CSE 135

Data-driven descriptions of the web application process & design patterns

The Larger Issue: Specification and Modularization

• Very soon you will not be a solitary hacker!
  – Customers w/ contracts,
  – Product managers who need to know what you build without looking into code,
  – Engineering managers,
  – Architects

• You will deal with frictions in specifying an application and turning specification into code systematically

• Inefficiencies in large project management
The Process and the Frictions

Business Process Owner (Client)

**Analysis/Specification Phase**

**COMMUNICATION**
- business process and specification of Web application
- Informal, imprecise specification by customer
- Accompanied by hard-to-built demos and diagrams

Chief Architect/Technical Project Leader

**Development Phase**

**COMMUNICATION**
- technical specification and development
- Code developed may be inconsistent with spec
- Significant effort in communicating spec formally

Developer

Problem is even worse in evolution phase when application logic is hidden in thousands of lines of code

The Problem: Communication

| Scene 1: The customer explained it | Scene 2: How the Project Leader understood it | Scene 3: How the Programmer wrote it | Scene 4: How the customer was billed | Scene 5: The customer really needed it |
Application specification

- Lightweight, process-oriented approaches
  - BPML describing sequence of steps
  - The specifier can leave too much detail out

- Close-to-the-code approaches
  - UML
  - Too heavy
  - Not useful to people interested in application behavior but not really its implementation details

Collaborative Dataflows

- Business process specification community realizing that data matter
  - Non-data-oriented precursors

- IBM’s Siena and Artifact-driven approaches
  - artifact = object, tuple

- Dataflow: An adjustment for human-centric web applications

- Informally, a dataflow summarizes the application using graph involving user groups, pages, actions and edges that describe the effect of actions on what users can do
  - Actions pertain to a context consisting of one or more “artifacts”
### WebDB 2010

#### Review Assigned Papers

<table>
<thead>
<tr>
<th>Title</th>
<th>PDF</th>
<th>Screenshot</th>
<th>Review</th>
<th>Comment</th>
<th>Score</th>
<th>Invite Author Feedback</th>
</tr>
</thead>
<tbody>
<tr>
<td>AquaLogic Data Services</td>
<td>aqua_log.pdf</td>
<td><img src="image1" alt="Screenshot" /></td>
<td>Novel idea, but is it practical?</td>
<td>?</td>
<td>Yes</td>
<td>edit</td>
</tr>
<tr>
<td>Kepler Scientific Workflows</td>
<td>kepler.pdf</td>
<td><img src="image2" alt="Screenshot" /></td>
<td></td>
<td></td>
<td></td>
<td>No</td>
</tr>
</tbody>
</table>

Showing 2 records (1 - 2 of 2)

---

### WebDB 2010

#### Provide Feedback

<table>
<thead>
<tr>
<th>Title: AquaLogic Data Services</th>
<th>PDF: aqua_log.pdf</th>
</tr>
</thead>
</table>

**Screenshot:**

![Screenshot](image3)

**Review:**

<table>
<thead>
<tr>
<th>Comment</th>
<th>Score</th>
<th>Invite Author Feedback</th>
</tr>
</thead>
<tbody>
<tr>
<td>Novel idea, but is it practical?</td>
<td>?</td>
<td>Yes</td>
</tr>
</tbody>
</table>

**Feedback:**

<table>
<thead>
<tr>
<th>Comment</th>
<th>Submit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Of course!</td>
<td>edit</td>
</tr>
</tbody>
</table>
The observable behavior of a collaborative web application

- The state of a collaborative human-centric application is a set of facts describing user rights at any given time
  
  \textit{readRecord(user, context)}
  
  \textit{someAction(user, context)}
  
- A run of the application is a sequence of states, where the transitions are caused by actions
  
- Our goal: Describe as much as possible about the possible runs using the data flow

At action level: How a user’s actions enable other users

When a \textit{ChairAssigns} (an assignment) to a reported pair of Paper and \textit{Reviewer}, then this \textit{Reviewer} can read this Paper
Project

• Create an E/R database design
  – A little touch-up to turn entities into artifacts

• Create data flow

• Make demo snapshots for the pages
  – An interesting snapshot has at least some data and illustrates actions that can be performed

• Add narrative/story around pages

• Submit a zip by Monday midnight