

ERItemManager — ein dynamisch rekonfigurierbares Webapplikationsframework

Sascha Preibisch, Tao Zheng und Wolfgang Renz *
Labor für Multimediale Systeme (MMLab),
Fakultät Technik und Informatik,
Hochschule für Angewandte Wissenschaften Hamburg,
Berliner Tor 7, 20099 Hamburg, Germany

Abstract:

XML-konfigurierbare Webapplikationsframeworks sind Stand der Technik. Deskriptive XML-Programmierung mit eingebetteten Aktivitäten auf Basis von Middlewareplattformen wurde in verschiedenen Kontexten als Alternative zur schwerfälligeren J2EE-Webapplikationsentwicklung vorgeschlagen und realisiert. Mit dem ERItemManager wird ein vom Ercaton-Paradigma[Ercato] inspirierter dynamisch rekonfigurierbarer ItemManager konzipiert und realisiert. Um den Entwickler dabei zu unterstützen, sich auf das fachliche Problem konzentrieren zu können, bietet das Framework die Möglichkeit, mit Hilfe einer validen XML-Datei sämtliche Teile der Anwendung, die nicht von der fachlichen Logik abhängig sind, generieren zu lassen. Darunter fällt der gesamte Bereich der Persistenz, der Schnittstelle zwischen Client und Server und die Erzeugung von Benutzerschnittstellen. Alle diese Elemente sind sofort, ohne jegliche Änderung des Frameworks, nutzbar. Einzig die fachliche Funktionalität oder eine spezialisierte Benutzerschnittstelle müssen vom Entwickler neu entwickelt werden. Dies geschieht gegen spezifizierte Schnittstellen. Exemplarisch wurde damit ein Prototyp einer Webapplikation zu Projektmanagement, Bugtracking und Zeiterfassung realisiert, um das geschaffene Framework bewerten zu können.

1 Zielsetzung

Für die Nutzung im Firmenumfeld sollte eine Software entwickelt werden, mit der Arbeitszeiten erfasst und ausgewertet, Benutzer verwaltet, Berichte erzeugt werden können und die Bugreporting ermöglicht. Im Laufe der Anforderungsanalyse hat sich der Entwicklungsauftrag dahingehend verändert, dass man doch lieber etwas entwickelt, mit dem man auf immer neue Anforderungen eingehen kann. Als im Sept. 2004 die Entwicklung des ERItemManager gestartet wurde, war keine passende Software bekannt, die die gesetzten Anforderungen direkt erfüllte. Ein Anspruch lag in niedrigen Beschaffungskosten, da es sich nicht um Software handelt, die direkt zur Umsatzsteigerung beiträgt. Zudem sollte die benötigte Infrastruktur möglichst einfach sein. Bei jeder Eigenentwicklung musste

* sascha.preibisch@ipt.ch, zhengtaofw@hotmail.com, wr@informatik.haw-hamburg.de

man sich mit den Themen Persistenz, Zugriffsschicht, Benutzerschnittstelle und Ablaufkontrolle beschäftigen. Ein sehr grosser Aufwand, der häufig dazu geführt hatte, auf die Entwicklung eines solchen Werkzeugs zu verzichten.

2 Das Framework

Das ERItemManager Framework sollte es nun ermöglichen, ein Projekt einzig in Form einer XML Datei zu beschreiben, die ggf. Referenzen auf in Java implementierte Aktivitäten enthalten kann. Diese Projektdatei sollte die Struktur von Geschäftsobjekten (Businesslogik) sowie die Konfiguration der Persistenzschicht und einer passenden Zugriffsschicht enthalten. Ausserdem sollte auf dieser Basis ein Standard Fat Client generiert werden. Nach der Fertigstellung der Projektdatei sollte man also folgende Bestandteile erhalten:

1. die Persistenzschicht in Form von Tabellen einer relationalen Datenbank inkl. sämtlicher Schlüssel und Indizes,
2. eine in Java implementierte Zugriffsschicht; für die Kommunikation mit der Persistenzschicht werden webservices genutzt,
3. Benutzerschnittstelle implementiert mit dem Eclipse RCP Framework.

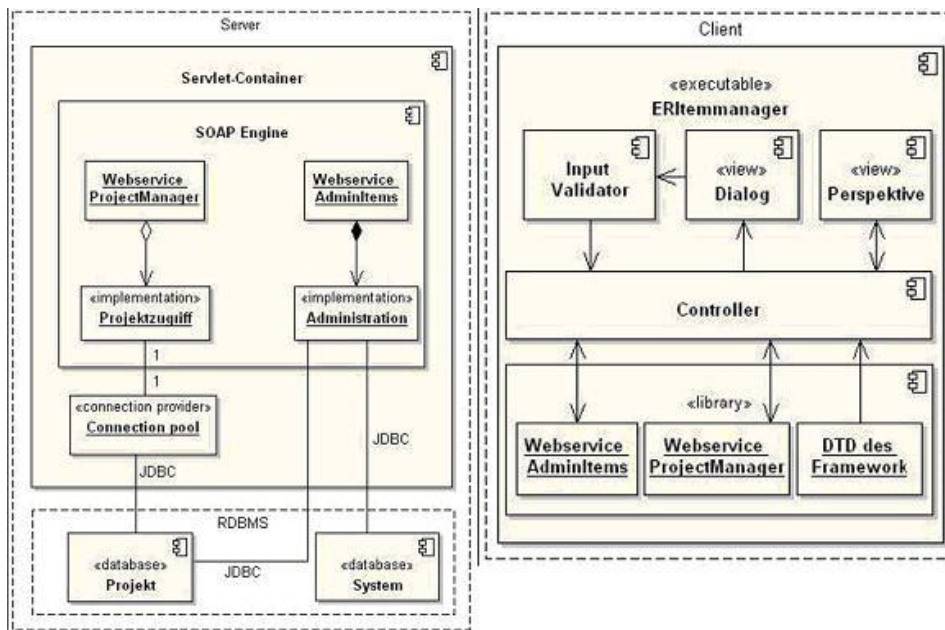
Bestandteile der zur Verfügung gestellte Infrastruktur müssen sein: Java 5, Apache Tomcat 5.5, Apache Axis 1.2, PostgreSQL 8.0 (oder ein anderes RDBMS) Der Workflow für die Erzeugung eines Projektes sieht wie folgt aus:

1. Anlegen zweier Datenbanken, eine für Projekttabellen, eine für Systemtabellen, die Projekt unabhängige Daten enthält,
2. Erstellen der Projektdatei mit Hilfe des enthaltenen Projekt Editors,
3. das Projekt mit Hilfe des vorhandenen Administrationsdialogs instanziiieren

In einer weiteren Abschlussarbeit wurde aus technischen Gründen die DTD durch Schema ersetzt. Wesentliche Erweiterung war, dass typische Fälle späterer Änderungswünsche für existierende Projekte erfasst wurden, und die Projektänderung ohne Datenverlust der zu erhaltenden persistenten Daten auf dieser Basis implementiert wurde. Erst dadurch wurde das Framework für den praktischen Einsatz akzeptabel. Außerdem wurde Internationalisierung mit Unterstützung der Sprachen deutsch, englisch und chinesisch realisiert.

3 Abgrenzung

ERItemManager verzichtet ganz bewusst darauf, vorhandene Frameworks zu nutzen oder zu erweitern. Die Komplexität und Einarbeitungszeit sollten gering gehalten werden. Solche Werkzeuge und Frameworks, die Bestandteil des ERItemmanager Frameworks sein könnten, auf die aber verzichtet wurde, sind:



- *Hibernate (www.hibernate.org):*
 Hibernate sorgt als objektrelationales Mappingtool für eine Abbildung von Objekten einer objektorientierten Programmiersprache auf Tabellen einer relationalen Datenbank. Hibernate ist in der Lage, sehr komplexe Abbildungen und Änderungen an Datenstrukturen durchzuführen und Transaktionsmanagement zu realisieren. Auf Hibernate wurde verzichtet. Sonst hätte sich jeder Entwickler von ERItemManager mit hibernate beschäftigen müssen, was zu einer grösseren Komplexität geführt hätte.
- *Spring (www.interface21.com):*
 Spring ist etwa zu der Zeit erschienen, als die Arbeit am ERItemManager-Framework begonnen wurde. Weil aber Spring in etwa die gleiche Intention hatte wie das ERItemManager-Framework, wird es hier aufgeführt. Die Entwickler von Spring hatten erkannt, dass es einfachere Lösungen für immer wiederkehrende Probleme geben muss, als z.B. Suns EJB Technologie. Das Spring Framework besteht heute aus verschiedenen Modulen, die allesamt unabhängig von einander genutzt werden können. So gibt es Spring MVC, Spring WEB, Spring AOP und andere. Bei jedem einzelnen Projekt steht immer das Ziel im Vordergrund, die Entwickler möglichst von wiederkehrenden und nicht fachlichen Aufgaben zu entbinden. Die Spring Entwickler haben jedoch nicht den Anspruch, vorhandene Dinge neu zu erfinden. So kann Spring u.a. hibernate für die Persistierung einsetzen. Der Entwickler muss für die Konfiguration allerdings kein hibernate Experte sein. Spring abstrahiert stark, sodass der Entwickler nur wenig über hibernate wissen muss. Eine Spring-Konfiguration enthält vier

Bestandteile:

- addressBookService: legt fest, welches Objekt die Businesslogik implementiert,
- addressBookDAO: stellt das Data Access Object dar; dies wird für die Persistierung des addressBookService Objektes genutzt,
- jdbcTemplate: weist Spring an, eine Standardkonfiguration für den Einsatz von JDBC zu nutzen; so wird z.B. die komplette JDBC Fehlerbehandlung von Spring behandelt; an die Applikation werden von der Datenquelle unabhängige einheitliche Fehlermeldungen übergeben,
- defaultDataSource: hier werden spezifische Werte für die eigentliche DB Verbindung festgelegt.

Das Spring Framework ist auch deshalb sehr reizvoll, weil allein durch die Konfiguration z.B. von einer jdbc-Datenquelle auf eine andere gewechselt werden kann.

- JSP/ Servlet (java.sun.com) : Bei JSP (JavaServer Pages) und Servlets handelt es sich um Technologien, um dynamisch webbasierte Benutzeroberflächen generieren zu können. Das ERItemManager Framework wurde zwar mit einem Fat Client ausgestattet, aber die offene Struktur ermöglicht es Entwicklern auch, Benutzeroberflächen einzusetzen, die in einem Browser dargestellt werden können. JSP/ Servlets wurden nicht für die Standardimplementierung einer Benutzeroberfläche eingesetzt, weil man nicht auf die Qualitäten eines Fat Clients verzichten wollte.

4 Zusammenfassung

Durch das ERItemManager Framework können Projektteams in sehr kurzer Zeit mit sehr geringem Kostenaufwand eine Software erstellen, die ihren Bedürfnissen angepasst ist. Mit den aus der Projektdatei generierten Elementen ist man in der Lage, Daten verschiedenster Typen zu persistieren, über Netzwerkgrenzen hinweg an einem Projekt zu arbeiten. Zukünftig wird es möglich sein, benutzerdefinierte Funktionen auf einfache Art in den Arbeitsprozess einzubinden. Damit ist die Personalisierung der generierten Elemente möglich. Für den sofortigen Einsatz ist nach der Erstellung der Infrastruktur alles vorbereitet. Eine erste Version des Frameworks wurde während der Diplomarbeit des ersten Autors für die Firma Comosoft entwickelt. In der Abschlussarbeit des zweiten Autors wurde das Framework wie erläutert erweitert. Die Arbeiten haben einschließlich Einarbeitungszeit ca. 6 Monate erfordert. Neben Standardwerken des Software-Engineerings sowie Büchern und Webseiten zu den eingesetzten Techniken wurde folgende Referenz benutzt:

Literatur

- [Ercato] O. Imbusch, F. Langhammer und G. von Walter. Ercatons: Thing Oriented Programming. *LNCS 3263*, 216–237, 2004