End User Development: Approaches Towards A Flexible Software Design

Michael Spahn, Christian Dörner, Volker Wulf

Track: Design Theory, Research & Practice in Information Systems

ECIS 2008, 9th – 11th June 2008, Galway, Ireland
Questions to Answer

- Why End User Development?
- What is End User Development?
  - Definition
  - Classification
  - Approaches
- How to use it?
Why End User Development?

- Designing flexible software seems to be an unreachable goal
- Why should we make it even more complex, by involving end users?

Design for change leads to cost reduction
The long tail of software development

What is End User Development?

Lieberman et al.: “End-User Development can be defined as a set of methods, techniques, and tools that allow users of software systems, who are acting as non-professional software developers, at some point to create, modify or extend a software artefact.”

EUD seems to be everything, except the usage of software

Imagine, I am a Software Engineer, who wants to build a configurable, flexible system by using EUD techniques. What do I need to know?

Overview of existing EUD approaches
A guideline, how to select appropriate approaches

What is End User Development?

In contradiction to other overviews, like the one of Germonprez, our classification differentiates the approaches by the two dimensions:

- **Complexity**: Technical knowledge, which is required to apply the principle
- **Adaptation power**: The level or extend of adaptations that can be realised

<table>
<thead>
<tr>
<th>Complexity</th>
<th>EUD Approaches</th>
<th>Supportive EUD Approaches</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adaptation Power</td>
<td>Customisation</td>
<td>Integration</td>
</tr>
<tr>
<td>Programmers</td>
<td>Programming</td>
<td>Component swapping at runtime; Separated tailoring interfaces</td>
</tr>
<tr>
<td>Local Developers</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Non-Programmers</td>
<td>Interface customisation; Parameterisation</td>
<td>Programming by demonstration (PBD); Accountants paradigm; Integrated tailoring interfaces</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Community aspects; Configuration files</td>
</tr>
</tbody>
</table>

EUD Approaches - Customization

Interface Customisation

Parameterisation
EUD Approaches - Integration

Programming by Demonstration


EUD Approaches - Integration

- Accountants paradigm
  - Well-known depiction of data in a table
  - Programming by
    - Formulas
    - Relations between cells / tables
    - Logical constructs

=EXP(5)

=A1+B1+Tabelle2!A1

=WENN(A1>B1;"bigger five";"smaller five")
EUD Approaches - Integration

Component swapping at runtime

EUD Approaches - Extension

Natural Programming

Supportive EUD Approaches

WYSIWYT

Question-based testing


How to apply EUD? - A Guideline?

- First, make sure that you have sufficient information about:
  - The users (e.g. who are they, how they are educated)
  - The domain / field (e.g. which kind of organisation is it)
  - The context (e.g. work processes, used information systems)
How to use it? - A Guideline?

- Second, fill the created complexity distribution with EUD approaches
- Keep these recommendations in mind
  - EUD environment of your system should be as natural as possible
  - The complexity slope should be gentle, if you want to address more than one user group
  - If users should be able to create and integrate extensions, alternative programming approaches, like programming by example, are required
  - To get users started quickly, runtime and design-time should be similar
  - The environment should provide results instantly
  - The environment should offer support mechanisms
    - Debugging should be natural (users could ask why...?, why didn't't...? questions)
    - Debugging tools should help to correct errors
    - Save exploration of adaptations should be possible
    - The user community should be supported (e.g. by exchangeable configuration files)
Conclusion

- EUD can help to increase the flexibility of a software, as it supports the “long tail” of end user developers
- IS developers should think how they can address the different end user developers of their system
- There are many EUD approaches, which may be suitable
- Our guideline can help to select the “right” ones in a structured way

Future work
- The guideline has so far not been evaluated in practice
- We work on prototypes with a “gentle complexity distribution”, which combines several EUD approaches to support different user groups
Contact

Christian Dörner
Research Associate

University of Siegen
F: +49 - 271 740 40 70
M: christian.doerner@uni-siegen.de

http://www.wineme.uni-siegen.de/

http://www.enduserdevelopment.org