Using BPEL as a workflow engine for local enterprise applications

Nicolas Biri, Pascal Bauler, Fernand Feltz, Nicolas Médoc, Céline Thomase
CRP – Gabriel Lippmann
EMISA 2007
Overview

• Luxembourgish context
• Project context of the Luxembourgish National Family Benefits Fund
• Agile approach applied to the specific project context
• Implications onto underlying IT architecture
• Importance of modeling
  – Business Processes
  – State machines
  – Model Driven Software Design
• Conclusion
Population: 459 500
39.6% non-luxembourghish citizens: > 180 000
In addition > 100 000 commuters

Annual country budget:
7.62 Billions €
Family allowances:
> 755 Millions €
(~10%)
e-Government in Luxembourg

Country-wide e-Government projects

Pension fund
Social insurances
Family allowances fund
Local infrastructure
Local applications

CISS
(Centre Informatique de la Sécurité Sociale)
(central data center of the social security sector)

CIE
(Centre Informatique de l'Etat)
(central data center of the government)

Customs office
Tax office
Environment office
Local infrastructure
Local applications

Project context

Shared infrastructure
Shared applications
Family Allowances and Commuters

Year 2005:

Commuters: 116381
France: 60027
Belgium: 31254
Germany: 25100

Workforce of the Family Benefits Fund is unchanged
Modernization project

• Phase 1: Urgent needs
  - French borderline commuters
  - Semi-automatic solution
  - Deadline: 8 months [August 2005]
  - Extended to German + Belgium commuters [end 2005]

• Phase 2: Automated solution for French commuters
  - Based on a long-term project [origins in 2001]
  - Our involvement started Q1 2006
  - Expected end: October 2007
Project overview

Data Exchange Protocol

Luxembourgish Back-end

• Java EE based
• New development
• No historical data

French Back-end

• Mainframe base
• Enhancement of operational system
• Historical data

Agile
  - 2 - 4 weeks cycles

Waterfall
  - 6 month cycles
International Data Exchange Protocol

- Global structure
  - Master Data
  - Error handling protocol
  - Normal Data Exchanges: country specific allowances

- Combining “Waterfall” and “Agile” development approaches

=> Data Exchange protocol defined based on a waterfall approach
<table>
<thead>
<tr>
<th></th>
<th>January 2006</th>
<th>June 2006</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Exchange spec.</td>
<td>OK</td>
<td>KO</td>
</tr>
<tr>
<td>Application req.</td>
<td>OK</td>
<td>KO</td>
</tr>
<tr>
<td>Development (France)</td>
<td>OK</td>
<td>On-going</td>
</tr>
<tr>
<td>Development (Lux.)</td>
<td>KO</td>
<td>On-going</td>
</tr>
</tbody>
</table>
Agile approach

- Iterative approach
- Prototype based development
- SCRUM
  - Short development cycles: Sprints
    - 4 weeks cycles first
    - 2 week cycles later
  - List of task items: Backlog
  - Review
    - practical demonstration: stakeholder / technical team
  - Tool support: ScrumWorks Danube
    - Documentation
    - Follow-up
  - Daily Meetings [short, ad-hoc]
  - Scrum “Management”, Scrum Master, Product Owner
    - Not really applied (small team)
• **Adequate IT tools**
  - Model Driven Software Development
  - Workflow systems

• **Skeleton based approach**
Degeneration of the IT architecture

- Client (Java SWT)
- JBoss Application Server
- Facade
- Business Logic
- Persistent Objects
- Database
  - Views
  - Tables

Integration

Reporting
Impact of Maintenance onto IT architecture

Client (Java SWT)

JBoss Application Server

Business Logic

Persistent Objects

Database

Views

Tables

Integration

Reporting
Reasons / Explanations

- Human error
- Time-pressure (debugging)
- Un-experienced developpers

- No systematic/automated control
- No notification of problems

=> Unsustainable systems
Proposed solution

Clear separation between application code and business processes

Client (Java SWT)

Integration

Workflow Engine

Workflow Engine

Workflow Engine

JBoss Application Server

Facade

Business Logic

Persistent Objects

Database

Views

Tables

Client (Java SWT)
Discussion

• Advantages
  - Visual representation of business processes
  - No accidental violation of architecture
  - Gain in flexibility and maintainability
  - Improved management/monitoring (workflow engine)

• Disadvantages
  - Increased complexity
  - Distributed application (Workflow Engine, Web Application Server)
  - Synchronisation between workflow and application state
Business Process Execution Language

• Business Process Execution Language (BPEL)
• OASIS: BPEL 1.1 2003; BPEL 2.0 2007
• Web Service Based
• Defacto SOA Orchestration Standard
  – Orchestration of distributed / heterogeneous services
  – Local applications are a simplified subset
• BPEL is “The Silver-Bullet” 😊

Have you ever seen a running BPEL process?
Sample BPEL process: Statistics
- 473 lines: Process description
- 41 lines: Deployment descriptor
- 114 lines: WSDL: Web Service Description Language
Total: 628 lines of XML (partly generated, manually adapted)

- Extremely technical
- Many limitations
  (only a sub-set of the workflow patterns are supported by BPEL)
BPEL and Agile development

**Iteration 1**
- Initiate
- Assign
- Control
- Assign
- Compute

**Iteration 2**
- Initiate
- Assign
- Treat
- Receive
- Block file
- Launch next process
- Relaunch file

**Iteration 3**
- Initiate
- Assign
- Assign
- Error handling
- Assign
- Assign
Types of communications

- BPEL -> Application
- Application -> BPEL
- BPEL <-> BPEL

Message centric communication (WS)

Possible problems:
De-synchronisation between BPEL process and Application flow
De-synchronisation problem

Workflow

Application

Failure

De-Synchronisation
Proposed solution: state-machine

Workflow

State Diagrams

Application

Failure

Strong Coupling

generated by means of Model Driven Software Development
Model Driven Software Development

- Abstraction layers: M0, M1, M2, M3
- Specific Meta-Models
  - Enterprise Java Beans 3
  - State-machine
  - Logging (Historical Data)
- Appropriate Frameworks

```
M 0
M 1
M 2
M 3
```

- Meta-Meta Model
- Meta Model
- Model
- Code
Agility and MDSD

Software upgrade due to Meta-Model upgrade

Meta-Model upgrade due to software upgrade
Conclusion

• BPEL as a workflow engine for local enterprise applications?
  - Agile development possible with BPEL
  - Need for appropriate frameworks (Workflow and application flow)
    => Model Driven Software Design
  - Conceptually OK
  - BPEL issues
    • Technical
    • Complex
    • Limitations (arbitrary cycles)
    • Too low level

• BPEL is at the beginning (comparable to EJB 2.0)
• Possible direction: BPEL generated out of BPMN?
Thank you for your attention
Any questions?