Migration einer komplexen ADF-Anwendung auf ADF Essentials

Matthias Neubert
Robotron Datenbank-Software GmbH
Dresden

Schlüsselworte
ADF, ADF Essentials, GlassFish, WebLogic, Migration

Einleitung
Mit ADF Essentials stellt Oracle eine lizenzkostenfreie Version des Application Development Frameworks (ADF) bereit. Damit wurde eine Möglichkeit geschaffen, ADF-Anwendungen auf einem beliebigen JavaEE-Server zu deployen. In diesem Beitrag wird gezeigt, wie eine bestehende, komplexe ADF 11g-Anwendung zur terminierten Aufgabensteuerung bei einem großen Energieversorger auf ADF Essentials migriert wurde. Als Zielsystem kam ein GlassFish-Server in der Version 3.1 zum Einsatz.
Es werden alle bei der Migration durchgeführten Schritte ausführlich erläutert und dabei insbesondere auf die Vorbereitung des GlassFish-Servers für ADF Essentials, die Änderungen an der Projektkonfiguration sowie die erforderliche Anpassungen am Quellcode und das Deployment der ADF Essentials-Anwendung eingegangen. Ein besonderer Fokus wird auf die Umsetzung des Rollen- und Rechtesystems gelegt, da ADF Security kein Bestandteil von ADF Essentials ist. Hierfür wurde eine flexible Lösung entwickelt, die es mit geringem Aufwand erlaubt, zwischen ADF Security und einer alternativen Implementierung zu wechseln.

ADF vs. ADF Essentials
Bei der Planung von Projekten mit ADF Essentials ist zu beachten, dass einige Features nicht verwendet werden können. Tabelle 1 stellt einen Vergleich der enthaltenen Features dar. Für Webanwendungen, die eine Benutzerauthentifizierung erfordern, ist insbesondere zu bedenken, dass auf ADF Security verzichtet werden muss.
Dem gegenüber machen die entfallenden Lizenzkosten und die Unabhängigkeit vom WebLogic-Server ADF Essentials zu einer interessanten Alternative.

Feature	ADF	ADF Essentials
ADF Faces Rich Client Components	✓	✓
ADF Controller	✓	✓
ADF Model	✓	✓
ADF Business Components	✓	✓
ADF Mobile	✓	
ADF Desktop Integration	✓	
ADF Security	✓	
ADF Webservice Data Control	✓	

ADF Remote Taskflows	✔	
ADF Business Component's Service Interfaces	✔	
ADF Data Controls for BI, Essbase and BAM	✔	
Integration with Oracle Fusion Middleware features such as MDS , …	✔	

Tabelle 1: Vergleich der Features von ADF und ADF Essentials

Projektrahmen AMT

Das Robotron Aufgabenmanagement-Tool (AMT) ist ein Werkzeug zur terminierten Aufgabensteuerung. Es erlaubt die Erstellung und Verteilung von Aufgaben und Arbeitsaufträgen. Zudem wird die Dokumentation und Archivierung erstellter Aufgaben ermöglicht. Abbildung 1 zeigt die Aufgaben-Übersichtsseite der Anwendung.

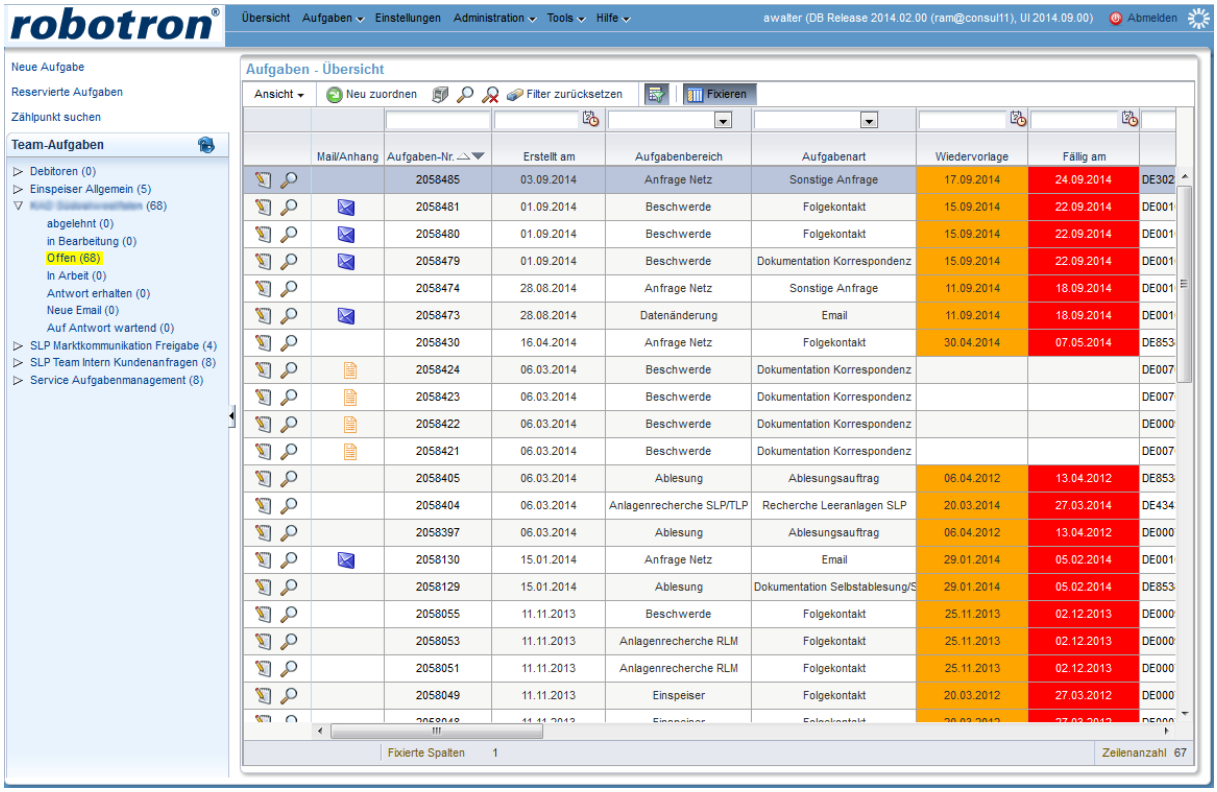

Abb. 1: AMT: Aufgaben-Übersicht

Die Funktionalität von AMT umfasst u.a.:

- Verwaltung und Verteilung von Aufgaben
- Aufgaben können über die Anwendung eingegeben oder über eine E-Mail-Schnittstelle automatisch importiert werden.
- Manuell eingegebene Aufgaben können auch über konfigurierbare Schnelltickets erstellt werden. Dabei werden die Pflichtfelder der Aufgabe in einem Schritt mit vorkonfigurierten Werten belegt.
- Die Bearbeitung der Aufgaben kann über Bearbeiter bzw. Gruppen erfolgen.

- Es können Rückfragen via E-Mail direkt über die Anwendung gestellt werden.
- Es kann nach Aufgaben recherchiert werden.
- Gesamtstatus und Teilstatus der Aufgaben sind konfigurierbar.
- Der Stand des gesamten Aufgabenbestandes ist über eine integrierte DWH/BI-Komponente auswertbar.
- Zuordnung verschiedener Objekttypen aus dem bestehenden Energiedatenmanagement-System zu Aufgaben, wie Zählpunkte, Verbrauchsstellen, Kunden und Lieferanten
- automatische Teamzuordnung zu Aufgaben auf Basis von Objektzuordnungen, Postleitzahlen etc.
- Integration mit verschiedenen Umsystemen wie GPKE/WiM-Wechseltools, dem EDM-System ***robotron*e✓count,*** Fehlermanagementsystemen, SAP BW, Archivsystemen usw.
- Korrespondenzunterstützung inkl. automatische Befüllung von MS Office-Vorlagen

AMT basiert auf Oracle ADF 11g (Oracle 11g, Oracle Business Components, Oracle Data Binding, Oracle ADF Faces) und wird in einem Oracle WebLogic-Cluster betrieben.

Das Cluster besteht aus drei produktiven WebLogic-Servern, die über einen Load Balancer ausgesteuert werden.

Migrationsschritte
Als Erstes muss der GlassFish-Server eingerichtet und für das Deployment einer ADF Essentials-Anwendung konfiguriert werden. Dafür sind die folgenden Schritte notwendig:

1. ADF Essentials-Library mit unzip -j adf-essentials11g.zip in ...\glassfish3\glassfish\domains\<Domain Name>\lib entpacken
2. glassfish.jstl_1.2.0.1.jar und javax.el-2.2.6.jar in den Ordner ...\glassfish3\glassfish\domains\<Domain Name>\lib kopieren
3. Neustart des GlassFish-Servers und Aufruf der Administrationskonsole (Standard: http://localhost:4848/)
4. Aufruf der JVM-Einstellungen unter Configuration / Server-config / JVM Settings / JVM Option Tab
5. -XX:MaxPermSize=<Wert> auf -XX:MaxPermSize=512m ändern
6. Eintrag -Doracle.mds.cache=simple hinzufügen
7. Neustart des GlassFish-Servers

Des Weiteren muss noch die Datasource der Anwendung erstellt werden. Unter der Voraussetzung, dass eine Oracle-Datenbank verwendet wird, erfordert dies folgende Schritte:

1. Oracle JDBC Treiber (ojdbc6.jar) in den Ordner in ...\glassfish3\glassfish\domains\<Domain Name>\lib kopieren und den GlassFish-Server anschließend neu starten
2. in der Administrationskonsole unter Ressourcen / JDBC / JDBC-Connection Pools eine neue Connection anlegen (Ressourcentyp: javax.sql.DataSource)
3. unter weitere Eigenschaften die Angaben zu username, password und url ergänzen
4. unter Ressourcen / JDBC / JDBC-Ressourcen eine neue JDBC Ressource hinzufügen und konfigurieren

Abgeschlossen wird die Einrichtung des GlassFish-Servers mit der Konfiguration des Security-Realms. In der Administrationskonsole ist dazu Konfiguration / server-config / Sicherheit / Realms aufzurufen und ein neuer Realm zu erstellen (siehe Abbildung 2). Der Name des Realms sollte für WebLogic und GlassFish identisch gewählt werden, um zusätzliche Änderungen in der Projektkonfiguration zu vermeiden.

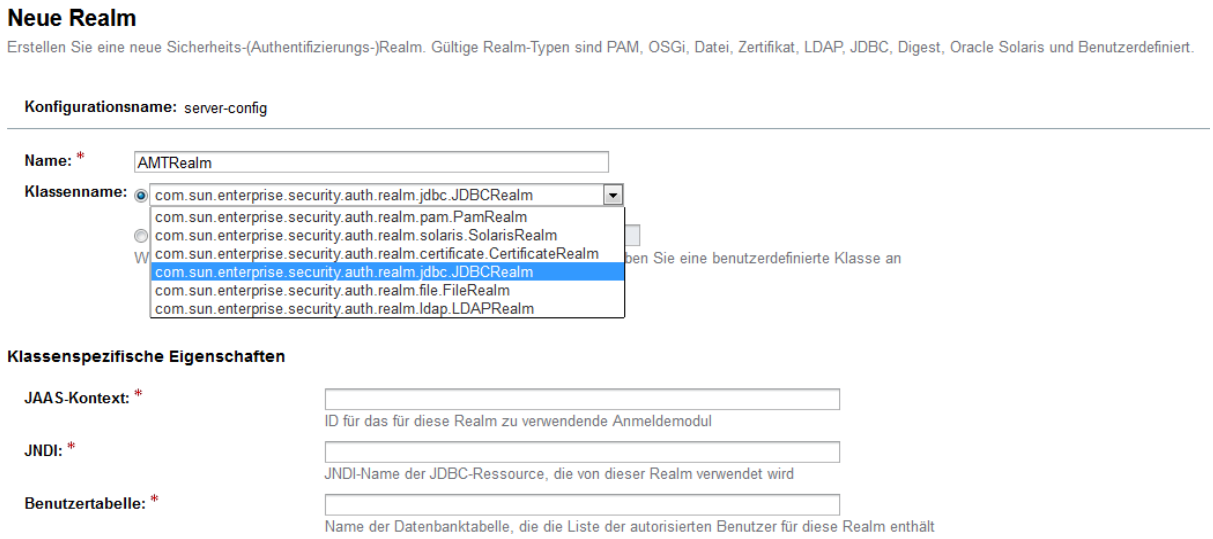

Abb. 2: Einrichtung eines neuen SecurityRealms

Für den Entwicklungsprozess ist es sinnvoll die Steuerung des GlassFish-Servers im JDeveloper einzubinden. Dafür muss die GlassFish Extension über Help / Check for Updates installiert werden. Anschließend sind über das Einstellungsmenü die Pfade zur GlassFish-Installation anzupassen (siehe Abbildung 3).

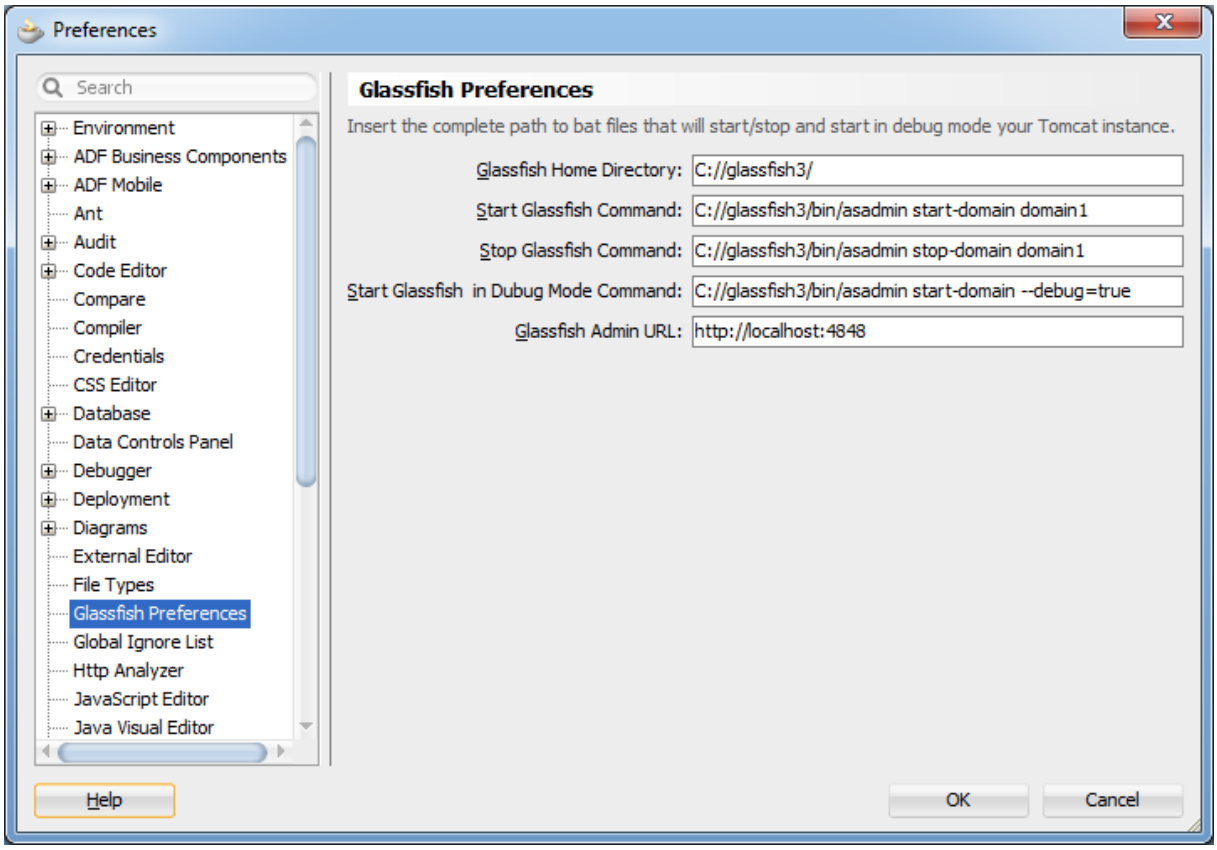

Abb. 3: Konfiguration GlassFish Extension

Es folgt die Anpassung der Implementierung. Die wichtigste Änderung im AMT-Projekt bestand darin, die Funktionalität von ADF Security zu ersetzen. Das bedeutet, es musste eine hierarchisches Rollensystem und eine Nutzerauthentifizierung umgesetzt werden. Es wurde dabei bewusst auf ein weiteres Framework (z.B. Apache Shiro[1]) verzichtet, um zusätzliche Abhängigkeiten zu vermeiden. Bei weitreichenderen Anforderungen wie z.B. Anwendung kryptographischer Verfahren ist die Verwendung aber durchaus empfehlenswert.
Damit für das Deployment auf einem WebLogic-Server weiterhin ADF Security verwendet werden kann und ein einfacher Wechsel zwischen verschiedenen Zielplattformen möglich ist, wurde diese Funktionalität in einem Wrapper gekapselt. Die Wrapper-Library muss in allen Teilprojekten eingebunden werden, die den ADF-SecurityContext oder die WLS API nutzen, z.B. bei Verwendung der Klassen weblogic.security.URLCallbackHandler, weblogic.security.services.Authentication und weblogic.servlet.security.ServletAuthentication.
Anschließend müssen im Quellcode die Aufrufe durch die Wrapper-Methoden ersetzt werden. Dieser einmalige Migrationsschritt eröffnet die Möglichkeit, diese Komponenten jederzeit durch alternative Implementierungen des Wrappers zu ersetzen. In der verwendeten Variante der Wrapper-Library sind die serverspezifischen Aufrufe für WebLogic und GlassFish enthalten. Eine Erweiterung für weitere Applikationsserver ist durch Implementierung der Interfaces problemlos möglich.
In diesem Zusammenhang wurde auch das Logging mit dem ADF-Logger im Wrapper gekapselt, um bei einem Deployment auf einem GlassFish-Server Log4J statt des ADF-Loggers zu nutzen.
Ferner sind diese Schritte durchzuführen:

1. Im ViewProject muss die Library des GlassFish-Servers (glassfish.jstl_1.2.0.1.jar) eingebunden werden.
2. Alle Methodenaufrufe, die eine Instanz von oracle.adf.share.security.SecurityContext nutzen, werden durch die statische Klasse SecurityContextWrapper ersetzt.
3. Ebenso werden alle Vorkommen der Klasse oracle.adf.share.logging.ADFLogger durch die Wrapper-Klasse LoggingWrapper ersetzt.
4. In der adfc-config.xml muss der securityContextWrapper mit der Klasse de.robotron.wrapper.securitycontext.SecurityContextBean im RequestScope angelegt werden. Anschließend ist in allen EL-Ausdrücken securityContext durch securityContextWrapper zu ersetzen.

   Beispiel:
   ```
   <af:commandToolbarButton id="cb1" icon="/icons/add.png"
   actionListener="#{ArbeitsgruppenTeamsCreateEditBean.onCreate}"
   disabled="#{pageFlowScope.isNewRow}"
   rendered="#{not securityContextWrapper.userInRole['app_ro_user']}"/>
   ```

5. Die Groovy-Expression adf.context.securityContext.userName muss in allen ViewObjects und EntityObjects durch den Ausdruck adf.context.sessionScope.userName ersetzen werden.
6. Für die Konfiguration der Anwendung müssen die Deployment Descriptors web.xml, glassfish-web.xml und wrapper-config.xml unter WEB-INF ersetzt bzw. eingefügt und angepasst werden. Die glassfish-web.xml enthält die SecurityRole-Zuordnung, z.B.

   ```
   <security-role-mapping>
     <role-name>app_administrator_rds</role-name>
     <group-name>RAM_RDS_ADMIN</group-name>
   </security-role-mapping>
   ```

[1] http://shiro.apache.org/

Die wrapper-config.xml dient der Festlegung der Zielplattform (siehe Beispiel im Listing). Wahlweise kann über den ServletContext auch eine automatische Erkennung vorgenommen werden.

```
<?xml version="1.0" encoding="windows-1252" ?>
<wrapper>
  <auto-configuration>true</auto-configuration>
  <application-server>GF</application-server>
</wrapper>
```

Nach Abschluss dieser Schritte ist die Anwendung für einen Betrieb auf GlassFish- und WebLogic-Server vorbereitet.

Deployment
Zum Bereitstellen des Deployments sind je nach Zielserver nur wenige, kurze Anpassungen zu erledigen.

GlassFish
Sofern noch kein Deployment-Profile für den GlassFish-Server vorhanden ist, sollte dieses zunächst erstellt werden. Dazu muss im ViewProject unter Properties / Deployment und ApplicationProperties / Deployment ein neues Profil angelegt werden. Als Platform / Default Platform ist GlassFish 3.1 zu wählen. Bevor das EAR generiert werden kann, müssen noch folgende Schritte ausgeführt werden:

1. ADF Security (Application / Secure / Configure ADF Security) muss deaktiviert werden.
2. Im ViewProject ist ADF MDS auszuschalten (Properties / ADF View / Enable User Customizations auf For Duration of Session einstellen)
3. Die Tag-Sektion <adf-mds-config></adf-mds-config> in adf-config.xml wird auskommentiert oder alternativ die adf-config.xml durch die GlassFish-Variante ersetzt
4. Bedingt durch die Deaktivierung von ADF Security müssen die Security-Constraints wie im Listing dargestellt in der web.xml eingefügt werden.

```
[…]
<security-constraint>
  <web-resource-collection>
    <web-resource-name>sqlqueries-task-flow(/WEB-
INF/taskflows/administration)</web-resource-name>
    <url-pattern>/faces/sqlqueries-task-flow/*</url-pattern>
    </web-resource-collection>
    <auth-constraint>
      <role-name>app_administrator_technisch</role-name>
      <role-name>app_ro_admin</role-name>
    </auth-constraint>
  </security-constraint>
[…]
```

WebLogic
Für das Deployment auf einen WebLogic-Server sind diese Schritte notwendig:

1. ADF Security über Application / Secure / Configure ADF Security aktivieren
2. ADF MDS im ViewProject über Project Properties / ADF View / Enable User aktivieren (Customizations Across Sessions using MDS)
3. In adf-config.xml die Tag-Sektion <adf-mds-config></adf-mds-config> einbinden oder alternativ die adf-config.xml durch die WebLogic-Variante ersetzen
4. In der web.xml die Tag-Sektionen <security-role></security-role> und <securityconstraint></security-constraint> auskommentieren

Fazit und Ausblick
Das Projekt hat gezeigt, dass es möglich ist, bestehende ADF-Anwendung auf ADF Essentials zu migrieren. Es entsteht jedoch zusätzlicher Aufwand durch die Umsetzung von Features, die ADF ansonsten bereits mitliefert. Zu bedenken ist dabei, dass die Implementierung wie im vorgestellten Beispiel möglichst flexibel gestaltet sein sollte, um ggf. notwendige Anpassungen bei Versionswechseln leicht vornehmen zu können. Des Weiteren ist entscheidend, welche Features ersetzt werden müssen (z.B. Security) und auf welche Features verzichtet werden kann (z.B. MDS).
Dafür bietet ADF Essentials aber die Möglichkeit, ein stabiles Framework zur Entwicklung von Enterprise-Webanwendungen auf verschiedenen Applikationsservern einzusetzen. Neben dem in diesem Projekt verwendete GlassFish-Server, kann auch JBoss oder Tomcat zum Einsatz kommen. Ein interessanter Aspekt ist die Performance der Anwendung auf unterschiedlichen Servern. Im Rahmen des Projektes wurden keine belastbaren Vergleichstests durchgeführt. Es zeigte sich aber schon bei einfachen Testläufen, dass sich die Anwendung auf dem GlassFish-Server agiler verhält. Weitreichendere Untersuchungen zu diesem Thema sind bspw. im Blog von Andrejus Baranovskis zu finden und untermauern diese Beobachtung[2].

Kontaktadresse:
Matthias Neubert
Robotron Datenbank-Software GmbH
Stuttgarter Str. 29
D-01189 Dresden

Telefon:	+49 (0) 351 25859 2450
Fax:	+49 (0) 351 25859 3699
E-Mail	matthias.neubert@robotron.de
Internet:	www.robotron.de

[2] http://andrejusb.blogspot.de/2012/10/adf-11g-r2-weblogic-1035-vs-adf.html