Manage User Evaluation in Software Development Environments

Yael Dubinsky
DIS seminar, March 2009

Joint work with Shah Rukh Humayoun and Tiziana Catarci
Accepted as a research demonstration paper at ICSE 09
Presenting Myself

- **Research**
  - Main interest: software engineering
  - Research colleagues: IBM, University of Rome, American University, Technion

- **Teaching**
  - Teaching project-based courses
    - CS, Technion

- **Practice**
  - Guiding agile software teams
    - IBM, Cisco, IDF, Ness, Rafael, Orca, Aladdin and more
Agenda

- Problem statement
- Solution suggested
- Presenting a prototype
- Summary
SE and HCI Communities

- Two different disciplines
  - Teaching, research, practice
- No clear way for merging activities
  - Tools and techniques, roles and responsibilities

- Project failures
  - Among others, because of a lack of usability and inefficient design [Landauer, 1995; Norman, 2006]
Agile Software Development

Main ideas:
- Customer collaboration
- Exhaustive automating testing
- Embrace change
- Small iterations
- Track your process
The Working Environment – Agile Room
User Centered Design (UCD)

- “… UCD is an iterative process whose goal is the development of usable systems, achieved through involvement of potential users of a system in system design." [Karat, 1996]
- “… user-centred design emphasizes that the purpose of the system is to serve the user, not to use a specific technology, not to be an elegant piece of programming. The needs of the users should dominate the design of the interface, and the needs of the interface should dominate the design of the rest of the system." [Norman, 1986]
Agile + UCD

- Customer collaboration
- Exhaustive automating testing
- Embrace change
- Small iterations
- Track your process
- End users collaboration
- Automate user experiments
- Iteratively evaluate along the process
- Add UCD metrics
UEMan (User Evaluation Manager)

- An Eclipse plug-in
- Developed using Eclipse PDE facility
- Aim at managing user evaluation from within the IDE (Integrated Development Environment)

The snapshots presented in the next slides are taken from a Technion project that was performed by the student developers: Haimovitch Y., Ben-David D., Pele L., Einav T., Nirenberg B., and Vinikov A.
Defining an Experiment

Experiment name: Music Album
Planned execution time: 11:14:00 AM, 7/14/2008
Experiment status: Configuration

Participating users:

<table>
<thead>
<tr>
<th>Id</th>
<th>First name</th>
<th>Last name</th>
</tr>
</thead>
<tbody>
<tr>
<td>2008809210</td>
<td>Alberto</td>
<td>Valero</td>
</tr>
<tr>
<td>22295300</td>
<td>Ugo</td>
<td>Coisanti</td>
</tr>
<tr>
<td>163121035</td>
<td>Matteo</td>
<td>Di Golia</td>
</tr>
<tr>
<td>539364223</td>
<td>Gabriele</td>
<td>Randelli</td>
</tr>
<tr>
<td>111884222</td>
<td>Fabio</td>
<td>Patrizi</td>
</tr>
</tbody>
</table>

Teammates in charge:

<table>
<thead>
<tr>
<th>Id</th>
<th>First name</th>
<th>Last name</th>
</tr>
</thead>
<tbody>
<tr>
<td>7259327025</td>
<td>Silvia</td>
<td>Bandini</td>
</tr>
<tr>
<td>30210071</td>
<td>Marie</td>
<td>Binding</td>
</tr>
<tr>
<td>440118666</td>
<td>Shel Patih</td>
<td>Hannayoun</td>
</tr>
</tbody>
</table>

Path to project's jar:

C:/Music_Project/Music_Library.jar

Description:
Music Album:
The user creates a new music album for his/her music library. He/She adds songs into this music album from songs already in his/her music library or from his/her personal collection.

Instructions:
Please make sure you understand the task before starting.
Do the task to the best of your understanding.
Please make sure that you already have few songs in your personal collection.

Tasks:
1. Creating Music Library
2. Editing Music Album Details
3. Creating Album Cover Page
4. Burning Album CD
Defining an Experiment
# Viewing the Results

## Results

**Experiment status:** Completed  
**Experiment started:** 13/07/2008 09:23:00  
**Experiment ended:** 13/07/2008 09:26:00  
**5 users have answered out of 5**

<table>
<thead>
<tr>
<th></th>
<th>Strongly Disagree</th>
<th>Disagree</th>
<th>Agree</th>
<th>Strongly Agree</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1. The system tells me what to do at every point.  
2. The system helps me how to do the next task.  
3. The system recovers from any mistake.  
4. The system prevents me to do anything wrong.  
5. It is easy to find out the related help material at any point.  
6. The system tells what it is going to do at every point.  
7. Overall, the design is good.  
8. Overall, it is easy to use the system without any prior training.  
9. Overall, I like the system.
Traceability

- Development tasks are derived from the experiments’ results and assigned to teammates.
- The code developed is associated with the relevant results.
Automatic Measures using Java Aspects

Create a new Automatic Measurement
Clicking Finish will create a new aspect. Thereafter you will proceed to implement the pointcuts.

Action name: Preferences hot keys

Aspect type: Mouse clicks, Key presses, Timer

This aspect is intended to count the number of key presses during the specified action.

The following pointcuts need to be implemented:
- startPoint
- endPoint

AutoMeasurementLibrary 1.0 is not on the build path of this project. Click here to add AutoMeasurementLibrary 1.0 to the build path of this project.
Automatic Measures using Java Aspects

```java
public aspect KeyPressesMeasurePreferencesHotKeys {

    @Override
    /**
     * @return The action name, as will be pre
     */
    public String getActionName() {
        return "Preferences hot keys";
    }

    /**
     * When: BEFORE the point where the counter
     * What: The counter will be started
     */
    public pointcut startPoint(); // TODO Aut

    /**
     * When: AFTER the point where the counter
     * What: The counter will be stopped (mouse
     */
    public pointcut endPoint(); // TODO Auto-

```
Evaluating Lobo using UEMan
Summary

- We present UEMan to bridge the gap between software development and use centered design

Future directions

- Continue the development of UEMan
  - E.g., add statistical analysis to support controlled experiments
- Evaluate UEMan while using it to evaluate other products that are under development