hersteller und Anwender erheblich erschwert. Dieses Hindernis führte zu einem Standardisierungsprozeß, den das OpenCard Konsortium vorantrieb und der in das *OpenCard Framework* mündete. Das OpenCard Framework (kurz OpenCard oder OCF) verbindet Komponenten der verschiedener Hersteller und macht ihre Funktionen über eine standardisierte Schnittstellen zugänglich. Um Interoperabilität zwischen der Komponenten von Smartcard-Systemen zu erreichen, stellen die einzelnen Hersteller für die Integration ihrer Karten bzw. Kartenleser Softwarebausteine mit definierten Schnittstellen zur Verfügung. Um auch auf Seite des Hostsystems plattformneutral zu sein, wurde für die Implementierung des Frameworks Java gewählt. Damit können OpenCard Anwendungen prinzipiell auch auf Systemen der Konsumelektronik Einsatz finden für die eine Java Virtual Machine verfügbar ist.
Dem OpenCard-Konsortium gehören zur Zeit die Firmen Bull, Dallas Semiconductor, First Access, Gemplus, IBM, NCI (Network Computer Inc.), Netscape, Schlumberger, SCM Microsystems, Sun Microsystems, Ubiq und Visa an (Abbildung 1). Die aktuell verfügbare Version 1.1 der Referenzimplementierung ist im Internet frei verfügbar [2].
Die Architektur von OpenCard ist in Abbildung 2 dargestellt. Kern des OpenCard sind definierte Schnittstellen für CardTerminal-Klassen und CardService-Klassen:

- Terminalhersteller können den Einsatz ihrer Lesegeräte durch Implementierungen des OpenCard Interfaces `CardTerminal` unterstützen, die die gerätespezifischen Eigenschaften kapseln. Zur Laufzeit einer Anwendung werden dann von OpenCard die zu den angeschlossenen Terminals passenden Implementierungen instantiiert.
- Kartenhersteller implementieren für ihre Karten `CardServices`. Dazu werden die von OpenCard API angebotenen Funktionen in Kommandos kartenspezifischer Betriebssysteme umgesetzt. Die die Auswahl des richtigen CardServices erfolgt durch Konfiguration von OpenCard, ohne daß Anwendungsprogrammierer oder Endbenutzer sich um Kartentyp und dessen technische Details kümmern müssen. Erstellte Anwendungen auf Basis von OpenCard sind damit vollkommen unabhängig von den Eigenschaften der Karte und des Terminals.

Abbildung 1: Beteiligte Unternehmen im OpenCard Konsortium

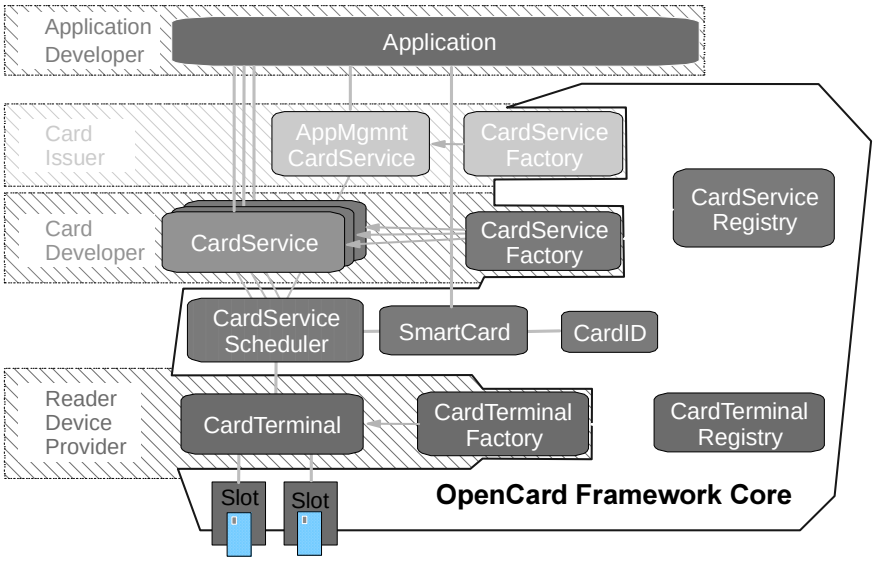

Abbildung 2: Das OpenCard Framework stellt Anwendungsentwicklern eine Schnittstelle für den Kartenzugriff zur Verfügung und erreicht durch standardisierte Schnittstellen für CardTerminal-Klassen und CardService-Klassen Interoperabilität zwischen verschiedenen Kartenlesern und Karten [2]

1.2 Open Card Application Framework

Das OpenCard Framework bietet ein high-level API für die Entwicklung von SmartCard Zugriffen. Der Entwickler benötigt keine detaillierten Kenntnisse über die verschiedenen Kartenleser Schnittstellen oder die Protokolle der Smartcard Betriebssysteme.
Um die Anwendungsentwicklung weiter zu erleichtern wird im Böblinger Entwicklungslabor der IBM das *Open Card Application Framework* (OCAF) entwickelt. Dieses Framework schließt die Lücke zwischen dem OpenCard API und visueller Programmierung. Es stellt handliche Komponenten zur Verfügung mit denen wesentliche Teile von SmartCard-Anwendungen abgedeckt werden können. Dadurch wird die Entwicklung von komplexen Smartcard Lösungen deutlich einfacher und effizienter. Zusätzlich zur Plattformneutralität und zur Unabhängigkeit von Karte und Terminal kommen nun die Vorzüge der visuellen Programmierung zur Geltung (Abbildung 3): Vorbereitete Komponenten des OCAF können in die Arbeitsoberfläche eines visuellen Entwicklungswerkzeuges eingefügt und intuitiv mit Elementen einer graphischer Benutzungsoberfläche verbunden werden. Anschließend generiert das Entwicklungswerkzeug lauffähigen Java-Code.

Abbildung 3: Die Prinzipien des Open Card Application Frameworks: plattformneutral, kartenunabhängig sowie für die visuelle Programmierung geeignet.

Diese Vereinfachung der Anwendungsprogrammierung ist insbesondere im Hinblick auf die zunehmende Komplexität vom SmartCard Lösungen von Bedeutung. Längst umfaßt eine Applikation mehr als nur einzelne Kartenzugriffe: „SmartCard Solutions" umfassen die Integration mehrerer SmartCard Anwendungen (beispielsweise die Unterstützung von Kredit- und Kundenkarte), die Anbindung an Netzwerke und Back-Ende Systeme (beispielsweise an das Reservierungsystem einer Fluggesellschaft) sowie die komplexen Kartenverwaltungssysteme der Kartenherausgeber.
In den folgenden Kapiteln soll das Open Card Application Framework vorgestellt werden. Zunächst werden die Anforderungen definiert, die aus Sicht von Anwendungsenwicklern an ein Framework für Smartcard-Anwendungen gestellt werden. In Kapitel 3 wird die Architektur, durch die sich diese Anforderungen erfüllen lassen erläutert. Den Schwerpunkt dieses Artikels bildet die Darstellung eines Entwicklungsprozesses für Smartcard-Anwendungen auf Basis des OpenCard Application Frameworks in Abschnitt 4. Ein Beispiel für die Verwendung des OpenCard Application Framework verwendet wird, um durch visuelle Programmierung auf einfache Weise eine Kartenanwendung zu erstellen rundet dieses Kapitel ab. Kapitel 5 faßt die Ergebnisse der Entwicklungsarbeit am Open Card Application Framework zusammen.

2 Anforderungen

Im Vorfeld unserer Arbeit haben wir mit Anwendungsentwicklern und Architekten gesprochen, um die wesentlichen Anforderungen an ein Framework für Smartcard-Anwendungen zu sammeln. Folgende Anforderungen erscheinen uns am wichtigsten:

- *Unterstützung von Anwendungsauswahl und Anwendungsinstantiierung* - Moderne Smartcards können mehrere Anwendungen unterstützen; denkbar ist etwa eine Smartcard die als elektronische Visitenkarte, Notizbuch und Kalender verwendbar ist. Die Anwendungen auf einer Smartcard können von verschiedenen Herstellern stammen. Es wird daher ein Verfahren benötigt, um festzustellen welche Anwendungen von der Karte unterstützt werden, die Anwendungen bei Bedarf über ein Netzwerk zu laden und schließlich zu instantiieren.
- *Modularität der Anwendungen* - Die arbeitsteilige Entwicklung von Applikationen sollte durch eine klare Trennung von Benutzerschnittstelle, Anwendungslogik und kartenbezogenem Teil unterstützt werden. Durch diese Abtrennung ist der kartenbezogene Teil austauschbar, die Anwendungslogik hängt nicht von dessen Implementierung und der zugrundeliegenden Plattform ab. Die Abtrennung der Benutzerschnittstelle ist erforderlich, um Herstellern komplexer Anwendungssysteme die Integration von Kartenanwendungen anderer Hersteller unter ihrer eigenen Benutzerschnittstelle zu gestatten.
- *Einfacher Einstieg und Erweiterbarkeit* - Um die Hürde beim Einstieg möglichst niedrig zu halten, sollten keine tiefergehenden Kenntnisse über Smartcards

notwendig sein. Es muß möglich sein, einfache Anwendungen durch visuelle Programmierung aus einigen Standardbausteinen zusammenzusetzen. Um andererseits auch die Realisierung komplexerer Anwendungen zu erlauben, muß ein Smart Card Application Framework erweiterbar sein, so daß sich selbstprogrammierte Bausteine einfach einfügen lassen.

- *Ereignismodell* - Anwendungen müssen über relevante Aktionen informiert werden. Wenn eine Karte herausgezogen wird, müssen die verbundenen Anwendungen benachrichtigt werden, so daß sie terminieren oder Kartenzugriffe bis auf weiteres unterdrücken können. Wenn eine Karte eingesteckt wird müssen bereits existierende Anwendungen benachrichtigt und damit reaktiviert werden.
- *Interaktion mehrerer Anwendungen* - Es soll möglich sein, daß Anwendungen interagieren. Als Beispiel dafür mag ein Übertragungsvorgang von einer elektronischen Börse auf eine andere dienen: Es werden zwei Börsenkarten in Leser eingesteckt, wodurch zwei Anwendungsinstanzen erzeugt werden, die sich über das Framework gegenseitig lokalisieren und dann interagieren.
- *Unterstützung verschiedener SmartCard Technologien* - Es sollen sowohl unterschiedliche dateisystembasierte Karten, als auch Java Cards durch das Framework berücksichtigt werden.
- *Durchgängiger visueller Entwicklungsprozeß* - Bereitstellung von Werkzeugen, die das Design von Karteninhalten, Applets und die darauf basierende Erzeugung von Java Beans für die visuelle Entwicklung von Anwendungen ermöglichen.

3 Architektur

Eine Referenzimplementierung des Open Card Application Frameworks baut auf dem OpenCard Framework auf. Die einzelnen Komponenten für den Aufbau von Smartcard Anwendungen werden den Anwendungsentwicklern in Form von JavaBeans und Java Interfaces zur Verfügung gestellt, die in einer visuellen Entwicklungsumgebung zusammengefügt werden können (Abbildung 4). Grundsätzlich gibt es Komponten, die anwedungsunabhängig sind, und Komponten, die auf eine bestimmte Anwendung hin zugeschnitten sind. Um dem Nutzer eine manuelle Programmierung der anwendungsspezifischen Komponenten zu ersparen, können diese mit Hilfe zugehöriger Werkzeuge automatisch generiert werden. Um die Anwendung des Frameworks zu erleichtern wird für darauf aufbauende Anwendungen eine klare Architektur vorgegeben.

3.1 Grundsätzlicher Aufbau einer Smartcard-Host-Anwendung auf Basis des Application Frameworks

Bei der Entwicklung von Smartcard Anwendungen gibt es gewöhnlich drei grundsätzliche Aufgaben: Das Design der Benutzerschnittstelle, die Implementierung der Anwendungslogik und die Implementierung des kartenbezogenen Teils. Diese Dreiteilung findet sich in der Architektur wieder, die das Open Card Application

Framework allen Anwendungen vorgibt, die auf dem Framework aufbauen sollen (Abbildung 5):

- Kartenbezogener Teil - Dieser Teil repräsentiert Objekte auf der Smartcard, sowie Methoden, die von diesen Objekten unterstützt werden.
- Benutzerschnittstelle - Dieser Teil ermöglicht die Interaktion zwischen Benutzer und Anwendungslogik - zum Beispiel durch eine graphische Benutzerschnittstelle, Spracherkennung, Pinpad oder Tastatur.
- Anwendungslogik - In diesem Teil werden Abläufe und Ereignismodelle der Anwendung implementiert.

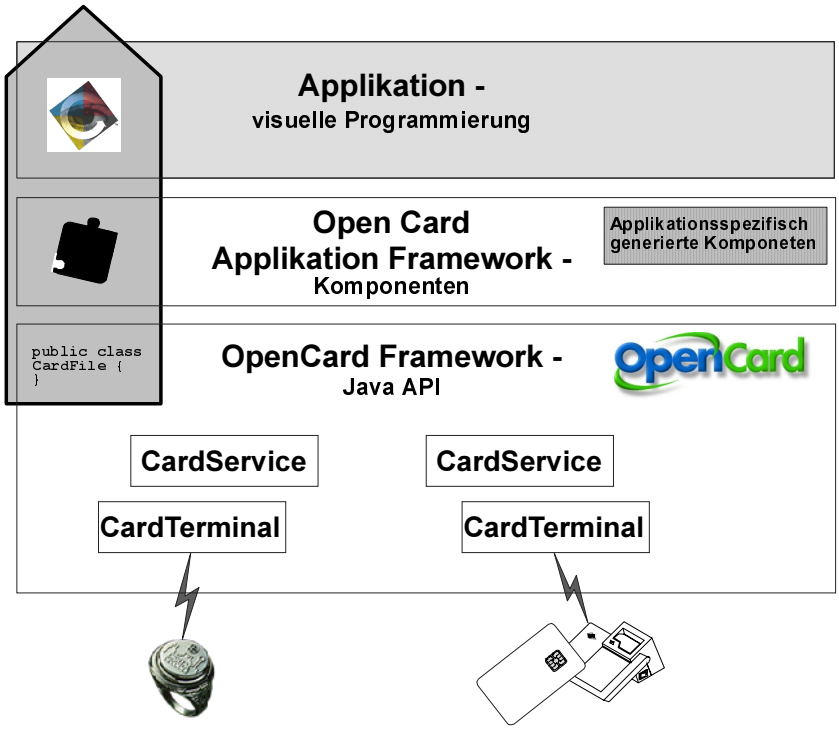

Abbildung 4: Eine mit dem Application Framework erstellte Smartcard Anwendung besteht aus drei Programmierebenen: Das OpenCard Framework wird benötigt, um auf die von der Karten unterstützte Funktionalität zuzugreifen. Darauf baut das Application Framework auf, das dem Anwendungsentwickler High-level Komponenten zur Verfügung stellt. Die oberste Schicht stellt die Ebene der visuelle Programmierung dar, bei der die eigentliche Anwendungssoftware zusammengefügt wird.

Kartenbezogener Teil

Um die Anwendung von einer spezifischen Karte so unabhängig wie möglich zu machen, werden eine Reihe von Interfaces zur Verfügung gestellt, mit denen die auf einer zur Anwendung gehörenden generischen Karte befindlichen Objekte (*Items*)

repräsentiert werden. Derzeit unterstützte Items sind u.a. primitive Java Typen, Arrays, Strukturen, kryptographische Schlüssel und Zähler. Interfaces definieren für diese Item-Typen die verfügbaren Methoden und können für die visuelle Programmierung von Smartcard-Zugriffen verwendet werden. Gleichzeitig verbergen sie kartenbezogene Details, die für den Anwendungsprogrammierer nicht von Bedeutung sind: dazu gehören interne Eigenschaften der wie Dateipfade, Offsets, Adressen oder ähnliches, die sogar bei einzelnen Karten gleichen Typs unterschiedlich sein können, aber natürlich auch kartentypabhängige Eigenschaften.

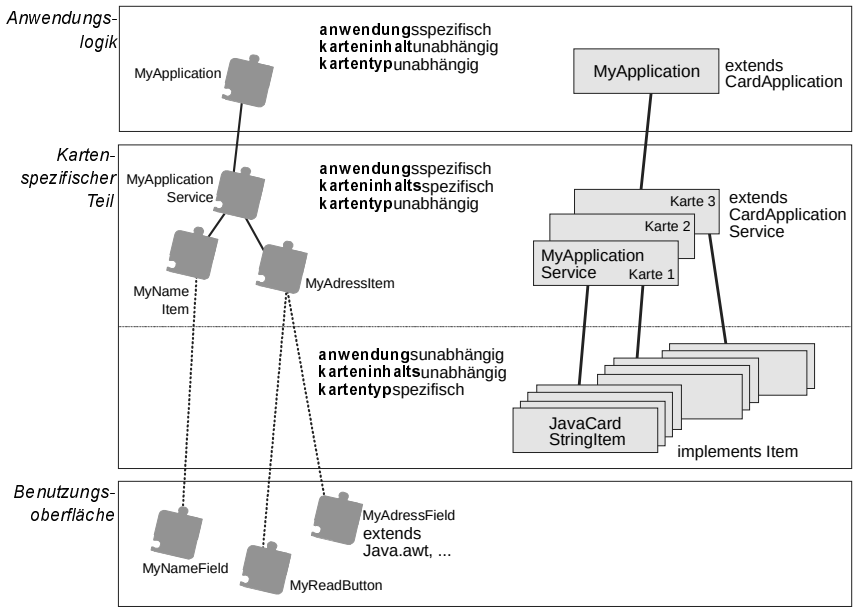

Abbildung 5: Aufbau einer Smartcard Anwendung auf Basis des Open Card Application Framework: Das Application Framework gibt Anwendungen eine bestimmte Architektur vor, die Benutzerschnittstelle und Applikationslogik vom kartenbezogenen Teil trennt. Der kartenbezogene Teil baut dabei auf dem OpenCard Framework auf und ermöglicht den Zugriff auf die Smartcard. Unterschieden wird zwischen karteninhaltsspezifisch und kartentypspezifisch: karteninhaltsspezifisch sind bestimmte Items auf einer Karte, die zu einer Anwendung gehören. Kartentypspezifisch sind anwendungsunabhängige Implementierungen dieser Items für einen bestimmten Kartentyp. Links sind die für den Anwendungsprogrammierer innerhalb einer visuellen Programmierumgebung sichtbaren Komponenten abgebildet. Rechts sind deren Repräsentation durch Java-Klassen zu sehen, die von den Framework-Klassen `CardApplication`, `CardApplicationService` abgeleitet sind, bzw. ein `Item`-Interface implementieren,

Für jede Anwendung muß eine Java-Klasse erstellt werden, die sämtliche Items, die sich auf der zu der Anwendung gehörenden Smartcard befinden, definiert und die von der abstrakten Klasse `CardApplicationService` abgeleitet ist, die vom

Application Framework bereitgestellt wird. Dazu ist ein graphischer Editor verfügbar mit dem die Items auf der Smartcard definiert werden können und der die zugehörige Card Application Service-Klasse automatisch generiert.
Aufgabe dieses Card Application Service ist es zur Laufzeit, abhängig von der verwendeten Karte die abstrakten Item-Interfaces durch eine konkrete Implementierung zu ersetzen. Diese muß dafür sorgen, die High-level Items der generischen Smartcard in die auf der tatsächlichen Smartcard verfügbaren Datentypen umzusetzen.
Da alle kartenbezogenen Operationen innerhalb solcher Implementierungen gekapselt werden, kann die gleiche Anwendung mit verschiedenen Kartentypen - wie etwa etwa Java Card oder einer herkömmlichen dateiorientierten Smartcard - arbeiten. Der Card Application Service muß lediglich eine andere Implementierung der Item-Interfaces wählen. Die kartenspezifischen Implementierungen der Items sind anwendungsunabhängig und müssen daher je Kartentyp nur einmal bereitgestellt werden.

Benutzerschnittstelle
Die Schnittstelle zum Anwender kann aus gewöhnlichen AWT Komponenten wie Textfield, Textarea, Panel, Button, etc. zusammengesetzt werden. Das Opencard Application Framework stellt zusätzliche Komponenten zur Verfügung, die für Smartcard-Lösungen typische Bedienteile wie Pinpad oder LCD-Display zugänglich machen. Die einzelnen Komponenten der Benutzerschnittstelle können durch Austausch von Ereignissen mit den definierten Item-Interfaces kommunizieren. Die relevanten Ereignisse werden in einer visuellen Programmierumgebung dabei graphisch mit den gewünschten Methoden der jeweiligen Items verbunden.

Anwendungslogik
Die Anwendungslogik wird durch eine weitere anwendungsspezifische Komponente vorbereitet, die von der Klasse `CardApplication` abgeleitet ist. Diese Komponente startet bzw. beendet den kartenbezogenen Teil der Anwendung und verwaltet den Anwendungsstatus. Sie übernimmt außerdem die Kommunikation mit der System-Applikation des Open Card Application Frameworks. Falls die Anwendung mit mehreren Karten gleichzeitig arbeitet, synchronisiert diese Card Application Komponente die Card Application Services der verschiedene Karten.

3.2 Die System-Applikation des Open Card Application Framework

Die einzelnen Anwendungen werden von einer System-Applikation des Application Frameworks gestartet und synchronisiert. Die System-Applikation stellt fest, welche Anwendungen eine eingeführte Karte unterstützt. Abhängig von Informationen auf der Karte und den Konfigurationsdaten des Systems können bestimmte Anwendungen automatisch aufgerufen werden (Abbildung 6). Sind benötigte Anwendungen nicht vorhanden, können sie über ein Netzwerk geladen werden. Auf diese Weise kann ein Gesamtsystem zur Laufzeit um zusätzliche Anwendungsteile ergänzt werden.

Außerdem wird es dadurch möglich, einzelne Anwendung separat zu entwickeln werden und zur Laufzeit zu einem bestehenden System nachträglich hinzuzufügen.
Das Opencard Application Framework stellt hierfür die Klasse `CardApplicationSystem` zur Verfügung.

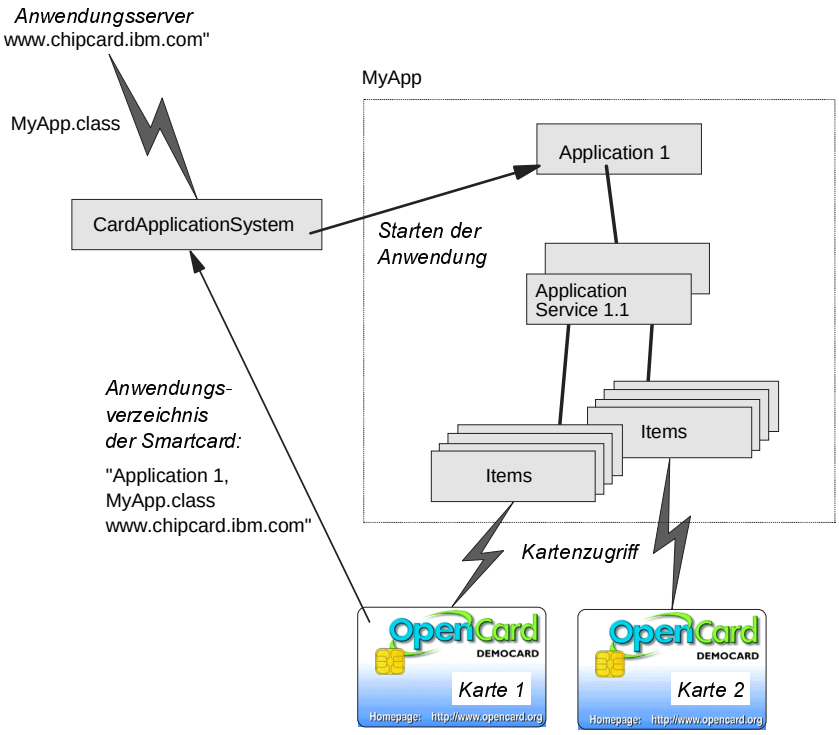

Abbildung 6: Eine übergeordenete System-Applikation `CardApplicationSystem` erlaubt es, nach Einführen einer Smartcard, in Abhängigkeit von auf der Karte gespeicherten Informationen verschiedene Anwendungen automatisch zu starten. Die Anwendung kann sich dabei auf dem Host selbst befinden oder über Internet von einem Server geladen werden, dessen Adresse auf der Smartcard gespeichert ist.

4 Entwicklung von Smartcard Anwendungen

Das Opencard Application Framework basiert auf Java Beans, die in einer visuellen Programmierumgebung zu individuellen Anwendungen zusammengebaut werden können. Zusätzliche Werkzeuge ermöglichen die durchgängige Entwicklung von Smartcard-Anwendungen von der Festlegung des Kartenlayouts bis hin zum Entwurf der Benutzerschnittstelle. Erwähnenswert ist in diesem Zusammenhang der *Card Application Wizard*, ein Tool zur visuellen Definition von Smartcard-Inhalten [8], sowie das *IBM Smartcard Toolkit*, das ausgehend von einer Beschreibung des

Karteninhalts die Produktion von Karten erlaubt [3]. Die Anwendungsentwicklung erfolgt in zwei Schritten, die in Abbildung 7 dargestellt werden:

- Entwicklung des Anwendungsteils auf dem Rechner: Implementieren der anwendungsspezifischen Beans, Auswählen der benötigten Frameworkbeans und visuelle Komposition der Anwendung.
- Entwicklung des Anwendungsteils auf der Karte: abstrakte Beschreibung der korrespondierenden Daten auf der Smartcard durch eine deklarative Sprache. Erzeugen der Produktionsdaten aus dieser Beschreibung und Produktion der Karten.

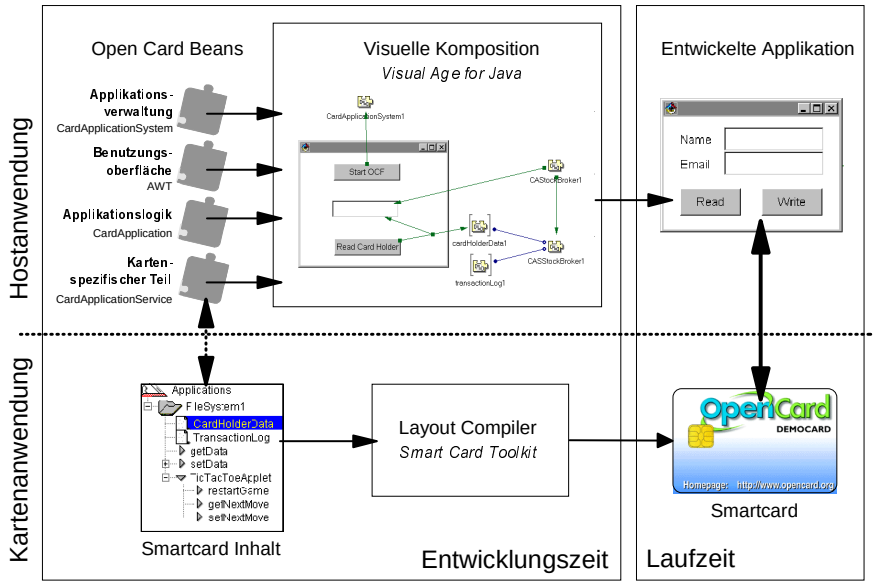

Abbildung 7 : Anwendungsentwicklung mit Hilfe des Opencard Application Frameworks und des IBM Smart Card Toolkits: Durch visuelle Programmierung werden Framework-Bausteine zu einer Anwendung zusammengesetzt. Aus einer abstrakten Beschreibung der korrespondierenden Kartendaten werden Daten für die Kartenproduktion erzeugt.

4.1 Entwicklung des Anwendungsteils auf der Karte

Die Entwicklung des Anwendungsteils auf der Karte wird durch den Card Application Wizard [8] unterstützt. Dieser generiert sowohl die Komponenten für den visuellen Zugriff auf die Karteninhalte, als auch die Definition der Objekte, die auf der Karte abgelegt werden. Zusätzlich können durch das IBM Smart Card Tookit die für die Produktion der Karte erforderlichen Daten erzeugt werden. Die Produktion der Karten kann durch ein einfaches Initialisierungstool oder im Rahmen eines Massenproduktionsprozesses erfolgen. Mehr Information zu diesem Thema ist in [3] zu finden

4.2 Entwicklung des Anwendungsteils auf dem Rechner

Die Entwicklung der Anwendungsteils, der auf dem Rechner läuft, erfolgt in drei Schritten. Zunächst sind anwendungsspezifische Beans zu entwickeln, welche die Anwendungslogik bzw. den kartenbezogenen Teil der Anwendung darstellen. Für einfache Anwendungen, die sich aus Standardkomponenten zusammensetzen lassen, kann dieser Schritt entfallen. Die Einzelteile werden mit Hilfe eines visuellen Werkzeugs zu einer Smartcard-Anwendung zusammengefügt. Schließlich werden so erstellte Anwendungen mit der Laufzeitumgebung des Opencard Application Framework verbunden.

Implementierung der anwendungsspezifischen Beans
Kartenbezogene Beans sind von der abstrakten Klasse CardApplicationService abzuleiten. CardApplication Services implementieren Methoden für den Zugriff auf Objekte auf der Karte. Objekte können Datenelemente, indizierte Datenelemente, Schlüssel oder anderes sein. Für jedes Objekt werden gemäß der Konventionen für Beans get- und set-Methoden implementiert, die von visuellen Entwicklungsumgebungen erkannt und dargestellt werden. Card Application Services können von Smartcard-Experten implementiert und von Anwendungsentwicklern in verschiedene Anwendungen integriert werden. Es ist auch denkbar, allgemein gehaltene Card Application Services als Templates zu verwenden, die bei der Anwendungsentwicklung nur noch modifiziert werden. Die Applikationslogik wird in Unterklassen der Frameworkklasse CardApplication implementiert. In diesen Klassen wird das Verhalten der Anwendung festgelegt, unter anderem die Reaktion auf Einführen oder Herausziehen von Karten oder Benutzeraktionen, die von der Benutzerschnittstelle gemeldet werden.

Visuelle Komposition der Anwendung
Zur Komposition der Anwendung fügt der Entwickler in einer visuellen Entwicklungsumgebung die selbst erstellten Java Beans mit bereits existierenden Beans zu einer Applikation zusammen. Hierbei werden als Icons dargestellte Beans durch Ereignisse verknüpft, die als Pfeile dargestellt werden.
Die Anwendungslogik kann zum Beispiel ein Label informieren, wenn sich ihr Zustand durch Einführen einer Karte ändert. Das Label kann die Zustandsänderung sichtbar machen, etwa indem es seinen Text auf "Karte eingesteckt" setzt. Umgekehrt kann ein Button ein Ereignis auslösen, woraufhin durch eine visuelle Verknüpfung eine Methode eines CardApplicationService Daten von der Karte liest und das Ergebnis in ein Textfeld schreibt. Zusätzlich zu den üblichen Java-Komponenten stellt das OpenCard Application Framework eine Reihe anwendungsunabhängiger Beans zur Verfügung, die in Anwendungen integriert werden können. Es ist zu beachten, daß eine Anwendung durchaus mehrere Kartenanwendungen unterstützten kann. Es müssen dazu für jede Kartenanwendung die entsprechenden anwendungsspezifischen Beans bereitgestellt und integriert werden

4.3 Werkzeuge für die automatische Generierung von anwendungsspezifischen Komponenten

Zusätzlich zum Open Card Application Framework wurde eine Reihe von Werkzeugen entwickelt, mit denen ein, in allen Phasen unterstützter, visueller Entwicklungsprozeß von Smartcard-Anwendungen möglich ist.
Um die Erstellung der anwendungsspezifischen Komponenten zu erleichtern, steht ein graphisches Werkzeug zur Definition von Anwendungsobjekten auf einer Karte zur Verfügung. Mit diesem Card Application Wizard wird eine abstrakte Beschreibung des gewünschten Karteninhalts erstellt, d.h. der Smartcard-Inhalt wird auf einer Meta-Ebene definiert und ist damit weder karten- noch terminalspezifisch [8]. Anhand der erstellten Metainformationen kann der erforderlichen Programm-Code der individuellen CardApplication und CardApplicationService Komponenten automatisch generiert und anschließend für die visuelle Programmierung verwendet werden. Zusätzlich können Informationen zum Initialisieren der Karten erzeugt werden, beispielsweise JavaCard Applets oder Dateien einer Dateisystem basierten Karte. Durch Exportfunktionen lassen sich diese Informationen auch nutzen, um ein Layout-File zu erstellen, welches beispielsweise vom IBM Smart Card Toolkit weiterverarbeitet werden kann (Abbildung 8 und 9).

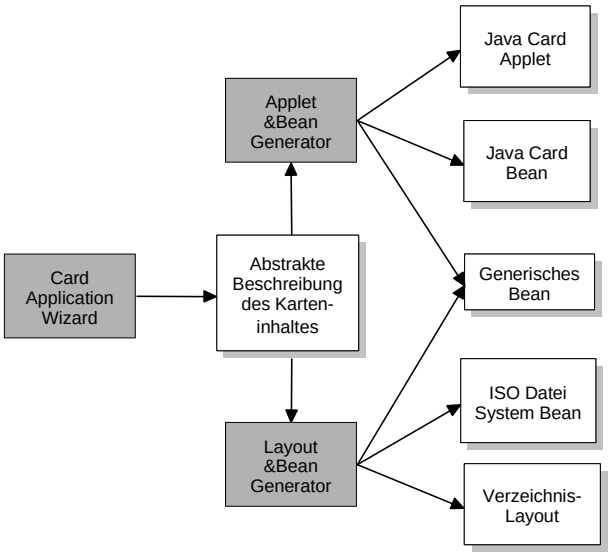

Abbildung 8: Automatische Erzeugung von Kartenobjekten und Java Beans zum Zugriff auf diese als anwendungspezifische Teile einer Smartcard-Anwendung: Mit Hilfe des Card Application Wizards wird eine abstrakte Beschreibung des Karteninhalts definiert. Aus dieser Beschreibung kann ein Generator JavaCard Applets oder Strukturen für dateiorientierte Karten, sowie die zum Zugriff auf die Karten erforderlichen JavaBeans erzeugen.

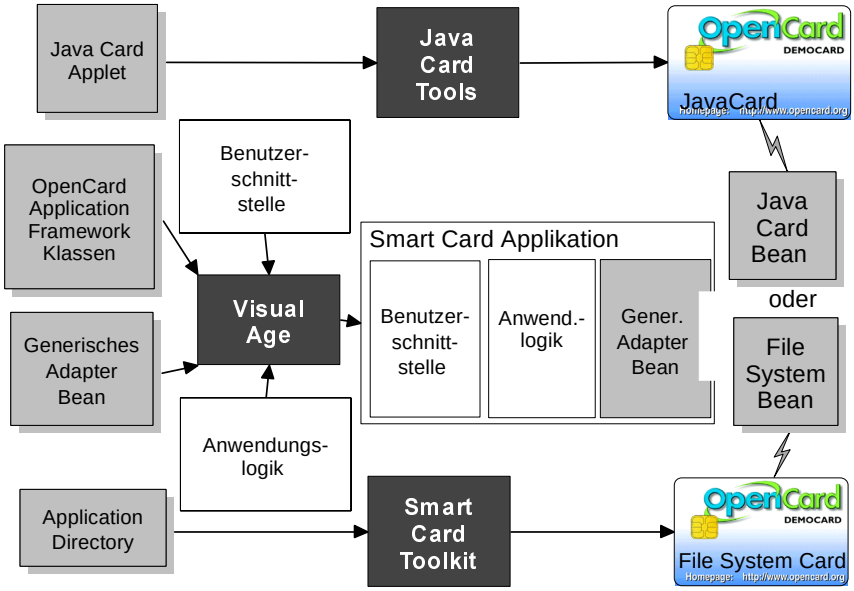

Abbildung 9: Integration der automatisch erzeugten Komponenten in eine Anwendung: Zuvor erzeugte JavaCard Applets werden durch JavaCard Tools auf die JavaCard geladen. Um Anwendungsverzeichnisse auf eine dateiorientierte Karte zu laden, wird das IBM Smart Card Toolkit verwendet

4.4 Referenzimplementierung

Zur Zeit existiert eine Referenzimplementierung des OpenCard Application Frameworks die auf dem OpenCard Framework für Smartcard-Anwendungen in Java basiert.

Für das visuelle Zusammenfügen der Komponenten wurden verschiedene Entwicklungsumgebungen evaluiert. Probleme bei der praktischen Arbeit sind nach unseren Erfahrungen insbesondere in der begrenzten Leistungsfähigkeit der verfügbaren visuellen Entwicklungsumgebungen zu sehen: Einige der visuellen Entwicklungsumgebungen implementieren die notwendige Unterstützung von Java Beans nur unzureichend, andere erlauben keine rein visuelle Programmierung sondern erfordern manuelle Nachbearbeitung des generierten Codes. Ferner sind viele der auf dem Markt befindlichen Produkte noch fehlerbehaftet.

Die in Abbildung 10 dargestellte Beispielanwendung wurde mit Visual Age for Java entwickelt, mit dem wir die besten Ergebnisse erzielt haben. Dieses Werkzeug verfügt über die umfangreichste Funktionalität der von uns benutzten Entwicklungsumgebungen.

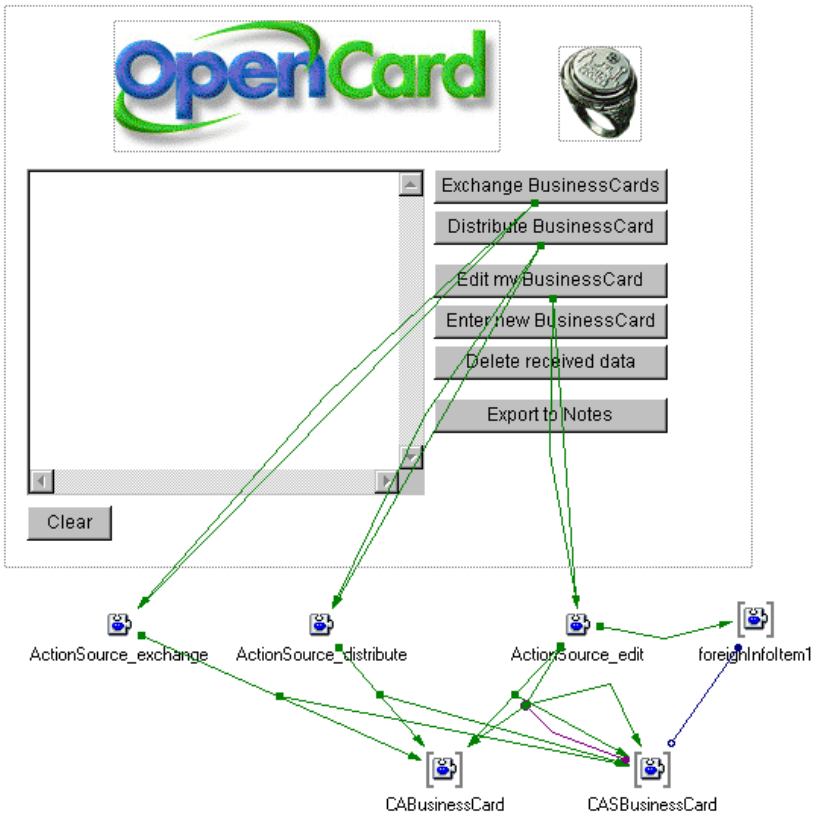

Abbildung 10: Beispiel einer rein visuell programmierten Anwendung zum Speichern und Austauschen von auf SmartCards gepeicherten Visitenkarten: Die Anwendung greift auf das Item `foreignInfoItem1` des Card Application Service Beans `CASBusinessCard` zu, um Visitenkarteninformationen von der Smartcard zu lesen. Abhängig vom Zustand des Applikation Beans `CABusinessCard` werden die Buttons aktiviert oder deaktiviert. Die `ActionSource` - Beans entkoppeln die Benutzungsoberfläche von Smartcard Zugriffe indem sie gesonderte Threads erzeugen [9].

5 Zusammenfassung

Das OpenCard Application Framework ist ein Beispiel dafür, wie mit Hilfe der Java Bean Technologie auf ein komplexes API einfache und gleichzeitig flexible Software-Bausteine aufgesetzt werden können. Diese Bausteine ermöglichen, durch visuelle Programmierung ohne tieferes Detailwissen über das API eines Smartcard-Frameworks und ohne lange Einarbeitung komplexe Anwendungen zusammenzusetzen.

5 Literaturverzeichnis

1. W. Rankl, W. Effing: Handbuch der Chipkarten. München; Wien: Hanser Verlag, 1996
2. OpenCard Consortium: The OpenCard Framework Whitepaper. 1998
   http:// www.opencard.org
3. IBM Deutschland Entwicklung GmbH: The IBM Smart Card Toolkit. 1998
   http://www.chipcard.ibm.com
4. ISO 7816, Teil 4: Identification Cards - Integrated circuit(s) cards with contacts. 1994
5. Sun Microsystems: Java Card 2.0 Language Subset and Virtual Machine Specification. 1997
   http://java.sun.com/products/javacar/
6. Sun Microsystems: JavaBeans 1.01 Specification,1997
   http://java.sun.com/beans/docs
7. IBM Deutschland Entwicklung GmbH: Open Card Application Framework - Programmers Guide. 1998
8. Vehns, Rainer: Card Application Wizard. Böblingen: 1998
9. Bröll, Thomas: Entwickeln einer Anwendung mit dem OpenCard Framework. Böblingen: 1998