Modellbasierte Softwareerstellung (Model-driven Architecture, MDA)

Agenda

- Problem Darstellung
- Entwurfsmuster-basierte Entwicklung
- Case Study: WebML
- Ant

Der Traum von Automasierte Softwareentwicklung

- *Eine deutsch-spanische Firma hat eine Maschine erfunden, die vollautomatisch Software schreibt und zwar weit schneller als menschliche Programmierer (http://www.programmiermaschine.de/)*

  SPIEGEL ONLINE,
  http://www.spiegel.de/spiegel/0,1518,303056,00.html

- Das Ende der Fleißarbeit?

Hype und Realität

- Kommerzielle Wekzeuge verstehen MDA als Muster-basierte Entwicklung

- *Ein Entwurfsmuster (englisch: design pattern) beschreibt eine in der Praxis erfolgreiche Lösung für ein mehr oder weniger häufig auftretendes Entwurfsproblem und stellt damit eine wiederverwendbare Vorlage zur Problemlösung dar. Entstanden ist der Ausdruck in der Architektur, von wo er für die Softwareentwicklung übernommen wurde [http://de.wikipedia.org/wiki/Entwurfsmuster]*
  - Standard References for Design Patterns: http://hillside.net/patterns/
  - Ein Implementationstechnologie-fokusierte Patterns Katalog: http://www-106.ibm.com/developerworks/patterns/

Entwurfsmuster können auf verschieden Abstraktionsebenen auftreten:

- Programming language idioms, z.B. Anonyme Klassen:

```
public class AnOuterClass extends Applet
    {
      int i=0;
      Button bt = new Button("OK");
      public AnOuterClass()
      {
        bt.addActionListener
       ( // Method argument
          new ActionListener() // no name given to this object
          {
            public void actionPerformed(ActionEvent e)
            {
              i++;
              System.out.println("Pressed "+i+" times");
            }
          }
        );
        add(t);
      }
    }
```

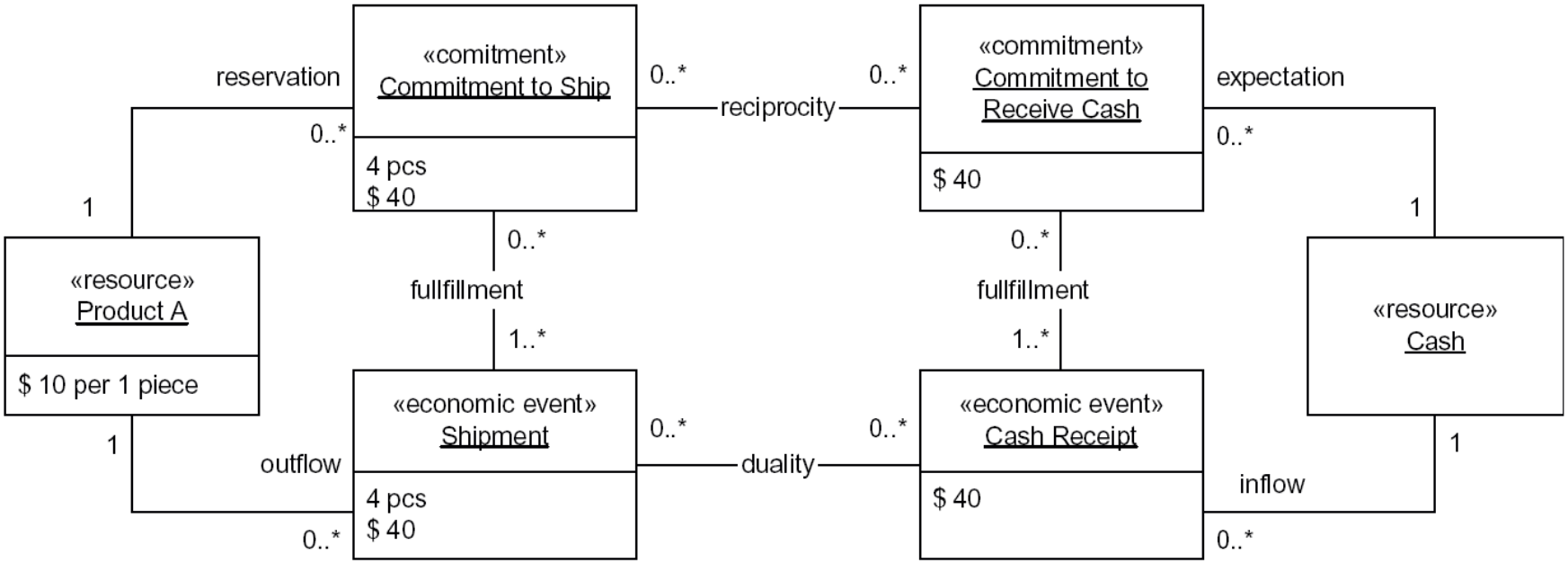
Analysis Ebene, z.B. Verpflichtung (Engl. Commitment) (1 / 4)

- Die meisten Unternehmensereignisse treten nicht unerwartet auf. Eine Vereinbarung zwischen Händlern ist vorher passiert. Die Details solcher Vereinbarungen sollen spezifiert werden.

- Wenn ein Business Partner A sich zur Überweisung von Ressourcen an einen Empfänger B verpflichtet (z.B. Waren bezahlen), wollte B informiert werden, ob die Ressourcen tatsächlich an dem in der Verpflichtung spezifizierten Zeitpunkt verfügbar sein werden

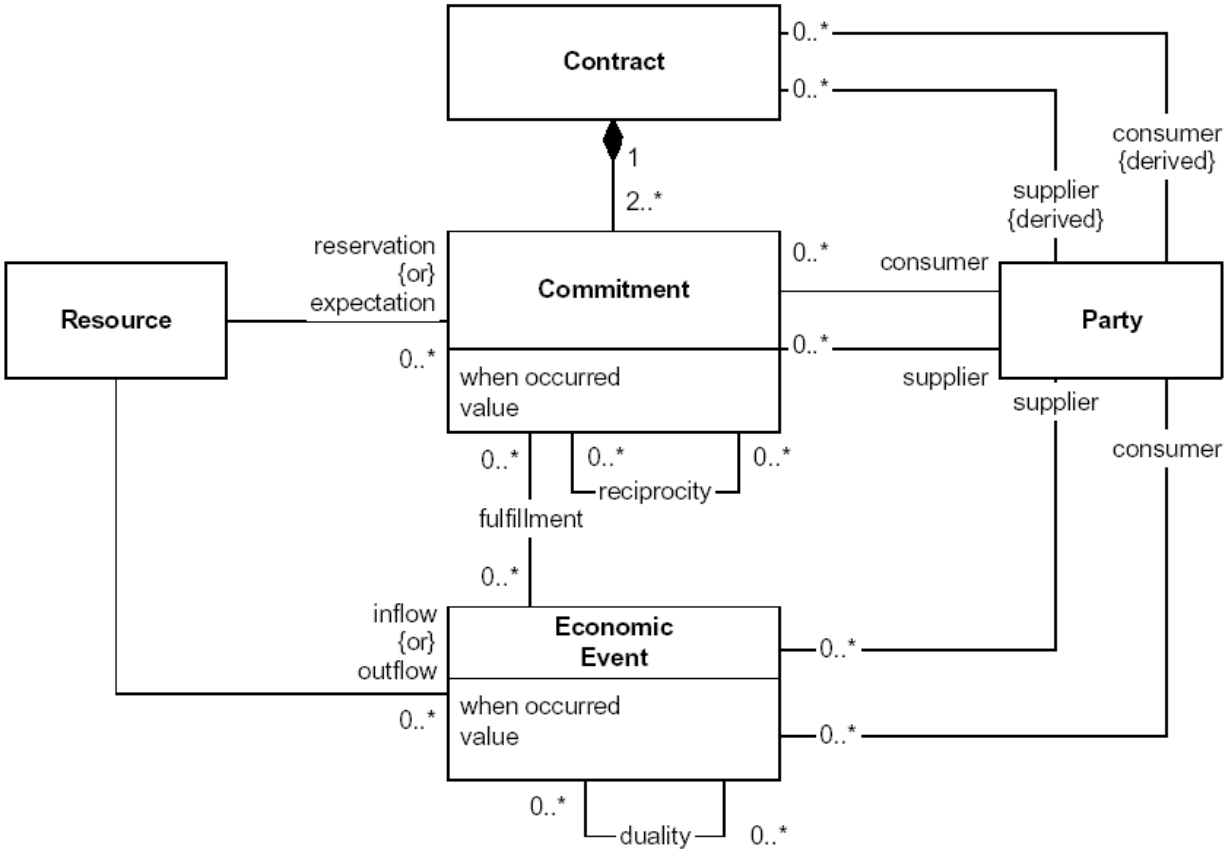
[source: Patterns of Business Software Applications in Model-Driven Architecture, Pavel Hruby, Microsoft Business Solutions]

Beispiel (2 / 4)



[source: Patterns of Business Software Applications in Model-Driven Architecture, Pavel Hruby, Microsoft Business Solutions]

Abstrakte Struktur (Verpflichtung Analysemuster) (3 / 4)



[source: Patterns of Business Software Applications in Model-Driven Architecture, Pavel Hruby, Microsoft Business Solutions]

Axiomen der Analysemuster Verpflichtung (4 / 4)

- Jede Verpflichtung muss zu einer Ressource azossiert sein. Z.B., a sales order line must specify the goods to be sold.

- Each commitment must have two relationships to parties that agree on the future exchange of resources. The customer and vendor, the employer and employee are the examples of parties related in contractual relationship.

- Each commitment must have the fulfillment relation to at least one economic event. For example, the purchase order line specifying the goods must be related to the shipment of these goods.

Das Model View Control Entwurfsmuster (MVC) trennt eine Anwendung in drei Schichten auf

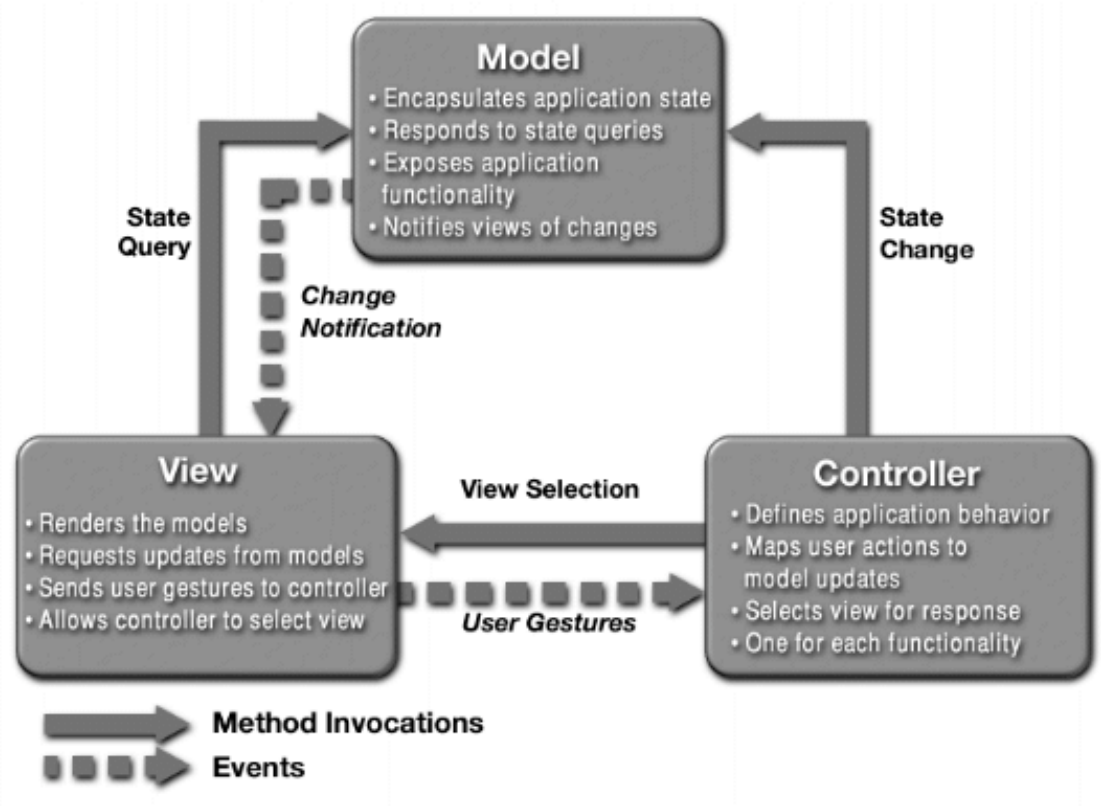
- Model
  Das Datenmodell enthält die dauerhaften (persistenten) Daten der Anwendung. Das Model hat also Zugriff auf diverse Backend-Speicher wie zum Beispiel Datenbanken.

- View
  Die Darstellungsschicht präsentiert die Daten - in der Regel jedoch nicht notwendigerweise zwecks Anzeige. Die Programmlogik sollte aus der View entfernt werden.

- Control
  Die Steuerungsschicht realisiert die eigentliche Geschäftsintelligenz und bildet mit Hilfe der anderen Schichten die Prozesse ab. Sie steuert den Ablauf, verarbeitet Daten, entscheidet, welche View aufgerufen wird, etc.

- Das MVC Entwurfsmuster wurde zunächst für Benutzeroberflächen in Smalltalk beschrieben, gilt mittlerweile aber als defacto Standard für den Grobentwurf aller komplexen Softwaresysteme (mit diversen Verfeinerungen und oftmals mehreren jeweils nach MVC pattern aufgeteilten Modulen). [source: http://de.wikipedia.org/wiki/MVC]

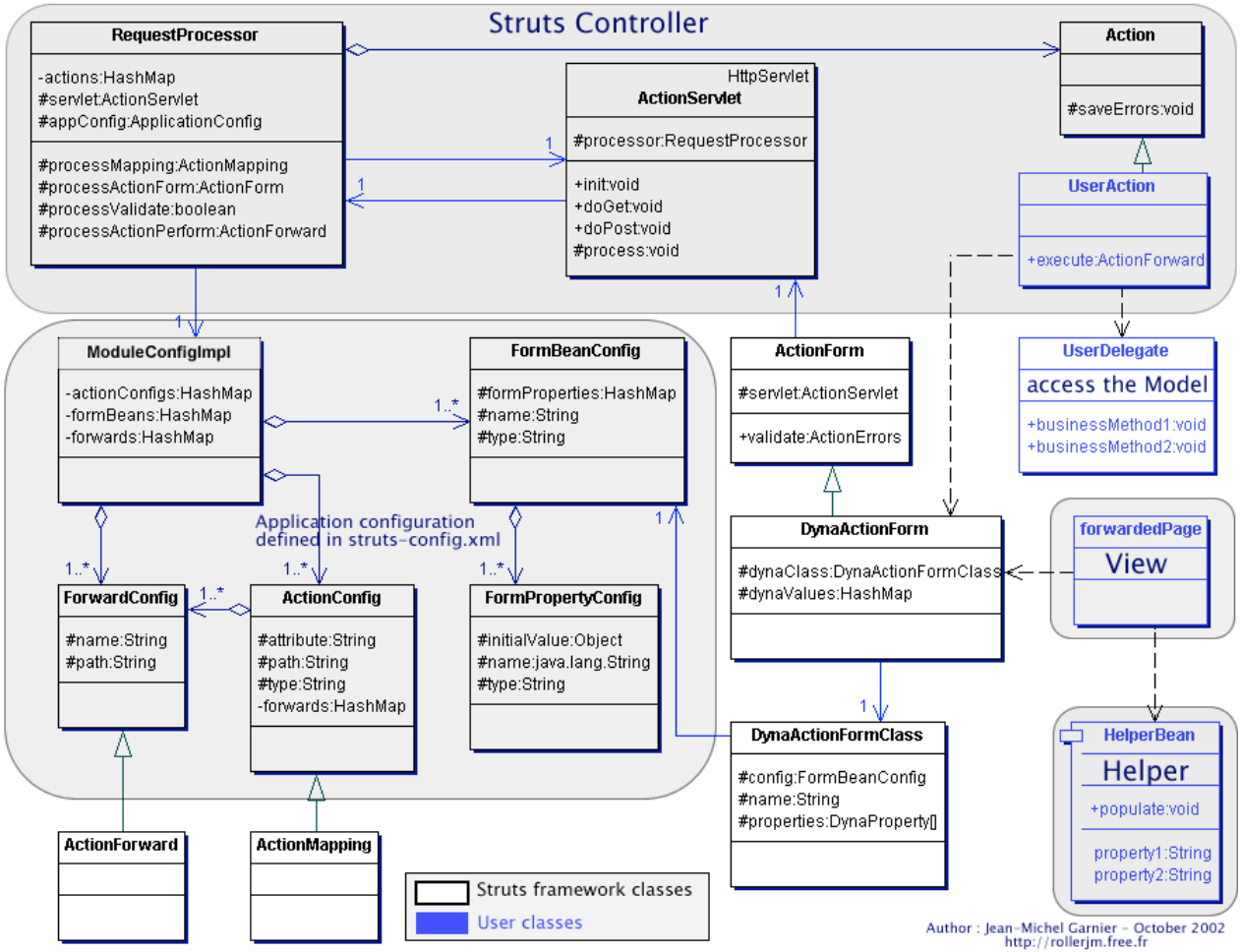
Graphische Darstellung



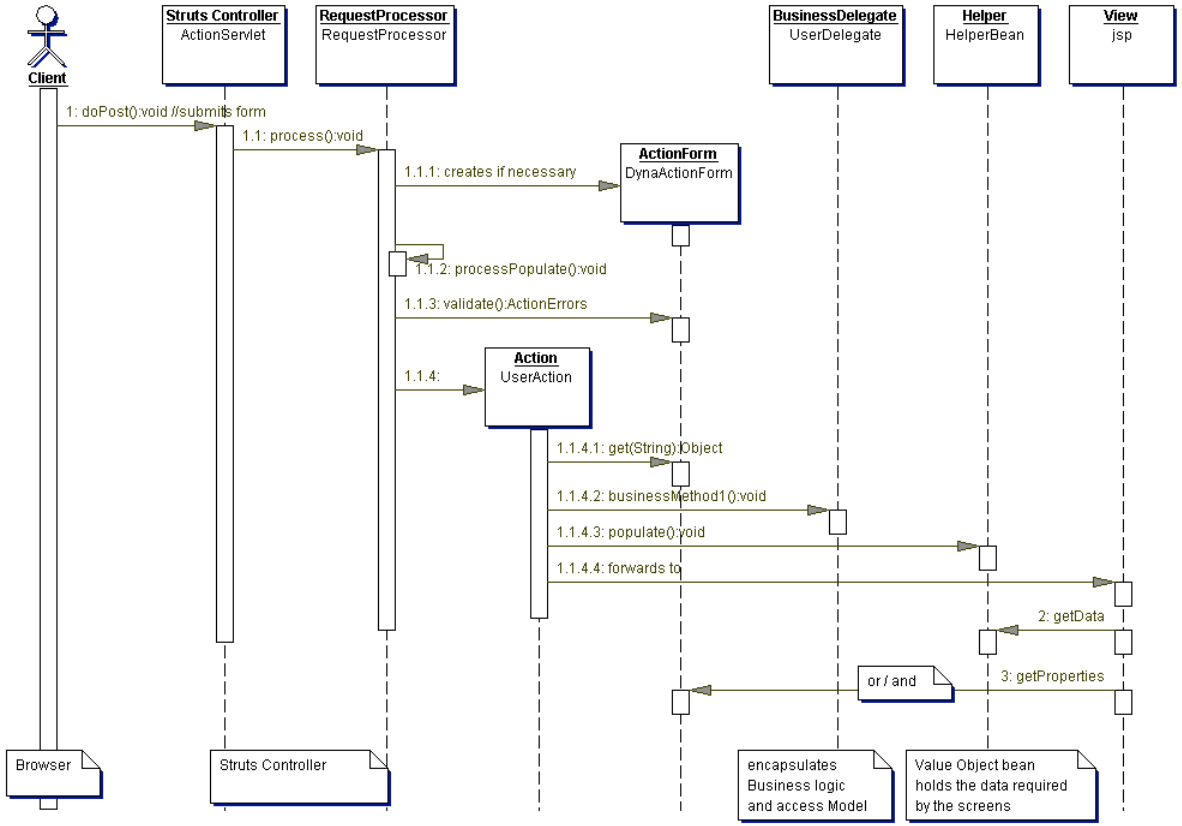
- Struktur des Model-View-Controller Entwurfsmuster
  http://java.sun.com/blueprints/patterns/MVC-detailed.html

Auf Code Ebene die Graphische Darstellung sieht ein bißchen anders aus …

- Struts: Ein Framework für MVC-basierte Webanwendungen
- Softwareentwicklung muss mit Details kämpfen (Details, details, details …)

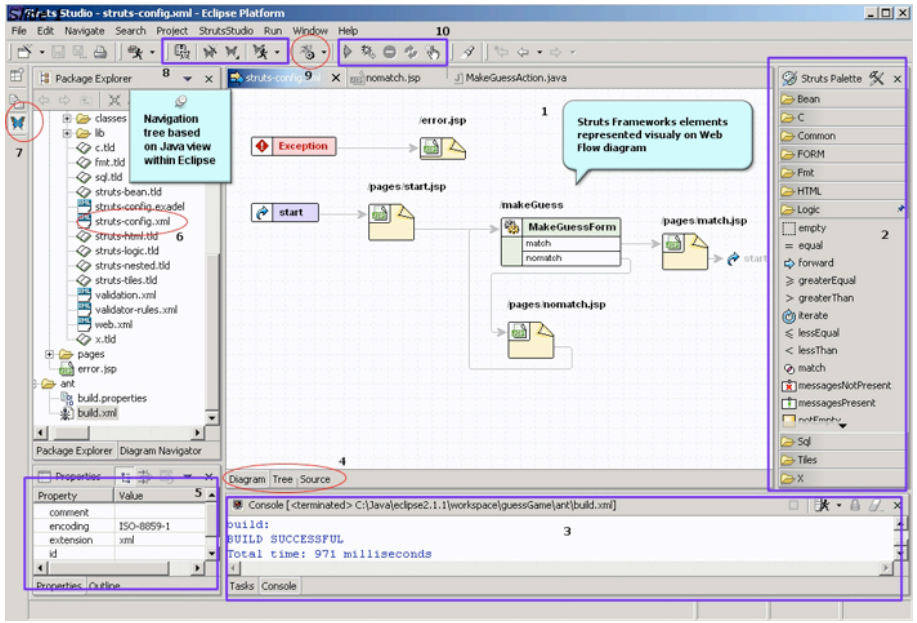


Dynamische Aspekte



[source: http://rollerjm.free.fr/pro/Struts11.html]

Modeling als Allheilmittel



1	**Web Flow**	A graphical representation of the *struts-config.xml* file. Transitions are defined by lines going from one element to anot her.
2	**GEF -Style Struts Pallete**	This holds drag -and -drop code snippets represented by icons for use with the JSP editor. You must have GEF installed to see such pallete. If no GEF installed, you will see the standard style pallete.
3	**Output Console**	Messa ges by Tomcat and Ant are written to this window.
4	**Web Flow tabs**	These tabs allows you to switch between different views of Struts configuration files.
5	**Properties window**	Properties of the currently selected object in the navigation tree.
6	**Tree View**	Tree view of your project (the `struts-config.xml` file is circled).
7	**Struts Studio Perspective Icon**	Clicking on this icon will put you "in" Struts Studio.
8	**Project Toolbar**	Controls that act as shortcuts for project -level actions like saving the project or launching the project as an application in a web browser.
9	**Deployment selector**	Allows to select if to deply to Tomcat or JBoss environments.
10	**Tomcat Toolbar**	Controls for the built -in Tomcat engine.

- Ein repräsentatives Werkzeug für Automatisierung von Struts Entwicklung:
- Struts Studio, http://www.exadel.com/tutorial/struts/5.2/guess/strutstutorial.html
- (es gibt viele andere)

Ungelöste Probleme

- Struts Studio modeliert “nur” die Struts-Teile einer Anwendung. Die Graphische Notation hat einen 1-zu-1 Zusammenhang zwischen Implementierungs und Graphischen Elementen.

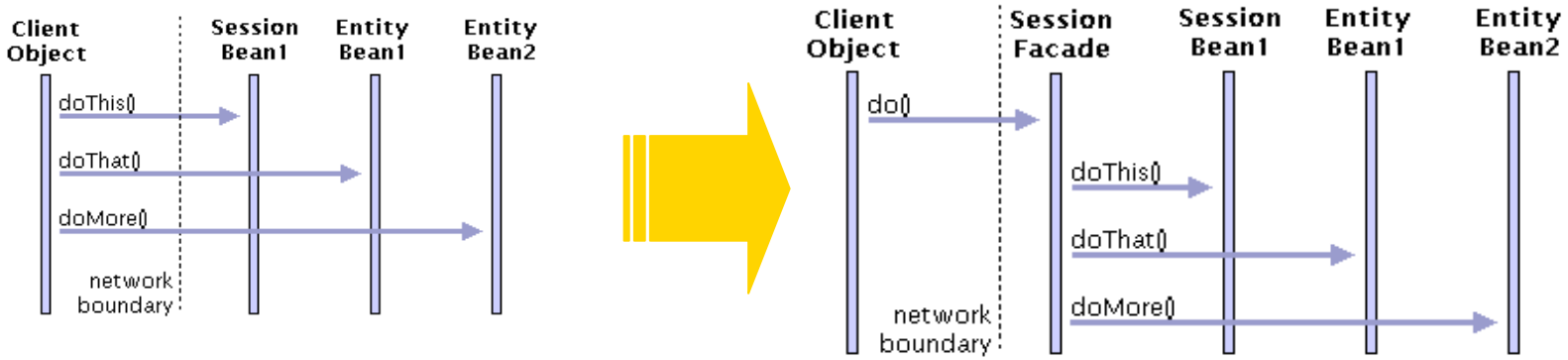
- Es gibt Prototypen, die ganze Anwendungen aus Modellen erstellen (z.B. WebML)

- Fazit: Wir brauchen Werkzeuge, um musternbasierte Abstraktionen von Modellen in Quellcode zu übersetzen (plus Konsistenz Überprüfung, Team Entwicklung, Integration mit Projektmanagement, Debugging auf Modellierungsebene, Versionierung, Refactoring, …)

Kleines Beispiel: Refactoring, das Entwurfsmuster betrachtet (1 / 2)

- Design Pattern Developer
  (Carmen Zannier, University of Calgary, Canada)
  http://sern.ucalgary.ca/~milos/papers/2003/ZannierMSC.pdf

- This plugin offer wizards to help in the complex refactorings needed to evolve towards a best-practices architecture. An example (Session Facade):



http://java.sun.com/blueprints/patterns/j2ee_patterns/session_facade/index.html

Kleines Beispiel: Refactoring, das Entwurfsmustern betrachtet (2 / 2)

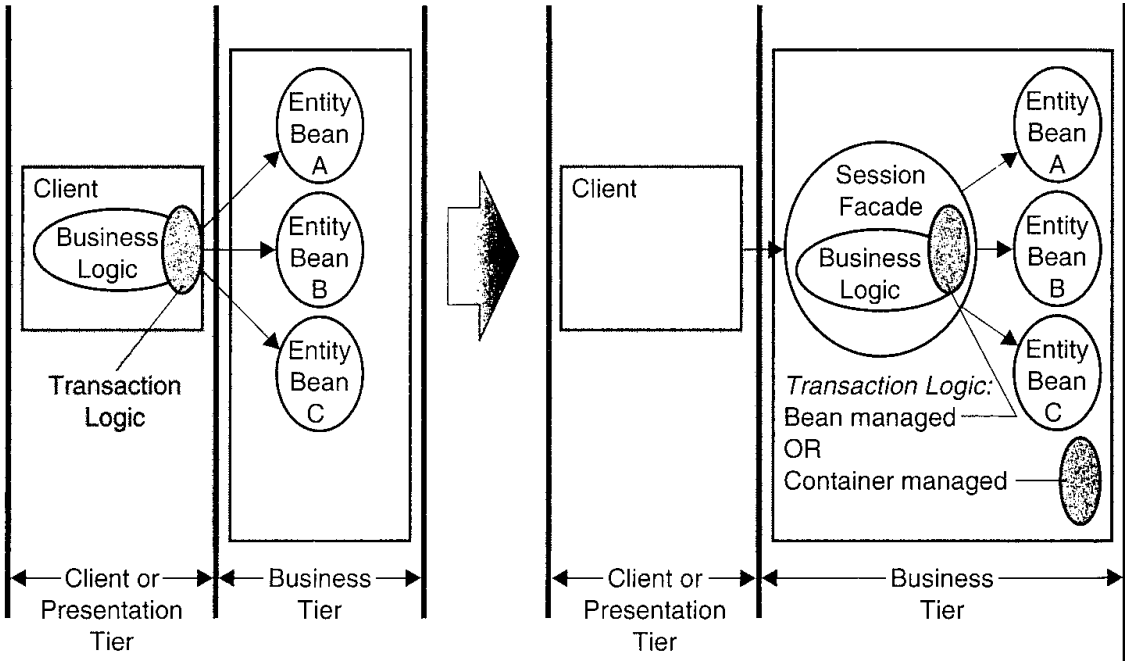


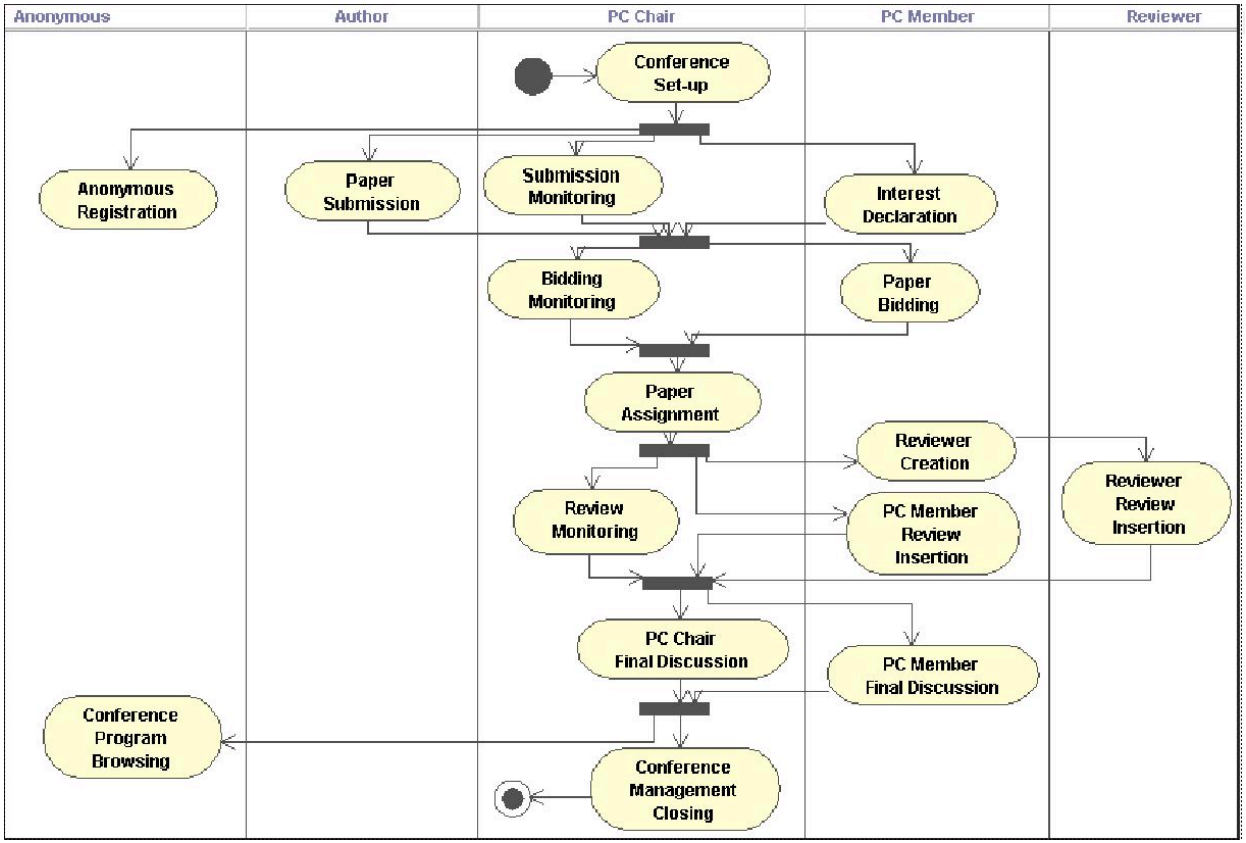
Figure 5.16 Wrap Entities With Session

- *References*
- *Alur, Malks, Crupi: Core J2EE Patterns: Best Practices and Design Strategies, Second Edition, Prentice Hall PTR, ISBN: 0131422464, 2003*
- *Java Blueprint Design Patterns, http://java.sun.com/blueprints/patterns/j2ee_patterns/index.html*

Case Study: WebML

- WebML ist eine auf Webanwendungen angepasste Modellierungssprache. Die Modelle können automatisch in laufende Software übersetzt werden (Java, .NET). Z.B.:
  - electronic catalogs, auctions, virtual marketplaces
  - online newspapers, digital libraries
  - order tracking systems, reservation systems, tourist information systems
  - portals, message boards, technical communities
- Also Anwendungen ohne komplexe algorithmische Verfahren

Sample system in WebML: Conference Management System (CMS)



http://citeseer.nj.nec.com/ceri01designing.html

Orthogonale Modelle in WebML

- Das Struktur Modell beschreibt die konzeptuellen Einheiten, mit deren sich die Anwendung beschäftigt; vergleichbar zu UML Klassendiagramme. Benutzer und Gruppen Modellierung ist eingebaut, um Personalisierung der Webseiten zu vereinfachen.

- Das Derivation Modell erweitert das Struktur Model mit redundanten Daten

- Das Hypertext Model enthalt die Seiten und Links von den Runtime Hypertext(s), und wird wie folgt untergliedert:
  - Kompositionsmodell, mit der internen Organisation für jede Seite (verknüpfte “content units”).
  - Navigationsmodell, beschreibt die Links zwischen Seiten und Content Units (Inter-page Navigation)

- Das Präsentationsmodell formuliert ein default Rendering in einer bestimmten Markup Sprache (z.B., HTML oder WML). Die Präsentation kann mittels XSL style sheets angepasst werden.

Anforderungen auf einer höheren Ebene, die Fallstudie (Teil 1)

- Das Conference Management System (CMS) bietet Unterstützung beim Einreichen einer wissenschaftlichen Veröffentlichung, nämlich die Auswertung und Anwahl für das Verlegen in einer Konferenz.

- Aktoren:
  - Das Präsidium leitet die gesamte Konferenz und stellt das Programm Komitee zusammen, in dem es die Mitglieder bestimmt.
  - Die Mitglieder des Programm Komitees leiten die Überprüfung der Veröffentlichungen, die sie vom Präsidium angewiesen bekommen haben.
  - Prüfer können von den Mitgliedern des Programm Komitees für die Auswertung der Veröffentlichungen ernannt werden.
  - Autoren können eine oder mehrere Veröffentlichungen einreichen.

- Alle Benutzer des Systems, außer das Präsidium, müssen sich registrieren um Zugang zu erhalten.

Anforderungen (Teil 2)

Die Konferenz durchläuft mehrere Phasen. Jede Phase definiert Aktivitäten die von verschiedenen Benutzergruppen durchgeführt werden:

- Die Organisation der Konferenz: Das Präsidium veranstaltet die Konferenz, legt die Themen und Fachgebiete fest und verzeichnet die vorläufigen Mitglieder des Programmkomitees.

- Das Einreichen einer wissenschaftlichen Veröffentlichung: Die Autoren können sich registrieren und Veröffentlichungen einreichen. Nach dem das Vorlagedatum vorbei ist, haben die Autoren keinen Zugang mehr zum System. In dieser Phase können die Mitglieder des Programmkomitees ihre bevorzugten Themen und Fachgebiete ausdrücken.

- Die Ausschreibung der wissenschaftlichen Veröffentlichungen: Die Mitglieder des Programmkomitees können die Veröffentlichungen nennen, die in den von ihnen bevorzugten Fachgebiete fallen und mindestens eins ihrer bevorzugten Themen entsprechen. Sie können auch Interessenkonflikte verkünden.

- Die Anweisung der wissenschaftlichen Veröffentlichungen: Unter Berücksichtigung der von den Mitgliedern des Programmkomitees angekündigten Interessen weist das Präsidum Veröffentlichungen an die Mitglieder des Programm Komitees für die Überprüfung an.

Anforderungen (Teil 3)

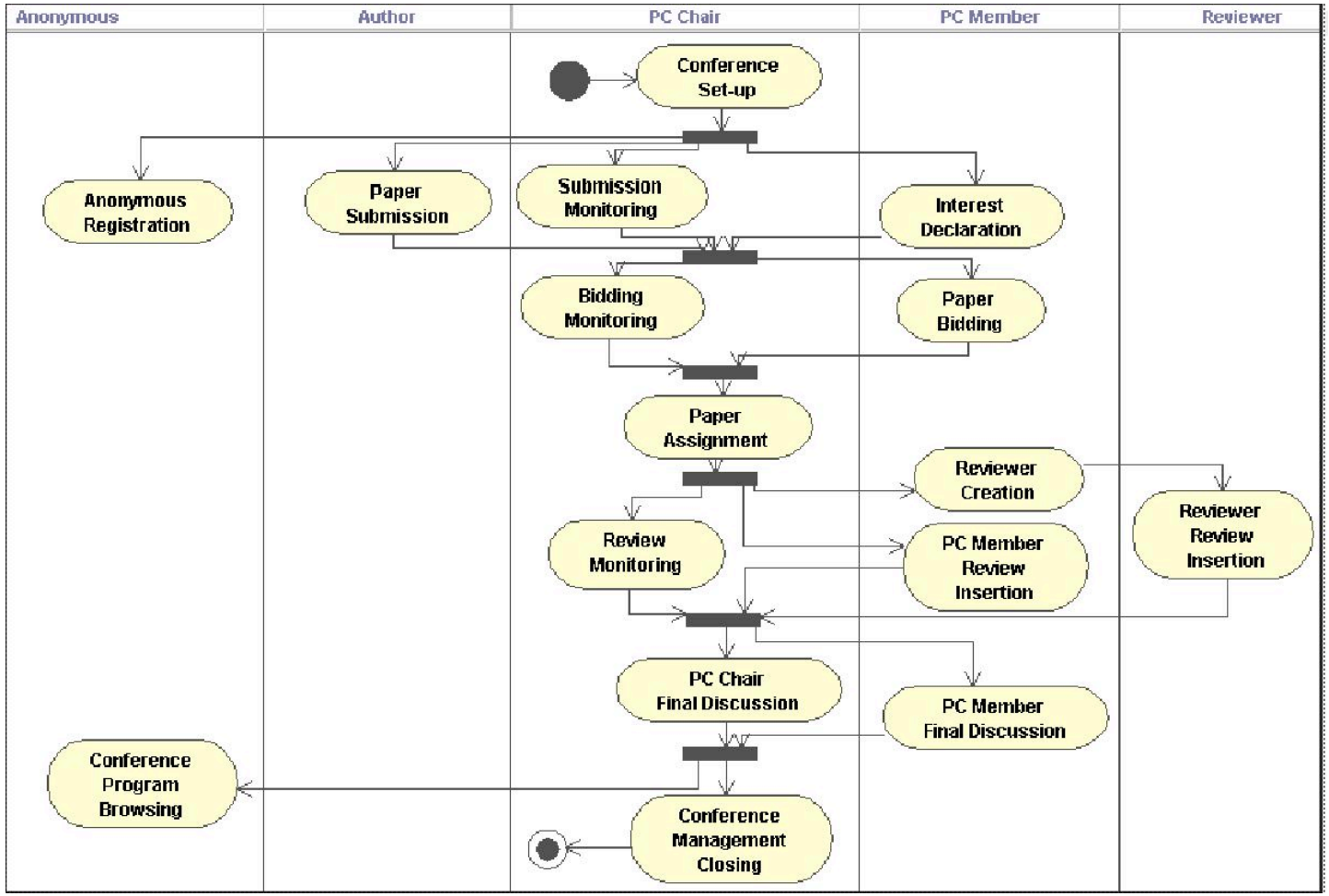
- Überprüfung der Veröffentlichungen: Die Mitglieder des Programmkomitees können andere Prüfer für die Konferenz registrieren und sie mit der Überprüfung der Veröffentlichungen beauftragen. Danach kann ein Mitglied des Programmkomitees oder ein Prüfer die Überprüfung einer Veröffentlichung sehen und bearbeiten bis sie als abgeschlossen gekennzeichnet wird. In dieser Phase kann sich jedes Mitglied des Programm Komitees nur dann auch Überprüfungen anderer Veröffentlichungen anschauen, nachdem sie ihre eigene Überprüfung abgeschlossen haben.

- Die endgültige Debatte: Die Mitglieder des Programmkomitees bekommen Zugang zu allen Veröffentlichungen, um sich auf die endgültige Debatte, die in der Komiteesitzung stattfindet, vorzubereiten.

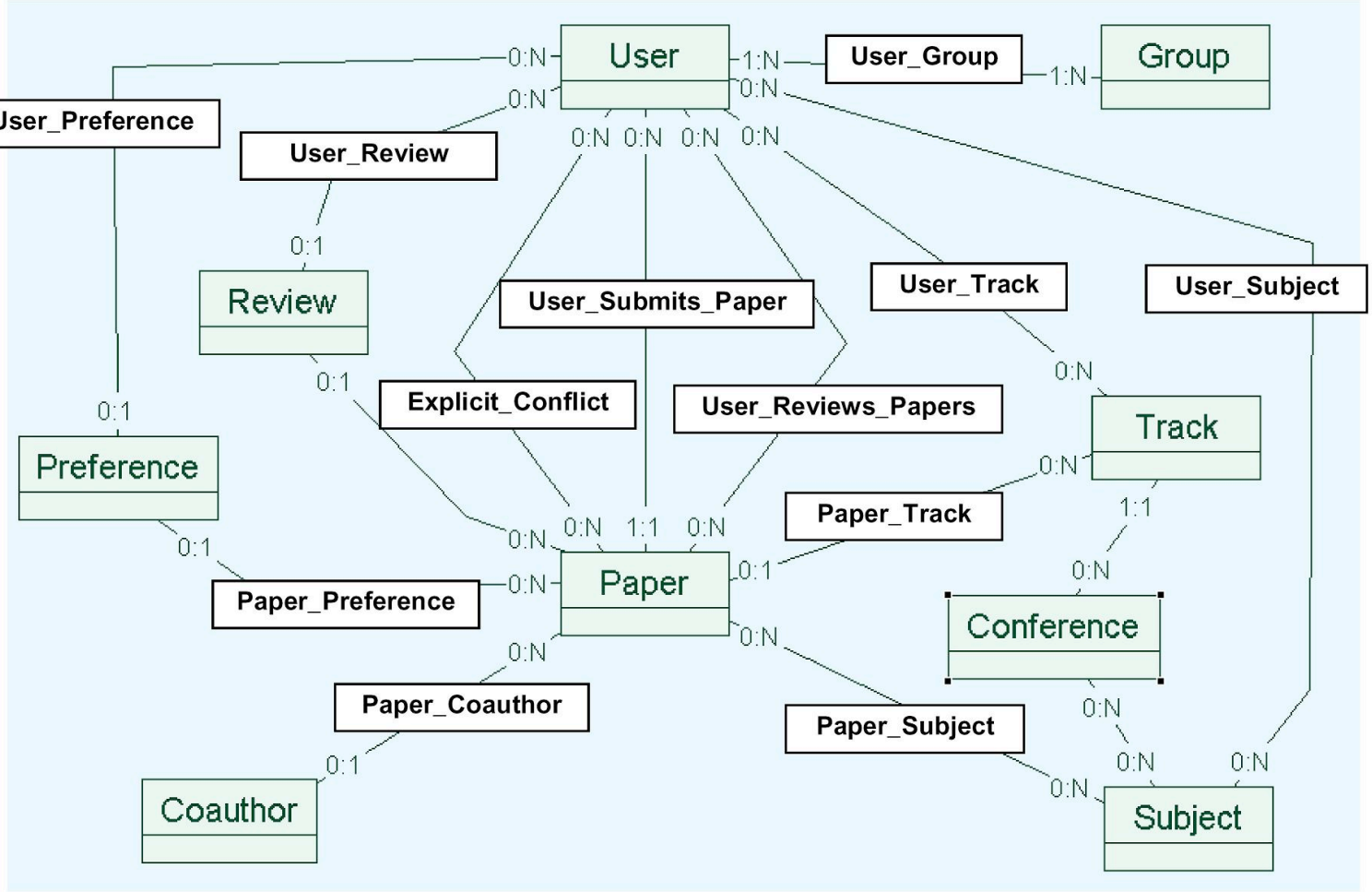
- Die Ankündigung der Akzeptanz: Das Präsidium verkündet die Akzeptanz oder Ansage der Veröffentlichungen. Im Falle der Akzeptanz wird der Autor mit einer Email Nachricht darüber informiert. Unabhängig von der Registiration werden alle Besucher der Webseite der Konferenz zu einer Seite mit dem endgültigen Programm der Konferenz geführt.

- Der Fortschritt des Konferenzablaufes durch die verschiedenen Phasen wird ausschließlich von dem Präsidum kontrolliert.

Aktivitätsdiagramm



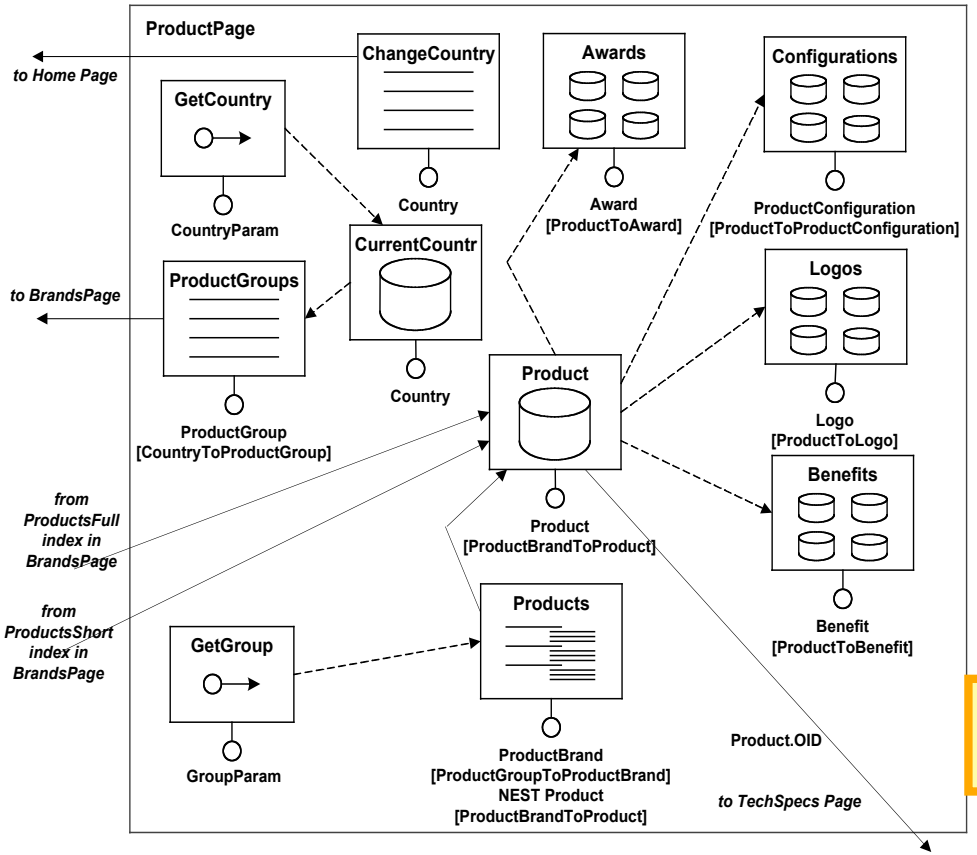
Struktur Modell

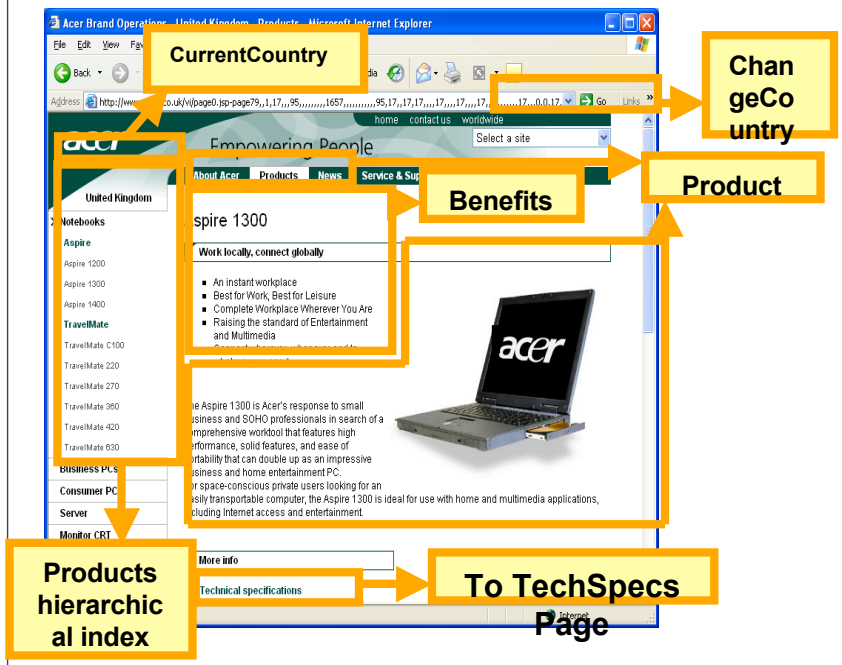


Hypertext Modell: Zuordnung von Site Views zu User Rollen (“access rights”)

Role / Phase	PC Chair	PC Member	Author	Reviewer (non-PC member)	Anonymous user
1. Conference setup	**Conference setup**	-	-	-	-
2. Paper submission	**Submission monitoring**	**PCM interest declaration**	**Author submission open**	-	**Registration**
3. Bidding	**Bidding monitoring**	**PCM bid**	**Login suspended**	-	**Everyone-Login**
4. Paper assignment	**Paper assignment**	**Login suspended**		-	
5. Paper review	**Review monitoring**	**PCM insert review**		**Reviewer insert review**	
6. Final discussion	**Final Discussion**	**PCM final discussion**		**Login suspended**	
7. Acceptance notification	**Conference Management Closing**	**Login suspended**			**Everyone–Conference program**

Design-time Modell vs. Runtime view

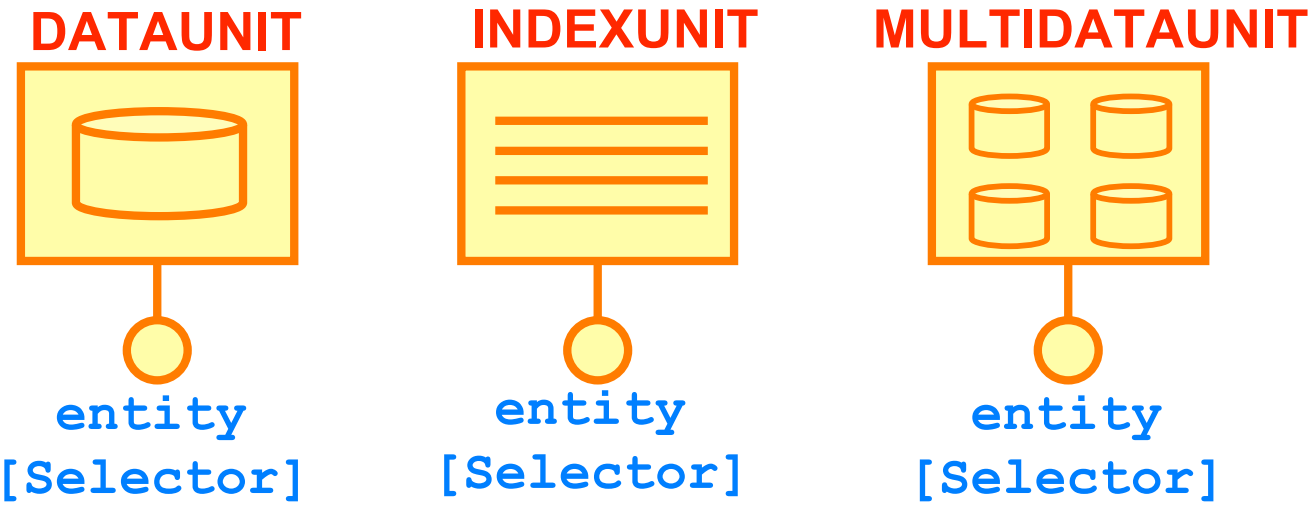




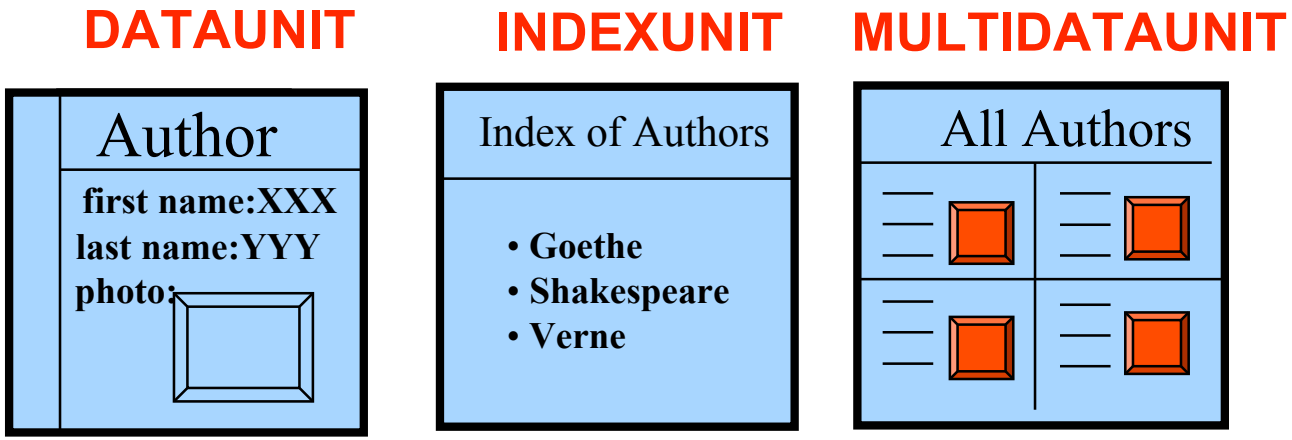
Content units in WebML (1 / 2)

Content:
• Instanzen von Entitäten

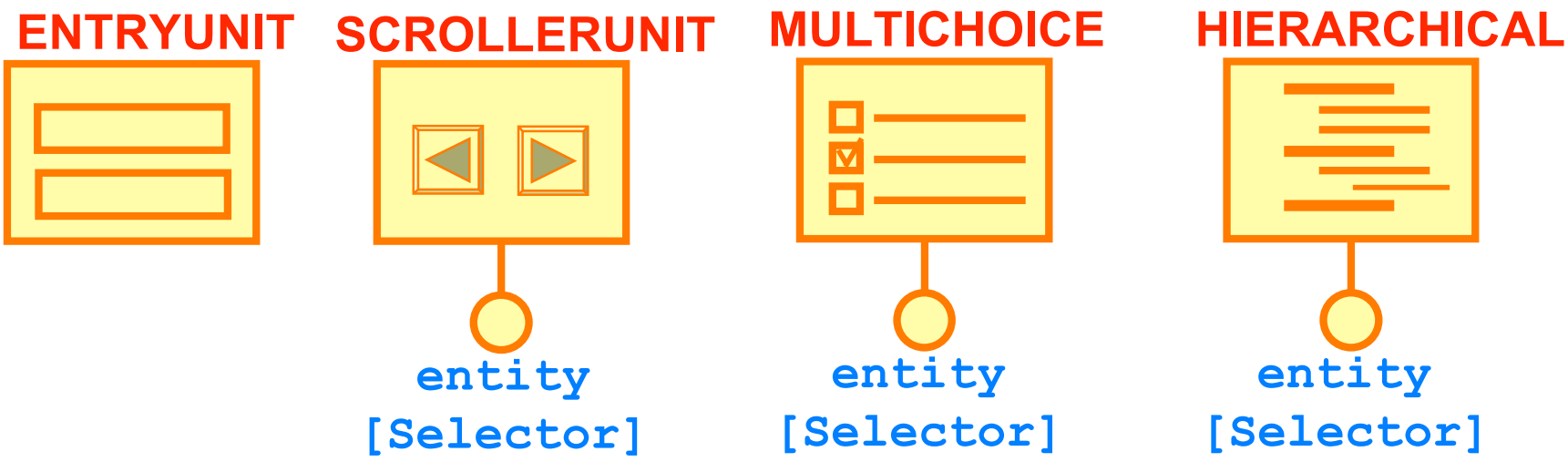
Selector:
• Bedingungen (z.B. "Name like Muste*"



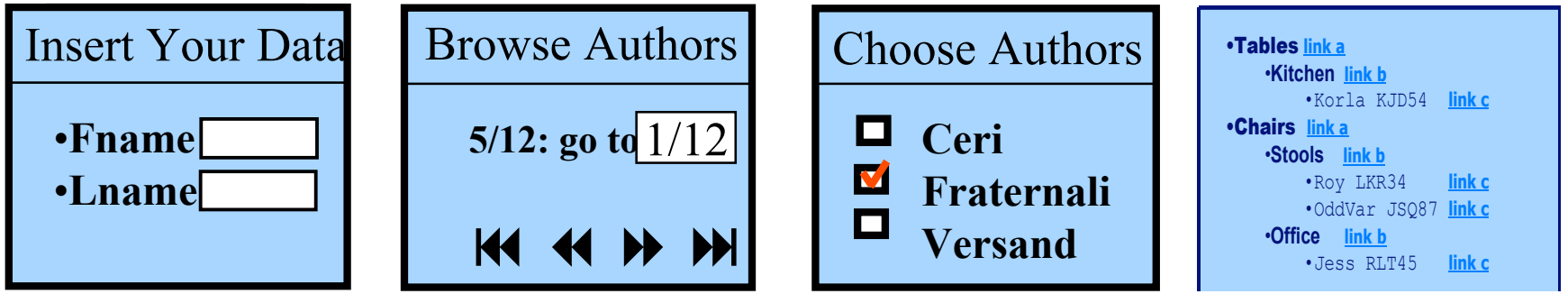
Beispiele:



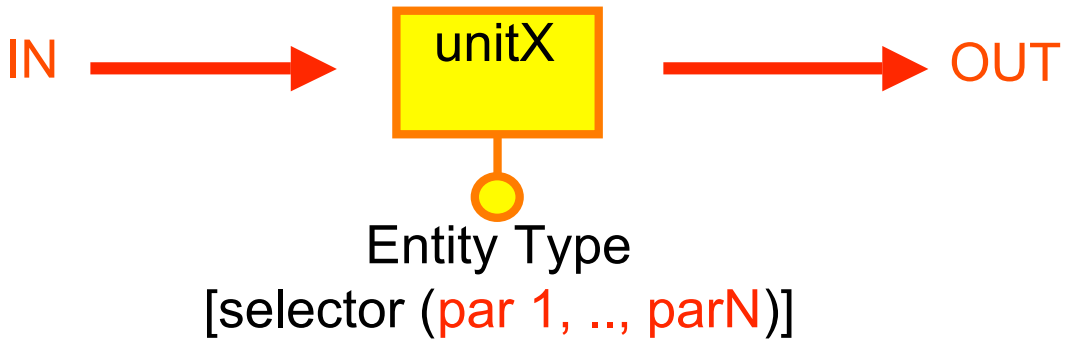
Content units in WebML (2 / 2)



Beispiele:

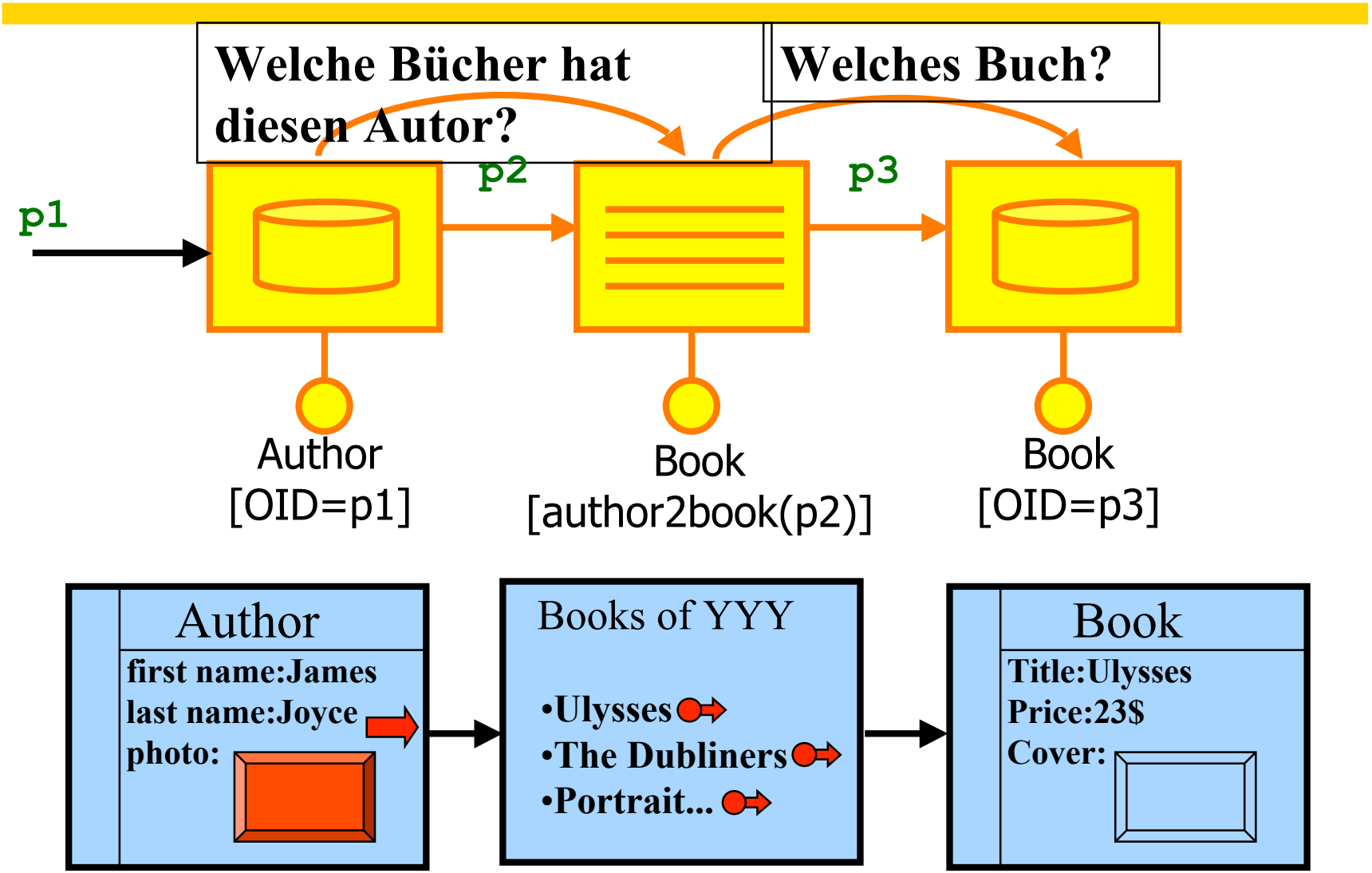


Wie Content Units funktionieren



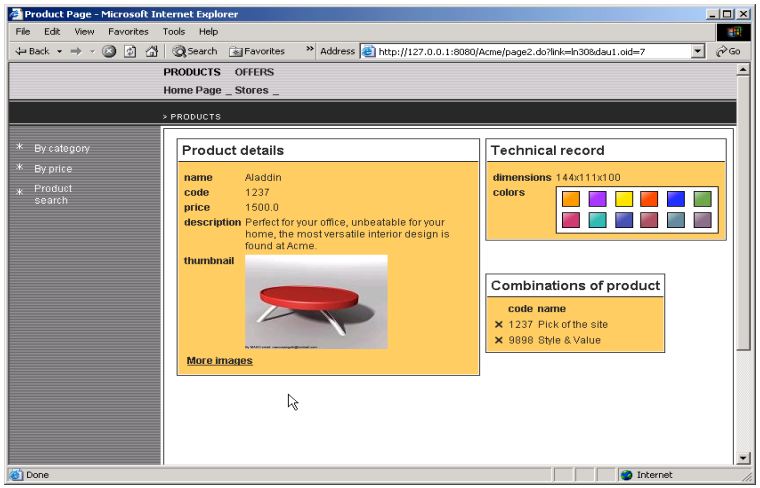
- Ein Content Unit braucht normalerweise ein “Kontext”, um seine Ergebnisse zu rechnen. Das Kontext enthält Ein- und Aus- Parametern

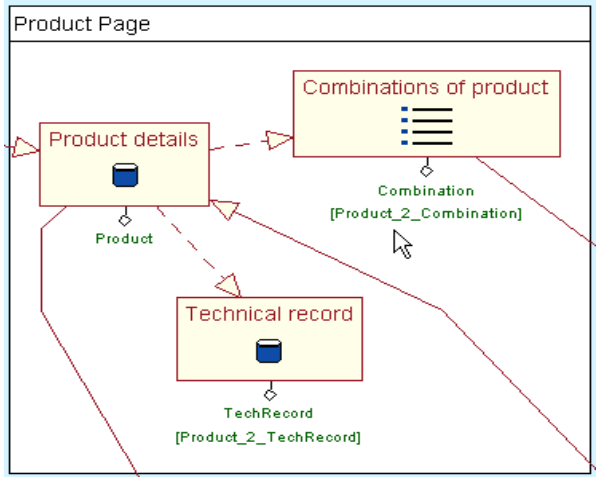
Content units, Navigation



Hypertext Hierarchie

- Eine Seite ist ein Container von (verknüpften) Content units, die wiederum eingeschaltete Seiten enthalten kann
- (“Parameter-less”) Links zwischen Seiten erlauben einfache Navigation
- Seiten werden in Bereiche (Engl. Areas) zusammengefasst, z.B. Sektionen eines Portals
- Ein Siteview ist eine Menge von Seiten und/oder Bereiche, die für eine Benutzer Rolle oder ein Gerät (WAP, Rich Client) bestimmt sind.





Operation units

- Aufruf von Backend Diensten (Anlegen, Storinierung einer Reservierung) wird durch Ikone modelliert im Hypertext

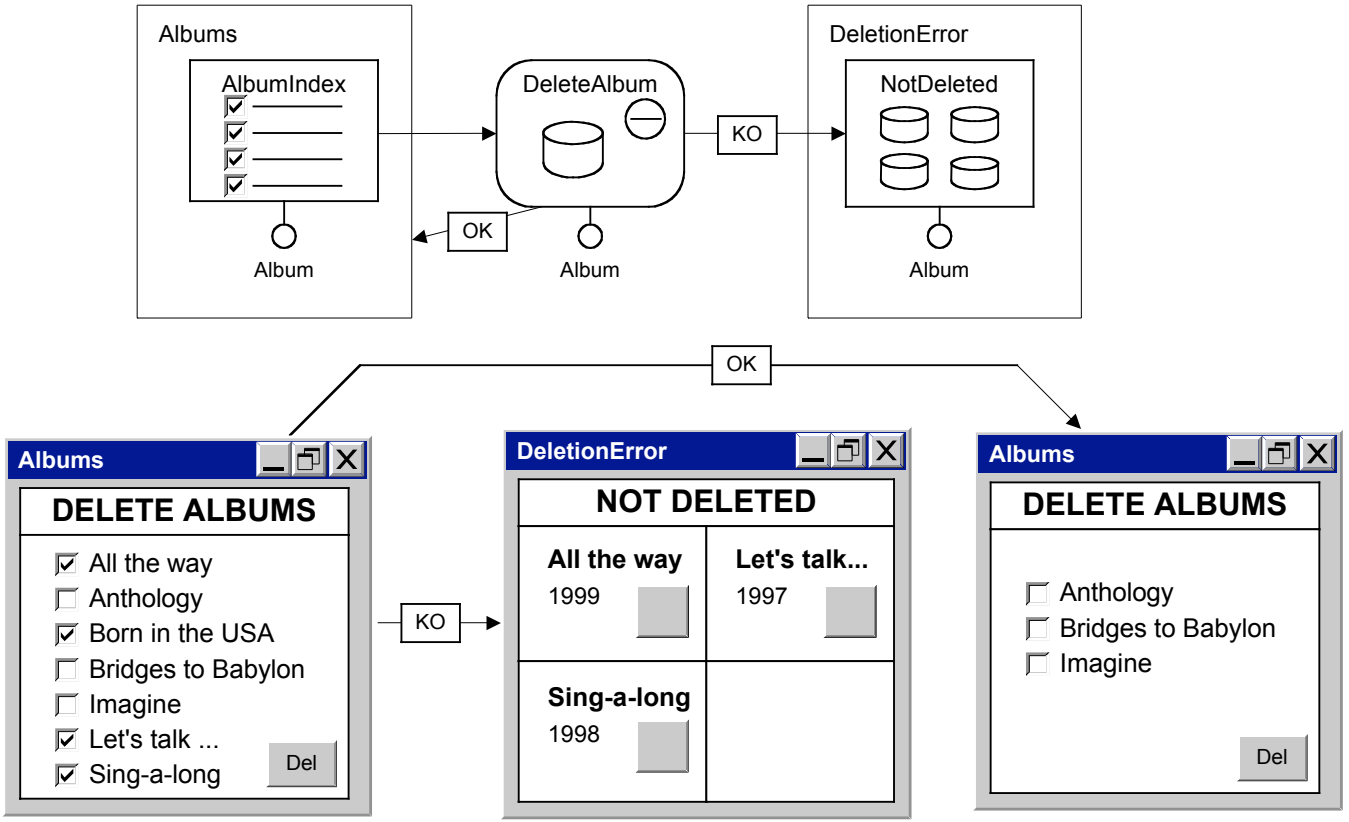
- Jede *Operation Unit* nimmt Eingaben von eingehenden Links und hat zwei ausgehende Links (OK, KO) für erfolgreiche bzw. Gescheiterte Durchführung

- Außer die vordefinierte WebML Einheiten können auch externe Operationen hinzugefügt werden

- Die eingebaute Operationen sind die traditionelle SQL Datenbank Operationen, wie z.B. INSERT, DELETE, UPDATE (auf Entitäten), erstellen und löschen von (n-ary) Beziehungen

- Zuzeit wird Notation in WebML eingeführt zu Beschreibung von Webservices und für Workflow

Beispiel (Delete)



Merkmale von WebML

- Visuelle Darstellungen sind zu Fragmenten einer Software Architektur gemappt (“pattern-based, model-driven software development paradigm”). Viele Werkzeuge basieren auf diesen Prinzip: AndroMDA, Versata, ArcStyler, OptimalJ, ArchitectureWare

- Keine zusätliche Customprogrammierung notwendig (kein Java, JSP, usw.)

Weitere Informationen zu Web CASE

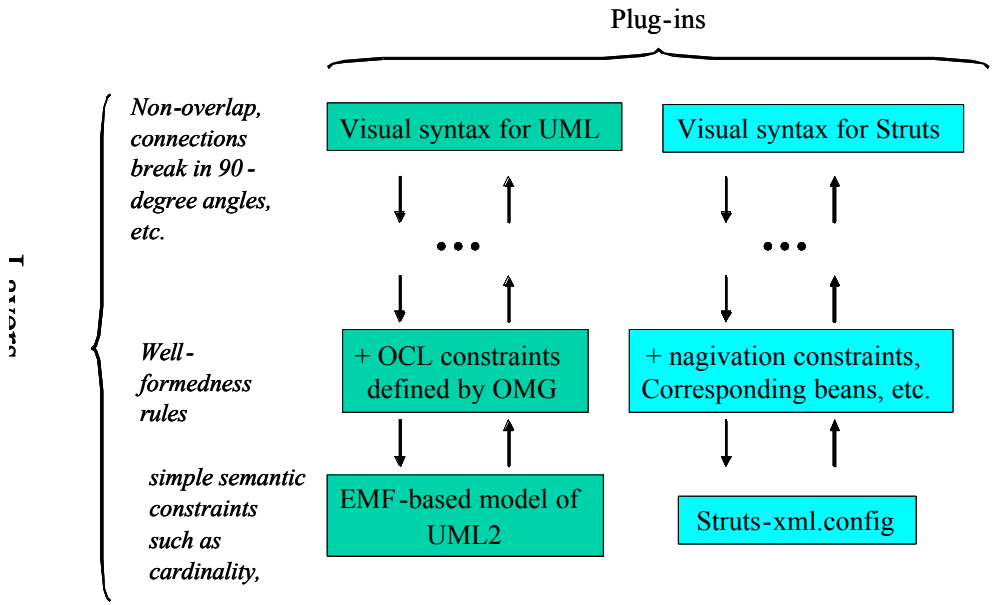
- Papers, Projekte, und Übungen: http://www.webml.org
- Evaluation version des Tools für Studenten zu download: http://www.webratio.com
- WebML Tutorial:
  S. Ceri, I. Manolescu, Constructing and integrating data-centric Web Applications: Methods, Tools, and Techniques, tutorial at VLDB 2003, Berlin
  http://www.webml.org/webml/upload/ent5/1/ceri-manolescu-VLDB_tutorial.pdf
- Dr. Reiko Heckel. Modellbasierte Entwicklung von Web-Anwendungen,
  http://www.uni-paderborn.de/cs/ag-engels/ag_dt/People/Heckel/DO/MEWA/
- Weitere Web modeling languages : HDM-W2000, OO-HDM, RMM, Araneus, Strudel, Tiramisu, UML Web Application Extension, WebArchitect.

Eclipse als MDA Plattform

- Eclipse ist ein Java IDE, aber auch ein Plattform mit der fähigkeit, Software Artefakten zu integrieren

Plug-ins manage software artifacts. They notify and accept commands to manipulate them, in accordance with well-formedness rules that were hardcoded (we cannot change them) by the plug-in writers. This is the crux of the problem in "post-fact tool integration“

[www.eclipse.org]

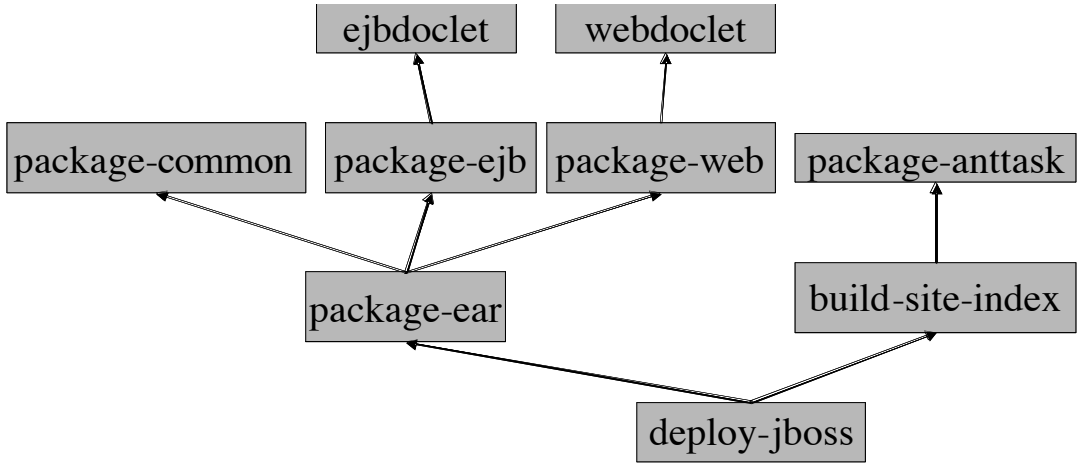


Weitere Informationen zu MDA

- http://www.davidfrankelconsulting.com/
- http://www.klasse.nl/english/mda/index.html
- http://www.omg.org/mda/
- http://www.systemwire.com/xlinkit/
- http://lci.cs.ubbcluj.ro/ocle/index.htm
- Eclipse projects: EMF, GEF, GMT, UML2
- http://www.eaipatterns.com
- http://www.umlcomponents.com/
- http://www.kc.com/

Ant: Eine Einführung

- Ant, http://ant.apache.org/, vereinfacht die Kompilierung von Java Anwendungen, unter Betrachtung der Abhängigkeiten zwischen Elementen. Z.B.:



- In Eclipse integriert!

  Ant in Eclipse Starthilfe,
  http://www.joller-voss.ch/tools/eclipse/ApacheAntInEclipseStarthilfe.pdf

[source: Developing J2EE Applications with Ant, Erik Hatcher]

build.xml

- XML Format
- Normalerweise in Projekt Root Ordner
- Deklarativ – Schritte werden definiert statt scripty Details
- Enthält alle Anweisungen um ein Projekt zu erstellen:
- Ein Projekt Enthält targets
- Targets enthalten tasks

```
<?xml version="1.0" ?>
<project name="antbook" default="default">
    …

    <target name="init">…</target>

    <target name="clean" description="Removes build artifacts">
      …
    </target>

    <target name="package-ear"
            depends="package-common,package-ejb,package-web"
            description="Package EAR">
      …
    </target>


    <target name="deploy-jboss"
            depends="package-ear,build-site-index"
            description="Deploy to local JBoss">
      …
    </target>

    <target name="default" depends="package-ear"/>

</project>
```

[source: Developing J2EE Applications with Ant, Erik Hatcher]

Command-line switches

- -projecthelp

```
> ant -projecthelp
Buildfile: build.xml
Main targets:

 checkstyle              Check code style
 clean                   Removes build artifacts
 compile-anttask         Compile anttask module
 compile-common          Compile common module
 compile-web             Compile web module
 deploy-jboss            Deploy to local JBoss
 ejbdoclet               Generate EJB code and descriptors
 generate-test-reports   Generate test reports
 package-ear             Package EAR
 package-ejb             Package EJB JAR
 package-web             Package WAR
 test-anttask            Test anttask module
 test-common             Test common module
 webdoclet               Generate web and Struts descriptors

Default target: default
```

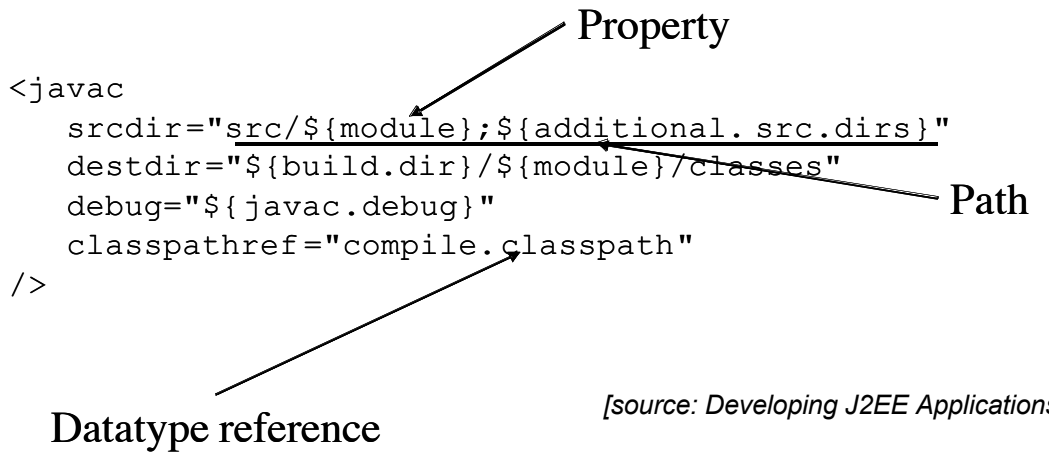
- -verbose / -debug
- -find <*file*>
- -diagnostics
- -logfile <*file*>
- -help
- *… and others*

[source: Developing J2EE Applications with Ant, Erik Hatcher]

Parameter für build.xml …

- All JVM system properties available
- Useful Ant built-in properties:
  - ant.project.name
  - basedir
- ${property.name} for property value

```
<property name="index.dir”
   location="${build.dir}/${site}/index”
/>
```



[source: Developing J2EE Applications with Ant, Erik Hatcher]

… erlauben Build Customization

- Z.B. unseres Projekt sollte für viele Webseiten leicht anpassbar sein. Wie kann man das machen?
- Mit der ${site} Property
- Artefakten die angepass werden müssen:
  - Index und Inhalt (natürlich)
  - application.xml (different WAR file names)
  - web.xml (different index directory references)
  - jboss.xml (different JNDI names for session bean)
  - search.jsp (results presentation differs)
- search.jsp customization:

```
<copy todir="${build.dir}/${site}" overwrite="true">
    <fileset dir="web" includes="search.jsp"/>
    <filterset>
       <filter token="LINK"
               value="${site.search.link}"/>
    </filterset>
</copy>
```

- WAR and EAR parametrisiert mit ${site}:

[source: Developing J2EE Applications with Ant, Erik Hatcher]

```
<war destfile="${dist.dir}/antbook-${site}.war"…>
```

Andere Werkzeuge bauen auf Ant auf

- Code Coverage: Clover
  http://www.thecortex.net/clover/index.html
- Code Generierung: XDoclet
  http://xdoclet.sourceforge.net/xdoclet/index.html