Agile Modeling

From Scott Ambler’s writing on the Ambysoft website
Carl Erickson
Atomic Object LLC
Background

- Scott Ambler is probably the most well-known and prolific thinker in the area of agile modeling (AM).
- Scott has identified core principles of AM.
- These are what I consider to be the most important of the core.
Model With A Purpose

• Who is the audience?
  – Dev team? Customers? Future dev?
• Why are you building it?
  – Selling? Maintenance? Understanding?
• How long will it last?
  – Short-term if fine. How do you preserve it?
Travel Light

• Everything you keep must be maintained
  – Detailed documents, diagrams, requirements
  – And usually violates DRY

• Heavy models take more
  – Time to create
  – Time to maintain
Rapid Feedback

• Shorten time between creation and use
  – Validate soon
  – Move the concept into code
  – Confirm with the customer

• Shared modeling technology helps this
  – So simple tends to be better (whiteboard, cards, post-its)
Assume Simplicity

• Don’t over model
  – What you need to understand today
• Don’t worry about all the details
  – You’re not going to need them
• Don’t expect to get it right the first time
  – Model incrementally
Software Is Your Primary Goal

• Modeling is a means to an end
  – There’s nothing virtuous about models for their own sake

• Turn models into source code
  – As quickly as you can

• Build models that help you make good software
Shared or Inclusive Models

• To satisfy the core principles, models need to be creatable and understandable by all stakeholders (dev, tester, analyst, customer, user, …)

• Creating together is important

“individuals and interaction over process and tools”
Simple Tools

• Whiteboards
• Index cards
• Post-it notes
• Flip charts
• String
• Word processor
• Spreadsheet
Simple Models

• Some common examples…
CRC cards

View
--- Render the Model
--- Transform coordinates.

Controller
--- Model
--- Interpret user input.
--- Distribute control.

Model
--- Maintain problem related info.
--- Broadcast change notification.
Essential Use Cases

**Name:** Enroll in Seminar

**Identifier:** UC 17

**Basic Course of Action:**

- Student provides her name and student number

- Registrar verifies the student is eligible to enroll in seminars. If not eligible then student informed and use case ends.

- Registrar asks student which seminar they'd like to enroll in. If they don't know registrar provides student with course catalog if required.

- Student chooses a seminar or decides not to enroll at all.

- Registrar checks student record to see if student has previously passed prerequisite courses. If not eligible student is asked to choose another.

- Registrar validates the seminar fits into student's schedule.

- Registrar calculates fees

- Student verifies the cost and either indicates she wants to enroll or not.

- Registrar enrolls the student in the seminar and bills them for it.

- The registrar writes enrollment receipt.
Interface
# Story Card

**Story Tag:** DocBookToHTML  
**Release:** Book  
**Priority:** 1  
**Author:** Joanne  
**Date:** 2/21/02  
**Accepted:** 3/17/02

**Description:** Make the DocBook files readable and printable.

**Considerations:** HTML has some drawbacks:  
- Printed version is not production quality.  
- Footnotes can't appear at end of page.

<table>
<thead>
<tr>
<th>Who</th>
<th>Task</th>
<th>Est.</th>
<th>Done</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rob</td>
<td>Simple tags: <code>&lt;chapter&gt;</code>, <code>&lt;title&gt;</code>, <code>&lt;para&gt;</code></td>
<td>1</td>
<td>2/24</td>
</tr>
<tr>
<td>Rob</td>
<td>Asymmetrical tags: <code>&lt;interaction&gt;</code></td>
<td>1</td>
<td>3/3</td>
</tr>
<tr>
<td>Rob</td>
<td>Contextually related tags: <code>&lt;title&gt;</code></td>
<td>1</td>
<td>3/11</td>
</tr>
<tr>
<td>Rob</td>
<td>Stateful output: <code>&lt;footnote&gt;</code></td>
<td>1</td>
<td>3/14</td>
</tr>
<tr>
<td>Joanne</td>
<td>Acceptance Test: Print the first chapter</td>
<td>.1</td>
<td>3/17</td>
</tr>
</tbody>
</table>
Free-form sketches
User interface sketches