Target audience
- Process Engineers, Project Managers, Designers

Objectives
- To get an understanding of the RUP J2EE Developer Roadmap and how Agile concepts can improve the System Development Process

Non-Objectives
- Sell a Religion
Agenda

- Why Agile is a challenger to RUP...
- The RUP J2EE Developer Roadmap
- Agile Software Development
- A proposed RUP & Agile process based on practical experience
- References

A Meandering River of SD Methods

Old Functional SD Survivors
Old OO SD Survivors
RUP Dev Cases
UP

Why Agile

SIS/RAS
SVEA
MBI
Data Logic
Bus. Eng.

OMT
Booch
Fusion
Coad/Yourdon

UP & UML

SFFS
J2EE

LD
XP
AM
ASD, FDD...
Scrum, Crystal, DSDM...
Why Agile has become a Challenger to RUP

- Up front planning will be wrong anyway, adapt as you go along ...
- An advantage with non rigid methods, when one adjusts under way ...
- Messy unreadable UC doc’s, we must ask the Customer anyway ...
- Business req’s are unpredicatable and change all the time ...
- To build for future features will fail, build for min. req’s ...
- A timeboxed delivery and a Customer flood of must-req’s does not work ...
- Not having working code available often enough, a menace ...
- Always changing/increasing features creates competetiveness ...
- Believing in control, causing unpreparateness when "things happen".

Why Agile has become a Challenger to RUP

- A "rich" process does not compensate for less competence anyway ...
- IT personell making inappropriate business priorities, lack of collaboration ...
- Customers taking IT decisions, lack of collaboration ...
- Unsensitive, time pushing, ignorant, not caring project managers ...
- Buggy code with tons of useless documentation to go with it ...
- A lighter backpack makes the traveller more endurant ...
- Simple solutions are easier to adapt ...
- People get stuck in roles, causing resource allocation problems ...
- Agile is recognized as a more "open source" methodology than RUP ...

...
Why RUP still is strong

- Widely spread, a lot of people with RUP experience
- Serious RUP Customers have made tailored Development Cases
- Strong connection to UML
- Easy choice in Company policy …

Picking the candy from both bags
J2EE Technology “The big boxes”

J2EE Application Component Technologies
- Applets
- Servlets
- EJB’s
- Application Clients
- JSP’s

J2EE Services
- JAXP-XML parsing API
- JMS-std interface to messaging
- JDBC-Db Connection
- JAAS - Java Auth&Auth
- JTA - Java Tr. API

J2EE Services
- JavaMail - Mail API
- JCA - Java Connector Arch.

J2EE Deployment “The big boxes”

Client Device
- HTML Client
- Client Container
- Java Client

Web Server
- Web Container
- JSP
- Servlet

EJB Server
- EJB Container
- EJB
**RUP – J2EE Design**

**RUP Definitions**

**RUP**
Rational Unified Process
“A SD Process Framework from which customized Processes can be Developed”

**The J2EE Developer Roadmap**
A Customized SD process that has been tailored to meet the needs of the “J2EE Developer”. Developed by Eeles, Houston, Kozaczynski

**J2EE Developer**
“...responsible for taking a Vision of a System through to an Implementation of the System using the J2EE platform”
= J2EE Spec. Role “Application Component Provider”
### RUP – J2EE Design
### J2EE Developer Roadmap

<table>
<thead>
<tr>
<th>Disciplines</th>
<th>Workflow details</th>
<th>Activities</th>
<th>J2EE Specific Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requirements</td>
<td>Define the System</td>
<td>Capture a common vocabulary</td>
<td>Identify what J2EE patterns are going to be used</td>
</tr>
<tr>
<td></td>
<td>Prioritize Use-Case</td>
<td></td>
<td>Identify what J2EE technologies are going to be used</td>
</tr>
<tr>
<td>Analysis</td>
<td>Define an Initial Architecture</td>
<td>Architectural Analysis</td>
<td>Identify JSP’s, Servlets, EJB’s and other J2EE elements</td>
</tr>
<tr>
<td></td>
<td>Analyze Behavior</td>
<td>Model the User Experience</td>
<td></td>
</tr>
<tr>
<td>Design</td>
<td>Refine the Architecture</td>
<td>Identify Design Mechanisms</td>
<td>Describe the use of Java threads and message-driven EJB’s</td>
</tr>
<tr>
<td></td>
<td>Identify Design Elements</td>
<td></td>
<td>Map J2EE modules to nodes</td>
</tr>
<tr>
<td></td>
<td>Incorporate existing Design Elements</td>
<td></td>
<td>Describe the interactions between collaborating J2EE elements</td>
</tr>
<tr>
<td></td>
<td>Describe Concurrency and Distribution</td>
<td></td>
<td>Describe subsystems in terms of their internal J2EE elements</td>
</tr>
<tr>
<td></td>
<td>Use-Case Design</td>
<td></td>
<td>Produce a detailed design of EJB’s</td>
</tr>
<tr>
<td></td>
<td>Subsystem Design</td>
<td></td>
<td>Produce a detailed design of JSP’s, Servlets and other Java classes</td>
</tr>
<tr>
<td></td>
<td>Component Design</td>
<td></td>
<td>Define the mapping between entity EJB’s and the underlying database</td>
</tr>
<tr>
<td></td>
<td>Class Design</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Database Design</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
RUP – J2EE Design
J2EE Specific Content Highlighted

- Identify what J2EE patterns are going to be used
- Identify what J2EE technologies are going to be used
- Identify JSP’s, Servlets, EJB’s and other J2EE elements
- Describe the use of Java threads and message-driven EJB’s
- Map J2EE modules to nodes
- Describe the interactions between collaborating J2EE elements
- Describe subsystems in terms of their internal J2EE elements
- Produce a detailed design of EJB’s
- Produce a detailed design of JSP’s, Servlets and other Java classes
- Define the mapping between entity EJB’s and the underlying database

RUP – J2EE Design
3 Contributions

- Predefined Input from Analysis
- A configured chain of worksteps for J2EE Design, with J2EE specific content
- UML mapping to J2EE concepts and suggested package structure in Design Model is done as part of the Roadmap
RUP – J2EE Design
1 Input from Analysis

Use-Case Diagrams

Analysis Classes

Class Diagrams

Sequence Diagrams

RUP – J2EE Design
2 A “clear” chain of worksteps ...

Refine the Architecture

Detail the Design
RUP – J2EE Design
3 UML Mapping – UML Package to J2EE Components

J2EE Component Structure in Java Petstore

Web Tier

EJB Tier

1 WAR

7 EJB-JAR

RUP – J2EE Design
3 UML Mapping – UML Components to J2EE EJB’s

J2EE EJB Model for the Customer Component in Java Petstore
RUP – J2EE Design
3 UML Mapping – UML Class to EJB Implementation

<<EJBRemoteInterface>>
Customer
changeContactInformation()
createAccount()
<<EJBServiceImplInterface>>
CustomerHome
<<EJBCreateMethod>>
<<EJBBusinessMethod>>
changeContactInformation()
createAccount()
<<EJBHomeMethod>>
accountHomeRef()

RUP – J2EE Design
3 Design Model Package Structure

Design Model
Package Structure of Java Petstore based on “the J2EE Developer Roadmap”
Manifesto for Agile Software Development

“We are uncovering better ways of developing software by doing it and helping others to do it. Through this work we have come to value:

• **Individuals and interactions** over process and tools
• **Working software** over comprehensive documentation
• **Customer collaboration** over contract negotiation
• **Responding to change** over following a plan

That is, while there is a value in the items on the right, we value the items on the left more.”

Kent Beck, Mike Beedle, Arie van Bennekum, Alistair Cockburn, Ward Cunningham, Martin Fowler, James Grenning, Jim Highsmith, Andrew Hunt, Ron Jeffries, Jon Kern, Brian Marick, Robert C. Martin, Steve Mellor, Ken Schwaber, Jeff Sutherland, Dave Thomas

Core Agile Methodologies & Practises

• LD – Lean Development (Robert N Charette)
• ASD – Agile Software Development (Jim Highsmith)
• Scrum (Ken Schwaber et al)
• XP – eXtreme Programming (Kent Beck et al)
• Crystal methods (Alistair Cockburn)
• FDD – Feature Driven Development (Jeff De Luca, Peter Coad...)
• DSDM – Dynamic Systems Development Method (Robin Smith et al)
• AM - Agile Modeling (Scott Ambler)
Agile Software Development
Main Focus

Main focus of Agile Methodologies & Practises

- A Customer focused way of working with many deliveries that requires an active Customer participation in the actual development work.

- A Feature driven way of working, delivering at each time “good enough functionality”, continuously reevaluating what’s been delivered and what’s to be delivered in the next iteration.

- A way of working that encourage changed requirements at any time. The customer not only adds requirements but also cut off req’s, focus on timebox.

- The project is setup to use a minimum of up-front planning and documentation. Instead focusing on constant adoptions and face to face communication between all team members. Working software is a primary focus, documentation secondary.

- Self organized Project teams, constantly reevaluating their way of working and changing accordingly. A major focus is on motivated and trusted individuals.

Agile Software Development
Prereq’s

- A trusting Management
- A participating Customer that takes decisions
- A self organized very skilled development team with XP mindset
- A non formal project management with focus on people
3 J2EE Design Contributions

- A New/Changing Architecture such as J2EE benefit from a built in **constant process improvement**
- Major focus is on **Continuous Design** not a heavy initial one
- Agile SD Design is built on extremely usefull **Design principles and Patterns**

---

**A proposed SD Roadmap based on RUP and Agile SD**

Overview

- Start
- High Level Requirement Analysis
- Detailed Analysis of Main Scope
- Min. Design & Implementation
- High level descriptions of Func. Req's and Non. Functional Req's clustered into Delivery Iterations based on Priority
- A Detailed Analysis of Req's into Analysis Classes with Operations. Work based on High Level Analysis as well as previous iteration
- Axi: Design with focus on each Delivery & Test Driven Implementation. Deeps follow agile discipline
- Feedback on Delivered Features as well as Working practices
- Customer feedback on delivery
- End

[Diagram of SD Roadmap]
References:

Building J2EE Applications with the Rational Unified Process
ISBN 0-201-79166-8 Peter Eeles, Kelli Houston, Wojtek Kozaczynski

Agile Software Development principles, patterns and practices

eXtreme Programming in Practice

Planning eXtreme Programming
ISBN 0-201-71091-9 Kent Beck and Martin Fowler

Links on Agile

http://www.agilealliance.com/home
http://www.extremeprogramming.org/
http://www.agilemodeling.com/
http://crystalmethodologies.org/
http://www.dsdm.org/kss/default.asp
http://www.featuredrivendevelopment.com/
http://www.controlchaos.com/