A Small Matter of Programming

Software engineering is a highly complex task, a microcosm of many challenges in HCI.

Making software engineering more accessible could empower millions to customize applications and write programs.
Today

Understand the challenges in programming
Design more effective software engineering interfaces
Aid novices in learning to program or writing programs
Abstract best practices into toolkits
Threshold/Ceiling Diagram

[Myers, Hudson and Pausch, TOCHI 2000]

Threshold: Difficulty of use

Ceiling: Sophistication of what can be created

C++
Web
Figma

To go after your own projects: are you trying to lower the threshold, or raise the ceiling?
Understanding programming practice
Information Needs in Programming

[Ko, DeLine and Venolia, ICSE '07]

Observed 17 developers in 90-minute sessions, transcribed all activities, and performed thematic coding of information needs

- Writing code: e.g., how do I use this method?
- Submitting a change: e.g., which files are included?
- Triaging bugs: e.g., is the problem worth fixing?
- Reproducing failure: e.g., what are failure conditions?
- Understanding execution: e.g., what caused this behavior?
- Design: e.g., why is the code implemented this way?
- Awareness: e.g., what are my collaborators working on?

Which one do you think was most common?

[1 min]
Designing effective APIs
[Robillard and DeLine, Empir. Software Engineering ’11; Myers and Stylos 2016]

Survey and in-person interviews, combined reaching 440 professional software engineers

Biggest challenge: inadequate documentation, e.g., how it was intended to be used

“Nowhere in there does it say, and we intended to be used for a few graphics of small size because the memory footprint is going to be this.”

Proposal: follow human-centered design methods for APIs

Elicititation methods (how do programmers think about this functionality?), cognitive dimensions of notation, Nielsen’s heuristics
Web foraging and programming

[Brandt et al., CHI ’09]

This paper articulated how programmers make heavy use of the web. Laboratory study: ask programmers to implement a chat room in PHP.

Average participant spent 19% of their programming time on the web

- JIT learning of new skills
- Clarifying existing skills
- Reminding themselves of details
Characterizing API usage

[Glassman et al. 2018]
Software engineering
interfaces
Asking ‘why’ questions of code

[Ko and Myers CHI ’04, ICSE ’09]

Debugging problems often reduce to “why” questions. Analyze program traces to answer many unanswered “why” and “why not” questions about what just happened.
Highlighting inactive CSS
[Lim et al. UIST 2018]

Visual regression testing: does toggling a CSS property influence the object of interest? If not, flag the property as inactive.

1. INITIAL STATE

2. DISABLE EFFECTIVE PROPERTY => REGRESSION

3. DISABLE INEFFECTIVE PROPERTY => NO REGRESSION

4. FINAL STATE: ALL INEFFECTIVE PROPERTIES PRUNED

Inactive properties (in red) have no effect on the the page when disabled.

Firefox 70 recently released a CSS inspector feature with the same core idea.
Languages that learn from crowds

[Fast and Bernstein, UIST ’16]

If your functions sent back information to a central community server, could they…

Recover from crashes?
Auto-optimize?
Test themselves?

Count the vowels in a string

```python
import re

@meta(parent="5700375c2f6a2f000330436a")
def count_vowels(s):
    return len(re.findall('[aeiou]', s, flags=re.I))
```

Warning: Meta has found a possible alternative that is \textbf{1.3} times faster

Example inputs:

- `count_vowels("UIST")` \implies 2
- `count_vowels("CHI")` \implies 1

Known errors:

- `count_vowels(['CHI', 'UIST'])` \implies expected string or bytes-like object

You can load this snippet with:

```python
count_vowels = meta.load("http://www.meta-lang.org/snippets/5700375c2f6a2f000330436a")
```
Learning programming
Logo: programming for children [Papert '93]

Constructionist learning: learning happens most effectively when people are making tangible objects.

Lego Mindstorms followed this mold and was named after it.
Recall: Scratch
[Resnick et al. CACM 2009]

Social: upload and remix others’ creations

All programming has been done online. This data has led to many papers on understanding notions of authorship and creative remixing.
Online python tutor

[Guo, SIGCSE '13]

Embeddable Python data structure visualization

Over 200,000 users and a dozen universities using it

```python
1  def listSum(numbers):
2      if not numbers:
3          return 0
4      else:
5          (f, rest) = numbers
6          return f + listSum(rest)
7
8  myList = (1, (2, (3, None)))
9  total = listSum(myList)
```
Watch many learners code and debug in real time
Clustering student programs

[Glassman and Miller 2015]
Programming by demonstration
Goals of PBD

Programming by demonstration (PBD): teach a computer a program by doing it yourself while it watches

Challenges

- There is an infinite, and hugely branching, space of programs that might be inferred
- Inferred macros can be extremely brittle
EAGER

[Cypher, CHI '91]

Infer a macro by watching the user's behavior
Excel flash fill

[Gulwani, PoPL 2011]

Develop a DSL of string transformations, and learn from examples how to decompose the string transformation into subproblems.

Machine learning ranks between all possible valid programs.
Toolkits
Research agenda: toolkits

Crystallize and formalize a perspective on a difficult engineering problem. What representation makes solving this problem easier?

If successful, toolkits shift the programming practice for the area
D3.js [Bostock and Heer 2011]

Previous representation: visualization as custom scene graph abstractions (e.g., HTML Canvas)

D3: use a DOM-style representation
Sikuli: programming with screenshots

[Yeh, Chang, and Miller, UIST ’09]

Visual template search in desktop scripting
Discussion

Find today’s discussion room at http://hci.st/room