# Programmers Are Users Too: Human-Centered Methods for Improving Programming Tools

Brad A. Myers, Andrew J. Ko, Thomas D. LaToza, YoungSeok Yoon

IEEE Computer, July 2016

Summary by Prof. Thomas LaToza

SWE 795, Spring 2017

Software Engineering Environments

# Programmers are user too

- Key idea:
  - Developers are users.
  - HCI methods can be applied to developers.
  - Methods may be applied to tools, APIs, libraries, documentation, programming language design
- Important for tools to be useful and address an important problem
  - Should be high frequency or have a large impact
  - Tools that do not address an important problem may not be adopted

| Method | Tool development activities supported | Key benefits | Challenges and limitations |
|---|---|---|---|
| Contextual inquiry | Requirements and problem analysis | » Experimenters gain insight into day-to-day activities and challenges.<br>» Experimenters gain high-quality data on the developer's intent. | » Contextual inquiry is time consuming.<br>» Recruiting professionals might be a challenge. |
| Exploratory lab studies | Requirements and problem analysis | » Focusing on the activity of interest is easier.<br>» Experimenters can compare participants doing the same tasks.<br>» Experimenters gain data on the developer's intent. | The experimental setting might differ from the real-world context. |
| Surveys | » Requirements and problem analysis<br>» Evaluation and testing | » Surveys provide quantitative data.<br>» There are many participants.<br>» Surveys are (relatively) fast. | The data is self-reported and is subject to bias and lack of participant awareness. |
| Data mining (including corpus studies and log analysis) | » Requirements and problem analysis<br>» Evaluation and testing | » Data mining provides large quantities of data.<br>» Experimenters can see patterns that emerge only with large corpuses. | » Inferring or reconstructing the developer's intent is difficult.<br>» Data mining requires careful filtering. |
| Natural-programming elicitation | » Requirements and problem analysis<br>» Design | Experimenters gain insight into developer expectations. | The experimental setting might differ from the real-world context. |

| | | | |
|---|---|---|---|
| Rapid prototyping | Design | Experimenters can gather feedback at low cost before committing to high-cost development. | Rapid prototyping has lower fidelity than the final tool, limiting what problems might be revealed. |
| Heuristic evaluations | » Requirements and problem analysis<br>» Design<br>» Evaluation and testing | » Evaluations are fast.<br>» They do not require participants. | Evaluations reveal only some types of usability issues. |
| Cognitive walkthroughs | » Design<br>» Evaluation and testing | » Walkthroughs are fast.<br>» They do not require participants. | Walkthroughs reveal only some types of usability issues. |
| Think-aloud usability evaluations | » Requirements and problem analysis<br>» Design<br>» Evaluation and testing | Evaluations reveal usability problems and the developer's intent. | » The experimental setting might differ from the real-world context.<br>» Evaluations require appropriate participants.<br>» Task design is difficult. |
| A/B testing | Evaluation and testing | » Testing provides direct evidence that a new tool or technique benefits developers.<br>» It provides objective numbers. | » The experimental setting might differ from the real-world context.<br>» Testing requires appropriate participants.<br>» Task design is difficult. |

# Design recommendations

- Good aesthetic and interaction design
  - Better interaction design leads to better tools
- Primacy of viewing code
  - Visualizations help as comprehension and navigation aids, but want to see code
- Importance of search
  - Developers must work with vast sets of artifacts, but have specific questions. Search can help directly express.
- Augmenting what developers do
  - Developers have specific strategies. Can integrate directly into these strategies.

# Questions for discussion

- Overall reactions
- Where might this method have the most impact?
- Where might this method be hard to use?
- You are trying to understand what causes defects. What method(s) might you use?
- You are trying to understand if developers are more productive writing web apps in PHP or in React. What methods might you use?