A Service-Oriented Architecture for Software Process Technology

Therese Helland
Computas AS
Norway
the@computas.com

John Grundy \(^1, \, 2\) and John Hosking\(^1\)
\(^1\)Dept. of Computer Science and
\(^2\)Dept. of Electrical and Computer Engineering
University of Auckland New Zealand
{john-g, john}@cs.auckland.ac.nz
Outline

• Aims and requirements
• IMÅL approach and decentralised architecture
• PPML: Process modelling language
• Example usage
  – Modelling and enactment
  – To do list
  – Process automation
• Design & Implementation
• Evaluation
• Summary
Aims and Requirements

• **Aim:**
  – A tool suite providing a service oriented approach to software process modelling and enactment

• **Requirements:**
  – Configurable process modelling notation
  – Modelling and enactment for simple and complex process flow
  – Decentralised process management based on a set of pluggable services
  – Reuse COTS services as components
IMÅL approach

- **Simple and tailorable modelling language**
  - Has enactment visualisation capability
  - Based on our Serendipity language

- **Highly distributed service oriented process enactment engine**
  - Multiple process engine services
  - Process state, notification, login, to do list, etc services

- **3rd party component integration:**
  - InfoPath: document display
  - Idiom: rule-based decision making
Process Modelling Notation: PPML

Start stage

Base stage

Service

Flow

Stop stage

RoleActor

Choice

Resource

Design Visio doc

if(I > 0)

else

Design

Visio doc

if(I > 0) else

Designer

Therese

Coder

Nigel

Claire

Tester

Therese

1 Design Add-on

Designer

design vsd

MS Visio

design the add-on

2 Code Add-on

Coder

code java

JCreator

code the add-on

SimpleFlow

3 Test Add-on

Tester

test last doc

MS Word

test Add-on

SimpleFlow

Finish
Process modelling & enactment
Web based to-do list

### Process:

**UpdateSoftwareA**

### To-Do Items for Therese:

<table>
<thead>
<tr>
<th>TASK #</th>
<th>TASK NAME</th>
<th>DESCRIPTION</th>
<th>ARTEFACT</th>
<th>STATUS</th>
<th>ACTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Design Add-on</td>
<td>design the add on</td>
<td>design unit</td>
<td>completed</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Test Add-on</td>
<td>test Add-on</td>
<td>test doc</td>
<td>ready</td>
<td>enact 3</td>
</tr>
</tbody>
</table>

### Overall Process Information:

<table>
<thead>
<tr>
<th>TASK NAME</th>
<th>DESCRIPTION</th>
<th>RESPONSIBLE</th>
<th>STATUS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Design Add-on</td>
<td>design the add on</td>
<td>[Therese]</td>
<td>completed</td>
</tr>
<tr>
<td>Code Add-on</td>
<td>code the add on</td>
<td>[Klaus Nagel]</td>
<td>completed</td>
</tr>
<tr>
<td>Test Add-on</td>
<td>test Add-on</td>
<td>[Therese]</td>
<td>ready</td>
</tr>
<tr>
<td>Check Time Spent</td>
<td>check time spent on process &amp; email notification</td>
<td>[AUTOMATED]</td>
<td>completed</td>
</tr>
</tbody>
</table>
Document viewing and editing

- Uses Microsoft InfoPath for form based interaction
- InfoPath server provides web based data i/o
Rule based decisions

- Uses Idiom as an alternative to simple flow engine
- Provides complex rule-based processing logic
- Can also automate activities based on rule-based decision making
  - Eg send an email if a computed trigger value is above a threshold
Design and Implementation

• Experimented with co-operating services to realise complex processing modelling and enactment tool

• Key components:
  – Process modelling & enactment tool
  – Main process engine
  – Process & State Database services
  – Simple & complex flow services
  – Refresher & To do list services
  – Tool, automation & external services
  – Document flow service
  – UDDI registry
IMÅL’s Architectural Design
Implementation

- Pounamu meta tool used to implement PPML
  - Look and feel readily modifiable
  - Functionality extendible
- Inter-component communications uses SOAP
  - Java & JWSDP
- SQL Server storage
- JSPs for web/thin client interfaces
Evaluation

• Three evaluations:
  • User evaluation
    – Favourable response, partic on usability
    – Some issues around simple/complex flow
    – Safety of look and feel mods
  • Cognitive dimensions
    – See written paper
  • Performance test
    – Can support over 100 concurrent users @ 1 enactment event/user/second
    – But dependent on automated task level
Conclusions & future work

• Demonstrated feasibility of distributed service oriented process modelling and enactment approach
  – Proof of concept implementation
  – Usability and performance shows good results

• Future work
  – Replicated instances of each service for fault tolerance
  – More versions of engine, notification services etc
References


