Orientation in Objects

XDoclet - MDA für Praktiker?

Das Doppelte Cödchen

) Schulung)

) Beratung)

) Entwicklung)

) Artikel)

AUTOR

Kristian Köhler
Orientation in Objects GmbH

Veröffentlicht am: 29.8.2003

ABSTRACT

Bei der Entwicklung mit J2EE werden oft ähnliche oder gar identische Codeblöcke eventuell in abgeleiteter Form an verschiedenen Stellen benötigt. Das Kopieren dieser Blöcke ist zeitintensiv und fehleranfällig, da diese Wiederholungsaufgaben meist mit wenig Sorgfalt ausgeführt werden. Durch den Einsatz geeigneter Codegeneratoren können wiederkehrende Aufgaben automatisiert und Pflege sowie Entwicklungszeit eingespart werden.

Der vorliegende Artikel beschreibt anhand eines Beispiels wie mit dem OpenSource Codegenerator XDoclet die Entwicklung von Anwendungen erleichtert und beschleunigt werden kann.

Orientation in Objects GmbH

Weinheimer Str. 68
D-68309 Mannheim

Tel. +49 (0) 6 21 - 7 18 39 - 0
Fax +49 (0) 6 21 - 7 18 39 - 50

www.oio.de info@oio.de

WAS IST XDOCLET?

XDoclet ist eine OpenSource Engine für Codegenerierung. Sie liest Java Quelldateien ein und wertet die darin enthaltenen Metainformationen aus. Die XDoclet Informationen werden in Form von JavaDoc Tags direkt im Quellcode angegeben. Mit Hilfe von Code-Schablonen und den ermittelten Metainformationen erzeugt die Engine die gewünschten Ausgabedateien. Mögliche Zielformate sind unter anderem XML (z. B. Deployment Deskriptoren), Java Quelldateien oder Implementierungen der J2EE Patterns.

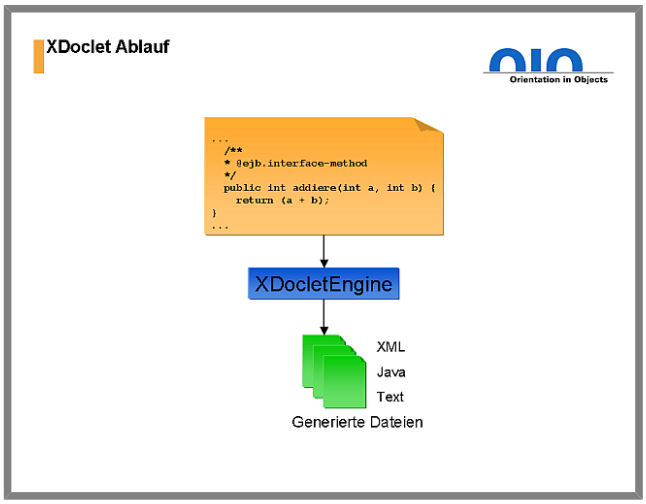

Abbildung 1: XDoclet Ablauf

Der Funktionsumfang von XDoclet kann über Module erweitert werden. Module enthalten die entsprechenden Generierungs-Schablonen für eine bestimmte Technologie. Folgende Module sind bereits in XDoclet integriert:

- EJB
- Web
- Hibernate
- Mockobjects
- JDO
- JMX

Die Anpassung bzw. die Erstellung eigener Module ist möglich.

Für die Ausführung der Engine wird in der aktuellen Version das ebenfalls als OpenSource verfügbare Buildwerkzeug Apache Ant benötigt. Die einzelnen Module stellen Ant Tasks zur Verfügung, mit deren Hilfe der eigentliche Generierungslauf ausgeführt wird.

XDOCLET UND J2EE

XDoclet bietet eine gute Unterstützung der J2EE Technologien. Neben der manuellen Erstellung bzw. Erweiterung der Deployment Deskriptoren entfallen z. B. auch zeitaufwendige Tätigkeiten wie die Erstellung von Home- und Remoteinterfaces bei der EJB Entwicklung. Es genügt, die Bean Klasse um Metainformationen in Form von Tags zu erweitern.

Dies hat zum einen zur Folge, daß innerhalb des Projektes keine veralteten Deployment Deskriptoren mehr enthalten sind, zum anderen erhält der Entwickler eine bessere Übersicht über das Projekt, da die Informationen nicht mehr auf mehrere Dateien verteilt, sondern zentral in einer Datei gepflegt werden. Mit Hilfe von XDoclet lassen sich neben Standard Deployment Deskriptoren auch containerabhängige Deskriptoren erzeugen. Unterstützt werden zur Zeit unter anderem:

- JBoss
- Tomcat
- Bea Weblogic
- IBM Websphere

Zusätzlich wird die Implementierung einiger J2EE Patterns erleichtert. Durch Deklaration in der Java Datei sowie in der Build Datei lassen sich so z. B. ValueObjects, Struts Formulare oder Schlüsselgenerierungs Session Beans erzeugen.

BEISPIEL EJB MIT XDOCLET

Abschließend wird anhand eines Beispiels der Einsatz von XDoclet während der EJB Entwicklung gezeigt. Implementiert wird eine stateless Addierer Komponente, mit deren Hilfe zwei Zahlen addiert werden können.

Ausgangspunkt der Entwicklung ist die Bean Klasse, die wir, entgegen der eigentlichen Vorgehensweise bei Session Beans, als abstrakte Klasse implementieren. XDoclet erstellt bei der Generierung eine abgeleitete Klasse, die die fehlenden Callback Methoden (setSessionContext, etc.) implentiert. Diese Klasse wird dann auch automatisch in den Deployment Deskriptor eingetragen.

In die Dokumentation der Klassen werden allgemeine Metainformationen wie Bean-Name und Typ des Beans aufgenommen. In der Dokumentation der Business-Methode wird definiert, daß die Methode auch im Interface aufgeführt sein soll. Damit ist die Bean Implementierung abgeschlossen.

```
package de.oio.server;
import javax.ejb.SessionBean;
/**
* @ejb.bean name = "Addierer"
* type = "Stateless"
* jndi-name = "AddiererJNDI"
*
**/
public abstract class AddiererBean
                  implements SessionBean {
   /**
   * @ejb.interface-method
   */
   public int addiere(int a, int b) {
      return (a + b);
   }
}
```

Beispiel 1: Bean Implementierung

Zur Generierung wird das XDoclet Modul ejbdoclet benutzt. Dieses muß hierzu in der Build Datei angegeben und aufgerufen werden. Im Beispiel enthält der Aufruf von ejbdoclet die Anweisung, daß sämtliche auf Bean.java endende Dateien im Source Verzeichnis in die Verarbeitung eingeschlossen werden sollen. Durch das session Element wird ejbdoclet angewiesen, Session Beans zu berücksichtigen und die entsprechenden Tags auszuwerten. Das Attribut acceptAbstractClasses erlaubt den oben bereits erwähnten Einsatz von abstrakten Klassen bei Session Beans. Im weiteren wird definiert, daß Remote- und Homeinterfaces erstellt werden sollen. Zusätzlich wird ein Standard Deployment Deskriptor sowie ein JBoss Deployment Deskriptor erzeugt.

Die so erstellten Java Klassen können nun compiliert und in ein JAR Archiv gepackt werden. Dieses kann anschließend in den Server deployed werden.

```
<target name="xdoclet-generate" depends="init">
   <taskdef name="ejbdoclet"
          classname="xdoclet.modules.ejb.EjbDocletTask">
      <classpath refid="compile.classpath" />
   </taskdef>
   <ejbdoclet destdir="${build.src.dir}"
           excludedtags="@version,@author"
           mergedir="${meta.dir}/ejb-jar/META-INF/">
      <fileset dir="${src.dir}">
         <include name="**/*Bean.java" />
      </fileset>
      <session acceptAbstractClasses="true" />
      <remoteinterface />
      <homeinterface />
      <deploymentdescriptor
          destdir="${build.dir}/META-INF" />
      <jboss version="3.0"
           xmlencoding="UTF-8"
           destdir="${build.dir}/META-INF"
           validateXml="false" />
   </ejbdoclet>
</target>
```

Beispiel 2: ANT-Target für XDoclet-Generierung

FAZIT

XDoclet ist sicher kein Model Driven Architecure (MDA) Tool im eigentlichen Sinne, es handelt sich vielmehr um ein hilfreiches Werkzeug, mit dessen Hilfe die Entwicklung besonders von J2EE Anwendungen vereinfacht wird. Der Entwickler kann sich wieder auf die eigentlichen Aufgaben konzentrieren und muß sich nicht mehr mit aufwendiger, fehleranfälliger Cut- and Paste Arbeit aufhalten.

REFERENZEN

- XDoclet
  (http://www.xdoclet.org)
- Apache Ant
  (http://ant.apache.org)
- OMG - Model Driven Architecture
  (http://www.omg.org/mda/)