CSE 190 / 291P: Usability of Programming Languages ("Programmers Are People Too")

Michael Coblenz

About Me

- Spent eight years as a software engineer at Apple
- But: no one knows how to design programming languages that are known to benefit people!
- Can we develop a *science* of PL design?
- Completed a PhD on PL usability at Carnegie Mellon (designed a new smart contract language; showed it benefits people)
- Postdoc at UMD

Learning Goals

- After successfully completing this course you will be able to:
  - Apply qualitative and quantitative research methods to obtain insights about programming language design choices;
  - Critically analyze design questions in the context of languages and programming systems;
  - Read and interpret research papers in the area of usability of programming languages, and summarize major findings to date.

Course Content

- Three categories of material: methods, history, results

Grading

Assignment	Weight	Due Date
Project	50%	Various
HOPL paper summary/presentation	5%	2/6
Reading responses	20%*	Before lecture
Methods Exam	20%	2/2 in class

* We'll drop the lowest two reading response grades.

Project Milestones

- Pitch (1/12)
- Team matching (1/14)
- Literature review (1/26)
- Meet with instructor (1/30)
- Proposal (2/4)
- Checkpoint (2/20)
- Report (3/13)
- Presentation (3/20)

On Reading Papers

	Textbook chapter	Academic paper
Reading order	Start at the beginning, finish at the end	Read the whole thing, but use random access to enhance understanding and focus your reading
Comprehension expected	Everything (could be on a test)	Main ideas; answer your own questions; identify areas for future study
Comprehension strategies when confused	Read all previous chapters first	Write down questions in the margins, continue reading, and re-visit questions after seeing more of the paper
Schedule	Read night before class, or after class, or not at all (instructor will present)	Read in advance and come to office hours with questions

About Class

- Discussion is an integral part of class!
  - Research has shown that *active learning* is superior
- We will have regular in-class activities (ICAs)
- Your final grade will be no higher than your ICA grade + 15%
  - To pass the class, you must participate in at least 65% of ICAs

TA: Ayla Kurdak

Logistics

- My office hours: Mondays after class (no OH on 1/19 and 2/16)

Welcome To Undergrads

- Undergrads belong in research too!
- 190 and 291 are the *same class*
- This class may require more *independence* than you are accustomed to
- Papers can draw on the *entirety of human knowledge*; do not expect to understand everything
- Please come to me with questions!

Resources

- Come to office hours!
  - Even just to say hello
- *Research Methods in Human-Computer Interaction* online (library)
- You should meet with me at least once to discuss your project.
  - You can make an appointment outside office hours if needed.

Collaboration

- Collaboration is encouraged! But *write your own answers*.
  - You may not use AI assistants to write your reading responses.
- Whiteboard discussion policy: discuss at a whiteboard, erase it, and write your own answers
  - Applies to ChatGPT, etc. as well.

Technology Policy

- Screens are distracting to others [1] and decrease learning outcomes
- Laptops, tablets, and phones may not be used in class except for note-taking, emergencies, and for disability accommodations

[1] https://blueprintcdn.com/wp-content/uploads/2016/05/Blueprint-Research-2016-Carter-Greenberg-and-Walker.pdf

TODAY

- Introduction to human-computer interaction, software engineering, and programming languages research

WHAT IS RESEARCH?

- Pursuit of *new* knowledge
- Contrast with other usages, e.g. "do some research at the library" — our focus here is on *new discoveries*

RESEARCH QUESTIONS

- Every research endeavor seeks to address research questions.
  - The only way there are no questions is if everything is known — and then you don't need to do any research!
- But what makes a good research question?

RESEARCH QUESTION CRITERIA

- *Novel*: we don't already have a satisfactory answer
- *Interesting:* we care about the answer
- Relevant: there will be impact from answering it
- Feasible: Not too broad, not too narrow

TYPES OF SOFTWARE ENGINEERING RESEARCH QUESTIONS

- Per Mary Shaw (2003):

Table 1. Types of software engineering research questions	
Type of question	**Examples**
Method or means of development	How can we do/create/modify/evolve (or automate doing) X? What is a better way to do/create/modify/evolve X?
Method for analysis or evaluation	How can I evaluate the quality/correctness of X? How do I choose between X and Y?
Design, evaluation, or analysis of a particular instance	How good is Y? What is property X of artifact/method Y? What is a (better) design, implementation, maintenance, or adaptation for application X? How does X compare to Y? What is the current state of X / practice of Y?
Generalization or characterization	Given X, what will Y (necessarily) be? What, exactly, do we mean by X? What are its important characteristics? What is a good formal/empirical model for X? What are the varieties of X, how are they related?
Feasibility study or exploration	Does X even exist, and if so what is it like? Is it possible to accomplish X at all?

ICSE 2002

- From Shaw 2003:

Table 2. Types of research questions represented in ICSE 2002 submissions and acceptances

Type of question	Submitted	Accepted	Ratio Acc/Sub
Method or means of development	142(48%)	18 (42%)	(13%)
Method for analysis or evaluation	95 (32%)	19 (44%)	(20%)
Design, evaluation, or analysis of a particular instance	43 (14%)	5 (12%)	(12%)
Generalization or characterization	18 (6%)	1 (2%)	(6%)
Feasibility study or exploration	0 (0%)	0 (0 %)	(0%)
TOTAL	298(100.0%)	43 (100.0%)	(14%)

IMPERFECT EXAMPLES

- Which makes web developers more productive: TypeScript or JavaScript?
  - By how much?
  - In what contexts?
- What aspects of TypeScript make it more effective than JavaScript for web developers?
  - But what tasks are they doing?
- Does adding typestate to REST API specifications reduce the rate of bugs?

VALIDITY

- *External validity*: to what extent do your results generalize to other situations?
  - Usual threats: limited sample of participants, short task duration, artificial task design
- *Internal validity*: to what extent are you measuring the thing you claim to measure?
  - Usual threats: experimental design mistakes
  - Suppose: AM participants get condition A, PM participants get condition B.
  - Maybe any effect is due to time of day, not due to condition.

METHODS

- The first three weeks will be about *research methods*
  - These enable answering some kinds of research questions!
- Today: summary of different kinds of research questions and corresponding methods

QUANTITATIVE RESEARCH

- Answering some questions requires *statistical analysis*
  - Do people fix bugs faster if they drink more coffee?
  - Does C++ lead to more bugs than Rust?
  - Do strong type systems inhibit or promote iteration?
- Answers can be very clear, but lack *external validity*

QUALITATIVE RESEARCH

- Often answers "why?" "how?" or "what?" questions
- What kinds of bugs do spreadsheet users insert?
- Why do newcomers often stop contributing to open-source projects?
- How do programmers decide when to write functions (vs. copy/pasting)?
- Provides depth, explains *why* (quantitative studies do not)
- Does not provide convincing evidence of causation

MIXED METHODS

- Combining qualitative and quantitative approaches is often best
- Example:
  - Interview developers to see why they chose to stay or leave OSS projects
  - Also gather demographic data (age, gender, years of experience, employment status) and do hypothesis tests (statistics)
  - Fabricated result: "Developers often reported leaving when others made insulting comments in their PRs; there was no significant effect of age, gender, experience, or employment."

OBSERVATION VS. EXPERIMENT

- A "study" involves gathering and analyzing data to answer a research question.
- **Observational** studies gather data about what *is*.
  - Example: *interview* engineers to find out what factors influence their API design decisions.
- **Experiments** manipulate what *is* to observe outcomes.
  - Example: *enable* or *disable* autocomplete in an IDE; measure time required to complete tasks.

YOUR TURN

- Identify a usability question YOU have about a COMMON PL.
  - With a partner.
  - Share afterward.

"The extent to which a product can be used by *specified users* to achieve *specified goals* with *effectiveness*, *efficiency* and *satisfaction* in a specified *context of use*."

RESEARCH QUESTIONS AND METHODS

- How can I understand this situation better? (*ethnography, contextual inquiry)*
- I think P is a problem. What fraction of people in a population want it fixed? (*survey)*
- I designed a tool. What challenges do people face when doing task X with my tool? (*usability study)*
- Can most people use my tool successfully to do task T? (*usability study* + *quantitative analysis*)
- Is my tool *better* than an existing one for task T? (*quantitative study: randomized controlled trial*)

ARGUMENTS

- Which do you find convincing?
- What form *should* evidence take?

READING ASSIGNMENT

- The article introduces the "language wars"
- Read it *critically*
- I don't agree with everything in there
- But it's key contextual information for the rest of the course

NEXT TIME

- Discuss the paper
- How to run a usability study