Der Artikel gibt einen Überblick über die Entwicklung mobiler Applikationen mithilfe von ADF Mobile als Teil des Oracle Application Development Frameworks (ADF).

Mobile Erweiterung der Unternehmens-Applikationen mit ADF Mobile

Volker Linz und Dr. Jürgen Menge, ORACLE Deutschland B.V. & Co. KG

Nachdem es im Privatbereich bereits zahllose Applikationen (Apps) für mobile Endgeräte gibt, halten jetzt die mobilen Komponenten Einzug in die Welt der Unternehmens-Anwendungen. Im Gegensatz zu früher (in der Oracle-Welt meist auf Basis von „Oracle Lite“ und „WebToGo“) sind sie heutzutage häufig Bestandteil des Anforderungskatalogs.

Die Entwicklung kundenspezifischer Apps wird zumeist an externe Firmen vergeben, die sich auf mobile Endgeräte und die Darstellung von Multimedia-Inhalten spezialisiert haben. Demgegenüber möchte man die Entwicklung mobiler Komponenten als Teil der Unternehmenssoftware gern vom eigenen Entwicklungsteam umsetzen lassen, da es eine enge Integration zur bestehenden Infrastruktur und Software-Landschaft gibt. Wie schafft es aber ein bestehendes Entwicklerteam, sich neben der Betreuung der laufenden Applikationen in diese neuartige Thematik einzuarbeiten? Verschärft wird dies noch durch die Anforderung, möglichst mehrere Mobil-Plattformen (iOS, Android, Blackberry, Windows Mobile) zu unterstützen.

Entwickler, die bereits Oracle ADF kennen, haben es hier leichter, da ADF Mobile als zusätzliche Komponente des ADF-Frameworks das gleiche Programmiermodell wie für die Entwicklung von Web-Applikationen verwendet. Es ermöglicht die Entwicklung sogenannter „hybrider Applikationen“, die die Vorteile Browser-basierter Applikationen mit denen von Applikationen, die nativ auf dem Mobilgerät laufen, verbinden (siehe https://blogs.oracle.com/fusionmiddleware/entry/developer_s_corner_developing_mobile). Die Unterstützung einer großen Zahl unterschiedlicher Mobil-Plattformen und -Geräte wird durch die Verwendung des Open-Source-Frameworks „PhoneGap“ (http://phonegap.com/) innerhalb von ADF Mobile erreicht.

Die Anwendungs-Architektur von ADF Mobile

Die hybride Anwendung läuft in einem separaten Applikations-Container auf dem mobilen Endgerät (siehe Abbildung 1). Die native Applikationsschicht unterstützt dabei folgende Funktionen:

- Die gerätespezifische Darstellung, die an das Look & Feel des Geräts angepasst ist
- Die Anzeige von Web-Inhalten (Content) innerhalb der Applikation (Web View)
- Die Integration mit spezifischen Gerätediensten wie GPS, Kamera oder Kontakte

Die Besonderheit an dieser Architektur ist, dass neben der Web View und dem gerätespezifischen JavaScript-API für PhoneGap eine Java Runtime (CDC) in

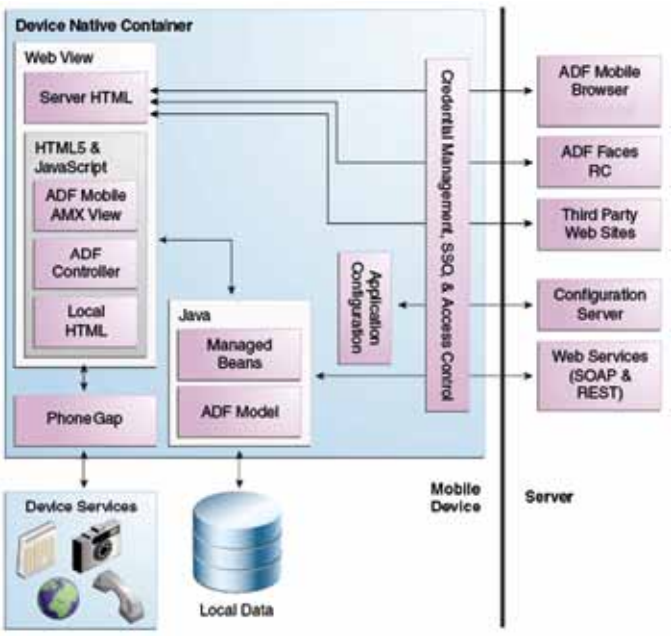

Abbildung 1: Architektur von ADF Mobile (Quelle: Developer Guide ADF Mobile)

den Anwendungs-Container integriert ist. In der Präsentationsschicht kommen Standard-Technologien auf Basis von HTML5, JavaScript und CSS zum Einsatz. Diese bilden die Basis, um Inhalte unterschiedlicher Herkunft innerhalb der Web View miteinander zu verbinden.

In ADF Mobile werden diese Inhalte als „Features" bezeichnet. Eine Anwendung wird aus mehreren Features kombiniert, die in der Datei „adfmf-feature.xml" zu definieren sind. Innerhalb der Web View können die folgenden Technologien miteinander kombiniert werden:

- *Server HTML (Remote URL)*
  Die Anwendungslogik befindet sich auf einem zentralen Server, der das HTML erzeugt
- *ADF Mobile XML (AMX Pages)*
  AMX-Seiten, die lokal ausgeführt werden und zur Laufzeit HTML5 und JavaScript erzeugen
- *Lokales HTML*
  HTML5-Seiten, die lokal auf dem Gerät ausgeführt werden
- *Native View*
  Inhalte, die speziell für eine bestimmte Plattform (etwa in Objective-C) entwickelt werden und damit nicht für diese Plattform zur Verfügung stehen

Die Entwicklung der Applikation erfolgt im Oracle JDeveloper, der um die Extension „ADF Mobile" erweitert ist. Erst zum Zeitpunkt des Deployment wird aus der Applikation eine ausführbare Anwendung erzeugt, indem man die plattformspezifischen Artefakte (Deployment-Deskriptoren, Metadaten, Projekt-Files etc.) für die jeweilige Plattform (iOS, Android) generiert und mit den notwendigen Bibliotheken zu einem Bundle zusammenfasst. Dabei werden die Werkzeuge aus dem SDK der jeweiligen Zielplattform genutzt.

Implementierungs-Pattern in ADF Mobile

Für den Zugriff auf Daten und Anwendungen innerhalb des Unternehmens werden für ADF-Mobile-Applikationen bestimmte Implementierung-Pattern empfohlen und durch entsprechende

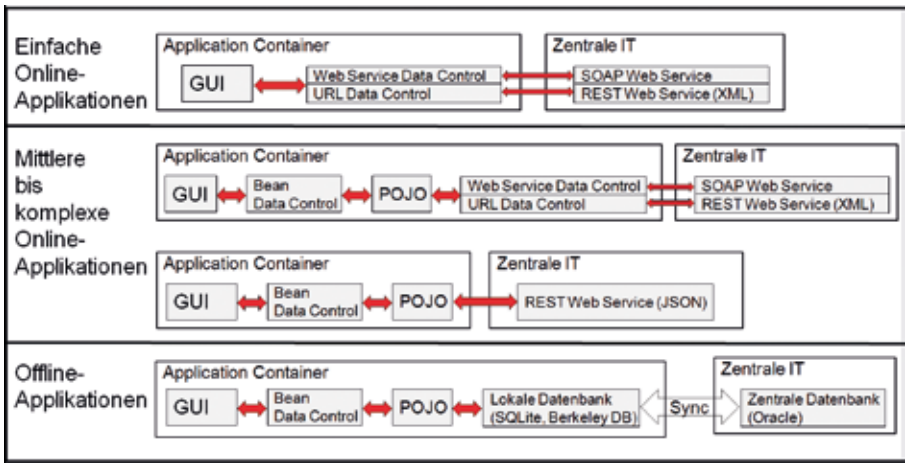

Abbildung 2: Implementierungs-Pattern in ADF Mobile

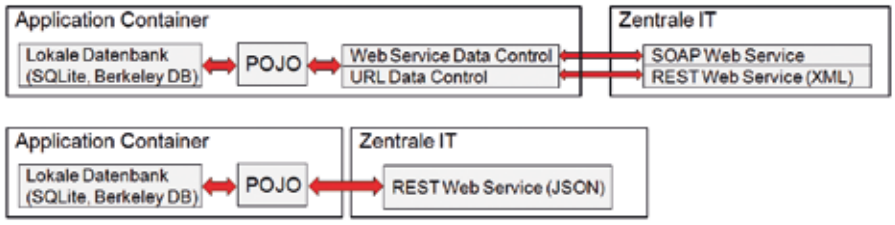

Abbildung 3: Synchronisation der lokalen Datenbank mit zentralen Systemen

Funktionalität des Frameworks unterstützt (siehe Abbildung 2).

Die Kommunikation mit der zentralen Unternehmens-IT erfolgt vorwiegend über Web-Services (SOAP und REST) und geht von der Annahme aus, dass durch eine verbesserte Netzabdeckung in den meisten Anwendungsfällen eine Online-Verbindung zur Verfügung steht. Für SOAP-basierte Web-Services (1.1, 1.2) können im JDeveloper die Web Service Data Controls auf Basis der WSDL erzeugt werden. Eine Absicherung der Web-Services durch Security Policies ist möglich.

REST-basierte Web-Services verwenden die Standard-HTTP-Methoden „GET", „POST", „PUT" und „DELETE". Die entsprechenden URL Service Data Controls können ebenfalls im JDeveloper generiert werden, wobei nur die Basic Authentication über HTTP/HTTPS unterstützt wird. REST-basierte Web-Services können alternativ über einen REST-Web-Service-Adapter, also ohne Data Control aufgerufen werden. Aufgrund der eingeschränkten Java-Funktionalität auf mobilen Endgeräten (Java ME) sind Web-Service-Proxies/-Clients nicht einsetzbar.

Interessant ist in diesem Zusammenhang auch die Möglichkeit, den ADF-Business-Components-Layer (ADFbc) einer zentralen Applikation für die Validierung der Daten im Back-End zu nutzen. Dies geschieht, indem die View Objects in ADFbc 11*g* als SDO-Services beziehungsweise in der nächsten Version von ADF 12*c* als REST-Services exponiert werden können.

Eine lokale Datenhaltung auf dem Endgerät ist als Zwischenspeicherung sinnvoll, wenn folgende Situationen eintreten:

- Die Verbindung zur zentralen Unternehmens-IT ist zeitweilig unterbrochen (Offline-Betrieb)
- Die Daten müssen mehrfach gelesen werden

Die lokale Datenbank fungiert in diesen Fällen als Cache. Lokale Datenänderungen sind mit den zentralen Systemen über Web-Services zu synchronisieren (siehe Abbildung 3).

Lokale Datenhaltung und Synchronisation

Als lokale Datenbank können auf dem mobilen Gerät „SQLite“ (siehe www.sqlite.org) oder die Berkeley DB (siehe http://www.oracle.com/technetwork/products/berkeleydb) zum Einsatz kommen. Da beide Datenbanken über ein identisches SQL-API (SQLite 3) angesprochen werden können, ist der Datenzugriff für die Anwendungslogik transparent. Während die SQLite-Datenbank nur eine schreibende und mehrere lesende Zugriffe erlaubt, sind bei der Berkeley DB mehrere konkurrierende Verbindungen möglich.

SQLite ist eine leichtgewichtige, relationale Datenbank (SQL92) mit ACID-Eigenschaften (Atomizität, Konsistenz, Isolation, Dauerhaftigkeit), die bei Verwendung in die Applikation eingebettet wird. Da SQLite keine Authentifizierung über Benutzer oder Rollen unterstützt, kann ADF Mobile den notwendigen Schutz der Daten über eine Verschlüsselung (Encryption) gewährleisten. Die lokale Datenbank kann entweder als Teil der Applikation ausgeliefert werden oder sie wird durch die Anwendung beim ersten Zugriff angelegt. Da im Anwendungs-Container weder die ADF Business Components noch TopLink beziehungsweise EclipseLink zur Verfügung stehen, muss der Entwickler die Datenzugriffe mittels JDBC selbst programmieren.

Zugriff auf Geräte-Funktionen

Das Open-Source-Framework „PhoneGap“ (Apache-Projekt Cordova) hat sich durch eine breite Plattform-Unterstützung bereits in der Praxis bewährt. Externe Applikationen beziehungsweise Web-Anwendungen, die Zugriff auf die Geräte-Funktionalitäten erhalten möchten, können nur mithilfe des JavaScript-API von PhoneGap darauf zugreifen.

Mit ADF Mobile besteht für lokale HTML5- beziehungsweise ADF-Mobile-AMX-Seiten die komfortable Möglichkeit, über ein spezielles Data Control (Device Data Control) auf die von PhoneGap angebotenen nativen Geräte-Funktionalitäten zuzugreifen. Dieses Data Control wird automatisch jeder ADF-Mobile-Applikation hinzugefügt. Vom Device Data Control werden folgende gerätespezifischen Funktionalitäten unterstützt:

- Geolocation Services (Maps)
- Adressbuch
- Kamera
- Zugriff auf Fotos
- E-Mail
- SMS
- Kontakte
- GPS

Sicherer Zugriff auf die mobile Applikation

Zur Authentifizierung beziehungsweise zur Autorisierung des Endanwenders für einzelne Features (kleinste abzusichernde Einheit in ADF-Mobile-Applikationen) werden JavaScript, ein PhoneGap-Plug-in und verschiedene native PhoneGap-Command-Handler genutzt. Diese Komponenten kümmern sich um folgende Aktionen:

- Interaktion zwischen Log-in-Seite und Teilen der Applikation (Features)
- Navigation zwischen den einzelnen Application Features einer mobilen Anwendung
- Interaktion zwischen Oracle Identity Connect IDM Mobile SDK, Oracle Access Manager (OAM) und Applikation

Die Klassen dieses Security-API (Oracle Identity Connect IDM Mobile SDK) stellen die Kommunikation und die Verifikation der Log-in-Daten sicher. Es gibt spezifische Client-seitige Security Services, um auch einen gesicherten Offline-Betrieb zu gewährleisten. Die Log-in-Daten sind dann abgesichert auf dem Gerät abgelegt. Die Log-in-Seite wird vom Framework bereitgestellt oder ist nach Kundenbedürfnissen individuell erstellbar. Der komplette Log-in-Prozess ist nativ, also ohne eingebettetes Java im ADF-Mobile-Framework möglich.

Ein Log-in ist erforderlich, wenn das Attribut „credential“ für ein Feature in der Konfigurationsdatei „adfmf-feature.xml“ gesetzt beziehungsweise das Timeout des gesicherten Features überschritten ist. In der genannten Datei wird auch festgelegt, ob die Authentifizierung „lokal“ oder „remote“ erfolgt.

Darüber hinaus werden dort auch die Berechtigungen und Rollen der Benutzer als sogenannte „Constraints“ abgelegt. Listing 1 zeigt ein Beispiel für ein Constraint.

Bei Verwendung einer eigenen Log-in-Seite ist in der Datei „adfmf-application.xml“ ein zusätzliches Tag eingefügt (siehe Listing 2). Zusätzlich kann die Verbindung (Connection) zum Authentifizierungsserver für jedes einzelne Feature in dieser Konfigurationsdatei definiert werden (siehe Abbildung 4). Die Default-Log-in-Seite kann als Vor-

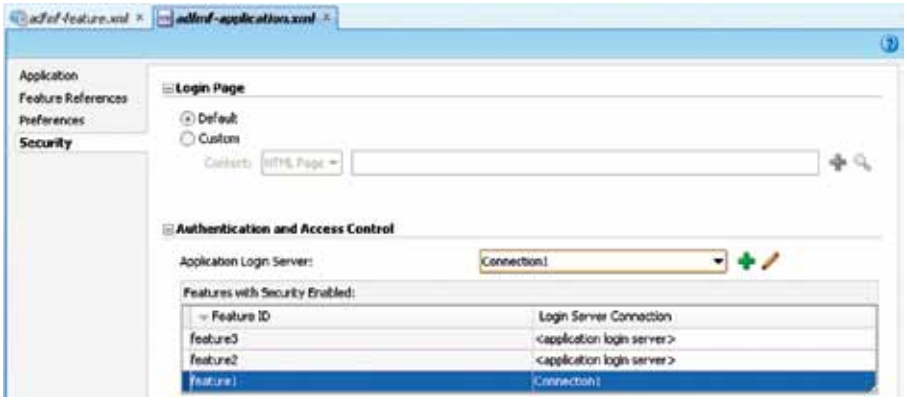

Abbildung 4: Konfiguration der Connections zum Authentifizierungsserver

```
<adfmf:constraints>
      <adfmf:constraint property="user.roles"
      operator="contains" value="manager_role"/>
      ...
</adfmf:constraints>
```

Listing 1

```
<adfmf:login defaultConnRefId="LoginConnection">
      <adfmf:localHTML url="newlogin.html"/>
</adfmf:login>
```

Listing 2

lage für die Erstellung eigener Log-in-Seiten dienen. Der Eintrittspunkt für den Authentifizierungsprozess eines Application Feature beginnt mit dem Event „activateLifecycle". Der Authentifizierungsprozess kann „lokal" auf dem mobilen Gerät oder „remote" gegen einen Authentifizierungsserver wie Oracle Access Manager (OAM) erfolgen. Das Framework ADF Mobile unterstützt jegliche Standard-konforme HTTP/HTTPS-Basic-Authentication.

Nachfolgend ist der Authentifizierungsprozess anhand der Berechtigungs- und Authentizitätsprüfung gegenüber dem Oracle Access Manager Server beschrieben, für den eine Verbindung (Connection) in der Datei „connection.xml" angelegt werden muss:

1. Das Framework präsentiert dem Anwender die Login-Seite (Standard- beziehungsweise selbst erstellte Seite), wenn die sicherheitsrelevanten Parameter in den Dateien „adfmf-application.xml" und „adfmf-feature-application.xml" gesetzt sind. In „adfmf-feature-application.xml" ist die Option „Enable Feature Security" gesetzt. Damit wird im Wizard der Datei „adfmf-application.xml" innerhalb der Registerkarte „Security" das abgesicherte Feature mit der dazugehörigen Verbindung zur Authentifizierungsstelle dargestellt.
2. Oracle Identity Connect IDM Mobile SDK APIs behandeln die Authentifizierung für beide Verfahren („lokal" gegen den Credential Store beziehungsweise „remote" gegen einen Authentifizierungsserver). Im Credential Store des Gerätes kann das gesicherte User-Objekt gespeichert sein, falls das Unternehmen eine lokale Speicherung der Log-in-Daten auf dem mobilen Endgerät erlaubt. Abhängig vom Erfolg beziehungsweise von einem Fehler bei der Authentifizierung liefert das API dem Framework ein User-Objekt oder eine Fehlermeldung.
3. Ist das Log-in erfolgreich, erhält ADF Mobile ein Token vom Oracle Access Manager (OAM), das im Cookie gespeichert und später in jeder Log-in-Connection verwendet wird. Dieses Cookie kann bei Authentifizierung gegenüber Web-Services (SOAP oder REST) direkt in den Web-Service-Aufruf injiziert werden (siehe http://docs.oracle.com/cd/E35521_01/doc.111230/e24475/security.htm#autoId10).
4. Oracle Identity Connect IDM Mobile SDK APIs speichern die Log-in-Daten im lokalen Credential Store des mobilen Endgeräts, sodass bei erneuter Authentifizierung auf den lokal gespeicherten Authentifizierungskontext im Credential Store zugegriffen werden kann. Damit ermöglicht das Framework eine gesicherte Offline-Funktionalität.
5. Schlägt das Log-in fehl, wird der Benutzer auf der Log-in-Seite festgehalten, damit weitere unerlaubte Interaktionen unterbunden werden.

Bei der Kommunikation der mobilen Anwendung mit der Unternehmens-Plattform (Middleware oder Back-End-Systeme) beziehungsweise mit Drittsystemen (wie Lieferanten-Plattformen) werden in der Regel Web-Services bevorzugt, die unter anderem eigene beziehungsweise öffentlich bestätigte Zertifikate für die Authentifizierung benötigen. Diese Zertifikate müssen in den Certificate Store „cacerts" unterhalb der Applikation im Ordner „resources/Security" importiert sein. Ein Import der Zertifikate kann mithilfe des Java-SE-Tools „keytool" erfolgen: „keytool -importcert -keystore cacerts -file new_cert -storepass changeit-noprompt".

Fazit

Oracle ADF Mobile zielt auf den Bereich von Unternehmens-Anwendungen, die als hybride Anwendungen auf mobilen Endgeräten sowohl im Online- als auch im Offline-Modus bereitgestellt werden können. Derzeit sind die im Markt dominierenden Mobil-Plattformen iOS und Android unterstützt. Die Entwicklung der mobilen Anwendung mit ADF Mobile folgt dem gleichen Programmiermodell wie die Entwicklung von ADF-Web-Anwendungen für den Desktop. Erst zum Zeitpunkt des Deployment wird aus der entwickelten Anwendung ein Plattform-spezifisches, ausführbares Paket erzeugt. Da es sich um das erste Release von ADF Mobile handelt, sind für die Zukunft funktionale Erweiterungen und auch eine Unterstützung zusätzlicher Plattformen zu erwarten.

Volker Linz
volker.linz
@oracle.com

Dr. Jürgen Menge
juergen.menge
@oracle.com