Creative Software Engineering

What is it?
Software engineering is traditionally seen as very structured and methodical. However, it often involves creative steps, such as requirements analysis, architecture engineering and GUI design.

Why do we need it?
The author hopes to re-provide technological experiences in physical contexts, broadening access to facilities such as social websites. Methods such as TAPT (below) help us understand user experiences.

Existing Techniques
Software engineers use the following complementary tools to understand how user groups may perceive and use technologies. This helps focus on users, find assumptions, and communicate clearly.

Personas are descriptions of people representative of certain user groups. We might define name, age, gender, job, ethnicity, family, socioeconomic status... Personas are based on real data.

Scenarios show situations in which technology is used. They are stories, with a setting, characters and plot. They help practitioners reflect on designs and find assumptions.

Architectural patterns are reusable solutions to common software design problems. They are abstract templates to be customised and applied to the specific problem at hand, leaving a creative jump between picking up the pattern and the finished solution.

Teasing Apart, Piecing Together (TAPT)
TAPT is a new process that helps software engineers better understand experiences, allowing the re-design of these in new contexts.

Stage one, teasing apart:
1) Briefly describe the functionality
2) Describe the surface elements, related to design
3) Describe experiential effects:
   - physical, emotional, intellectual, social
4) Review the effects, identify those which seem key
5) Use this data to describe the abstract experience in one neutral sentence.

<table>
<thead>
<tr>
<th>Experience (1)</th>
<th>Surface elements (2)</th>
<th>Experienced effects (3 &amp; 4)</th>
<th>Distilled experience (5)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Make a comment</td>
<td>Interface, navigation</td>
<td>Embarrassed by others, feeling dumb</td>
<td>People share updates to stay in touch with each other</td>
</tr>
<tr>
<td>Friends online</td>
<td>Visual elements, layout</td>
<td>In touch with friends, happy</td>
<td></td>
</tr>
<tr>
<td>Messages</td>
<td>Interaction, sound</td>
<td>Connected with friends, happy</td>
<td></td>
</tr>
<tr>
<td>Conversations</td>
<td>Text, images, videos</td>
<td>Social, emotional</td>
<td></td>
</tr>
</tbody>
</table>

Remember to note key effects!

Stage two, piecing together:
1) Brainstorm on how the distilled experience might manifest in the new context
2) Build a scenario
3) Check the reconstruction: did you capture everything?
   Are there unintended key effects? Adjust accordingly.

<table>
<thead>
<tr>
<th>Distilled experience</th>
<th>Piecing together social status updates</th>
</tr>
</thead>
<tbody>
<tr>
<td>People share updates to stay in touch with each other</td>
<td>Social Status Updates</td>
</tr>
<tr>
<td>Handheld Devices (Individual)</td>
<td></td>
</tr>
</tbody>
</table>

Name: Model-view-controller
Problem: Applications can become cumbersome and complex; splitting them into separate layers can lead to better implementation.
Solution: A three-part model of the following structure...

Student: Clare Hooper (clare@ecs.soton.ac.uk)
Advisor: David E. Millard
University or work place: University of Southampton, UK