End-User Software Engineering and Distributed Cognition

Margaret Burnett, Christopher Bogart, Jill Cao, Valentina Grigoreanu, Todd Kulesza, Joseph Lawrance

Oregon State University

Project Director, EUSES Consortium
http://eusesconsortium.org/
Toward “More Discipline”

Premise: EUPs won’t become “more disciplined” UNLESS:

- Perceived payoff (to their priorities) is obvious,
- or
- Discipline is so low in cost, don’t have to worry about cost.
What is Distributed Cognition?

Defn: “cognition beyond the individual to encompass interactions between people, resources, materials, environment”

When one user-one computer:

⇒ user does some cognition,

system does some cognition.
So What?

Instead of:
What can we figure out automatically?

Problem statement becomes:
How can we help the end user think?
Rest of This Talk: Concrete

Past Work/New Work Examples:

WYSIWYT and Surprise-Explain-Reward

Rothermel/Burnett and many students
WYSIWYT Testing
(What You See Is What You Test)

For end users & spreadsheets, what is testing?

Test: A decision if some output is right given the input.

Test Adequacy: Have “enough” tests been performed?
Attention Investment (Blackwell)

Usual SE view:

“If we build it, they will come”

Attn. Investment: why should they?

PERCEIVED Costs of new feature:
   Learning it, doing it.

PERCEIVED Benefit of new feature?
   Not clear without paying the costs.

PERCEIVED Risks:
   Eg: wasted cost (time), getting environment into an odd state from which can’t easily recover, ...
Our Strategy: 
Surprise-Explain-Reward (Burnett)

Surprise: show presence of information gap. 
   Basis: curiosity theory. 
   Negotiated interruption: user decides.

Explain: users seek explanation to close gap. 
   Self-directed learning. At user’s initiative. 
   Basis: education theory. 
   Suggest actions we want them to do &…

... the Rewards: 
   $ Basis: attention investment. 
   Quality benefits. (Plus affective rewards.)
√: User Notices a Correct Value...

Cell turns more blue (more “tested”).

Testing also flows upstream, Coloring other affected cells too.

If this value is right, √ it; if it’s wrong, X it. This testing helps you find errors.

At any time, user can check off correct value.
WYSIWYT and Distributed Cognition

Colors: External memory of things-to-test. Promotes discipline in testing in that:
- Things-to-test based on formal test adequacy criterion.
- User can’t forget how much progress has made on that things-to-test list.

Summary of distribution of cognition:
- System: tracks progress on “things-to-test”.
- User: makes value correctness judgments.
X: User Notices an Incorrect Value...

And this one’s wrong too.

User notices that this value is wrong and “X”s it out.

The formula error is in one of these red cells -- probably the darkest.
Fault Loc. and Distributed Cognition

System:
2. Finds possible culprits, prioritizes user attention with colorings.

User:
1. User makes value correctness judgments.
3. User systematically follows priorities in making formula correctness judgments.
Debugging Learned Programs: (An Emerging New Project)

Kulesza/Wong/Stumpf/Burnett/Ko
Debugging Learned Programs (after the ML specialist goes home)

Many machine learning programs adapt to an end user’s behavior

Eg. Email classifier

```java
public class MyFrame extends JFrame {
    private AsciiFrameManager reader;

    public MyFrame(String filename) throws Exception {
        reader = new AsciiFrameManager(filename);
        ...}
```
Problem

The machine learned program is never 100% accurate

Based on your interest in:

We recommend:

End user is the only one present to “fix” this
Old Solution (Discipline-free)

User must label new examples.
No way to plan: User doesn’t know how many to do.
No way to judge: User doesn’t know effects on other examples.
Luck: need appropriate examples to arrive.

We recommend:

<table>
<thead>
<tr>
<th>Film</th>
<th>Outcome</th>
</tr>
</thead>
<tbody>
<tr>
<td>The Texas Chain Saw Massacre</td>
<td>No</td>
</tr>
<tr>
<td>Friday the 13th</td>
<td>No</td>
</tr>
<tr>
<td>Nightmare</td>
<td>No</td>
</tr>
<tr>
<td>Aladdin</td>
<td>Yes</td>
</tr>
</tbody>
</table>
New Solution: WhyLine for Email

Let the end user fix the logic directly.

1. Allow end-users to ask questions of machine-learned programs.
2. Explanations represent “code” (not just outputs):
   Provide details about the classifier’s logic.
3. Explanations are modifiable:
   Allow users to fix the logic.
Why will this message be filed to "Systems"?

- Ask why will this message be filed to "Systems"?
- Ask why won't this message be filed to...?
- Why won't this message be filed to "Enron News"?
- The message will be filed to "Systems" instead of "Enron News" because Systems rates more words in this message near Required than Enron News does, and it rates more words that aren't present in this message near Forbidden.
- When choosing where to file a message, the system examines how required or forbidden.
- If all 20 of the messages in Systems were still in the inbox, they would now all be automatically filed correctly.
- If all 20 of the messages in Enron News were still in the inbox, 16 of them would now be automatically filed correctly.

Bar graph (source code):
End-User Debugging of ML and Distributed Cognition

System:

2. Adds answers to the questions.
4. Apprises user of effects of these corrections.

User:

1. Asks good questions.
3. Adds corrections to the answers.
Debugging via Information Foraging

Lawrance/Burnett/Bellamy/Bogart
Information foraging theory in debugging

Adapted from optimal foraging theory.
How to seek, gather, and use information.

Concepts:

- Information scent.
- Information diet.
- Information patches.
- And...topology among patches.
public static void main(String[] args) {
    startupProcess(args);
    new RSSOwlLoader();
}

private static void startupProcess(String[] args) {
    createWorkingDir();
    createArchiveDir();
    createCacheDir();
    createLogDir();
    createTempDir();
    setFilePathes();
    setUpProperties();
}

private RSSOwlLoader() {
    Display.setAppName("RSSOwl");
    display = new Display();
    invisibleShell = new Shell(display);
    new GUI(display, invisibleShell).showGui();
}
Empirical Study (prof. programmers): PFIS predict navigation?

12 IBM programmers.
Harvested from RSSOwl (a popular Open Source news reader):
  Fix a bug.
  Develop a missing feature.
Empirical Study of End-User Programmers Debugging Excel

Highest activity for all 4: Foraging!
What Information Foraging Theory Says about Distributed Cognition

1. Environment provides cognition in the way of signposts (cues) pointing the way.

2. User (programmer) follows the scent of these cues.

3. User can add to the cues through enrichment (notes) to make the scent stronger.

4. System can help the user go faster by providing topological links that reduce the cost of following scent.

   eg: Visual Studio’s “javadocs” with a hover.
Conclusion:

\[ \text{EUSE} = \text{EUP} + \text{rest of lifecycle} + \text{systematicness} \rightarrow \text{quality.} \]

A distributed cognition view of how to do this:

Don’t base tools on “what can we figure out”.

Do base tools on helping the user think.

Keeping in mind perceived costs/benefits