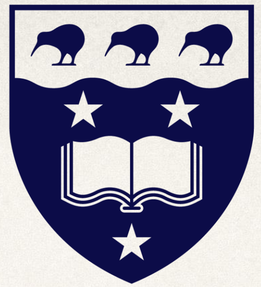


Waipapa Taumata Rau
University of Auckland

Finding Your Fit

How Your Thinking Style Shapes Your Software Engineering Experience

Associate Professor Kelly Blincoe, University of Auckland

BRIDGES Summer School in the South Pacific, 9 January 2026

About Me

Waipapa Taumata Rau
University of Auckland

- Associate Professor of Software Engineering at the University of Auckland in New Zealand
- Chair of Software Innovation New Zealand
- Member-at-large of ACM SIGOSFT
- Rutherford Discovery Fellow
- Research topics: human and social aspects of software engineering, software dependencies and ecosystems, diversity and inclusion in software engineering, inclusive software

Career journey

- 2004: Bachelor of Engineering
- 2004-2012: Software Engineer
- 2009-2014: PhD, Drexel University
- 2014-2015: Postdoctoral fellowship, University of Victoria
- 2015-now: Academic

Waipapa Taumata Rau
University of Auckland

Waipapa
Taumata Rau
University of Auckland

Image source: https://www.pngitem.com/download/oihJow_world-map-outline-png-transparent-png/

Tech in Aotearoa New Zealand

- Tech sector is one of the country's largest and fastest-growing industries
- In 2024:
  - contributed $23.8 billion to GDP (8% of the economy)
  - employed more than 119,000 people (4.8% of the workforce)
  - generated $11.4 billion in exports (New Zealand's third-largest export earner after dairy and tourism)

Source: https://technewzealand.org.nz/nzs-tech-sector/

Gender diversity problem

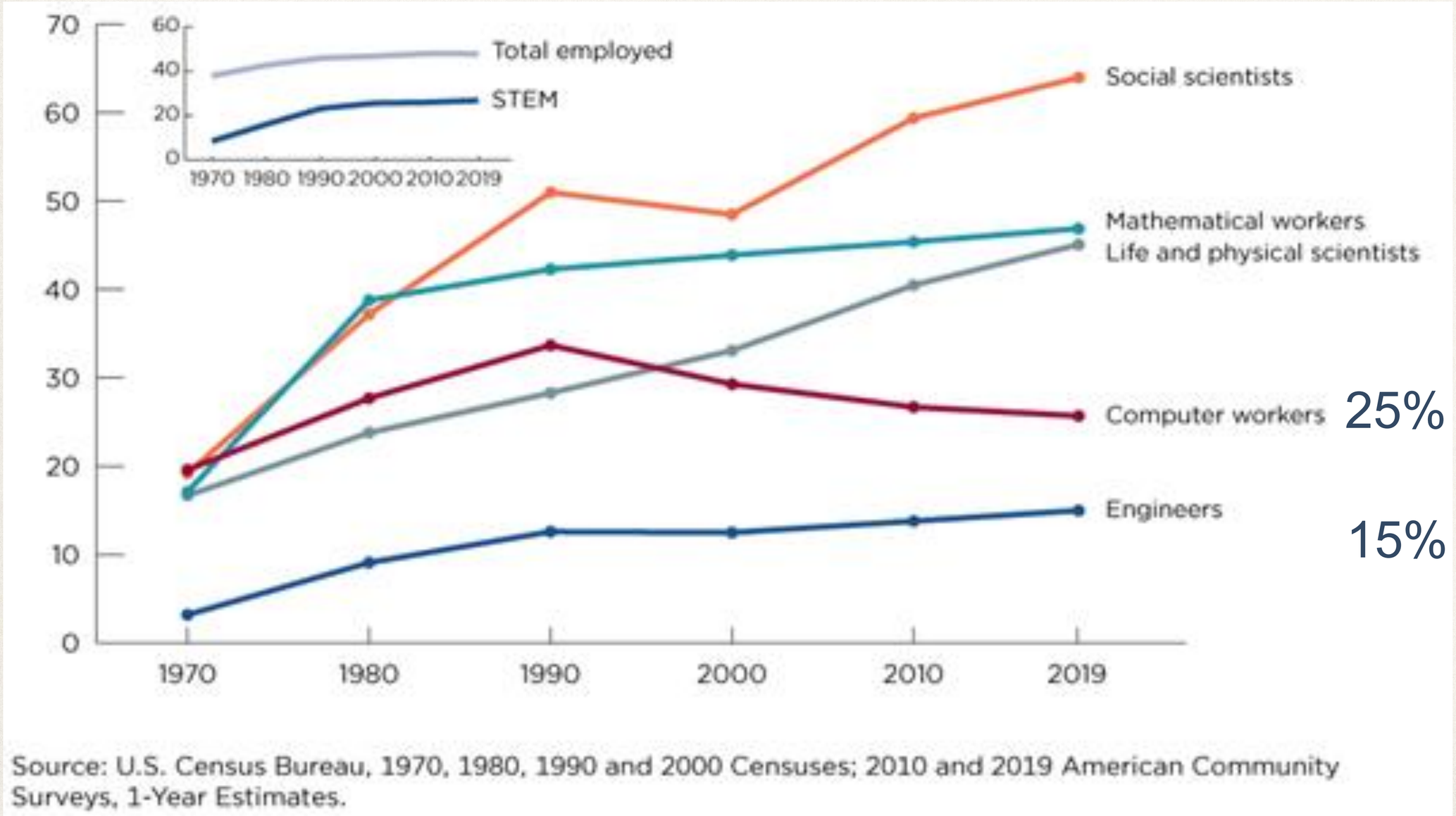
Tech

only 23% women

Engineering

only 14% women

Women in STEM jobs (USA)



Source: U.S. Census Bureau, 1970, 1980, 1990 and 2000 Censuses; 2010 and 2019 American Community Surveys, 1-Year Estimates.

Image source: https://ssti.org/blog/tech-industry-booming-women%E2%80%99s-participation-continues-lag

Engineering Sector Diversity (NZ 2024)

	NZ Population	Workforce	Senior leadership
Māori	18%	3.2%	3.2%
Pacific Peoples	9%	1.9%	0.9%
LGBTQIA+	5%	3.0%	3.3%
Disability	25%	2.0%	2.0%
Neurodiverse	20%	4.6%	3.8%

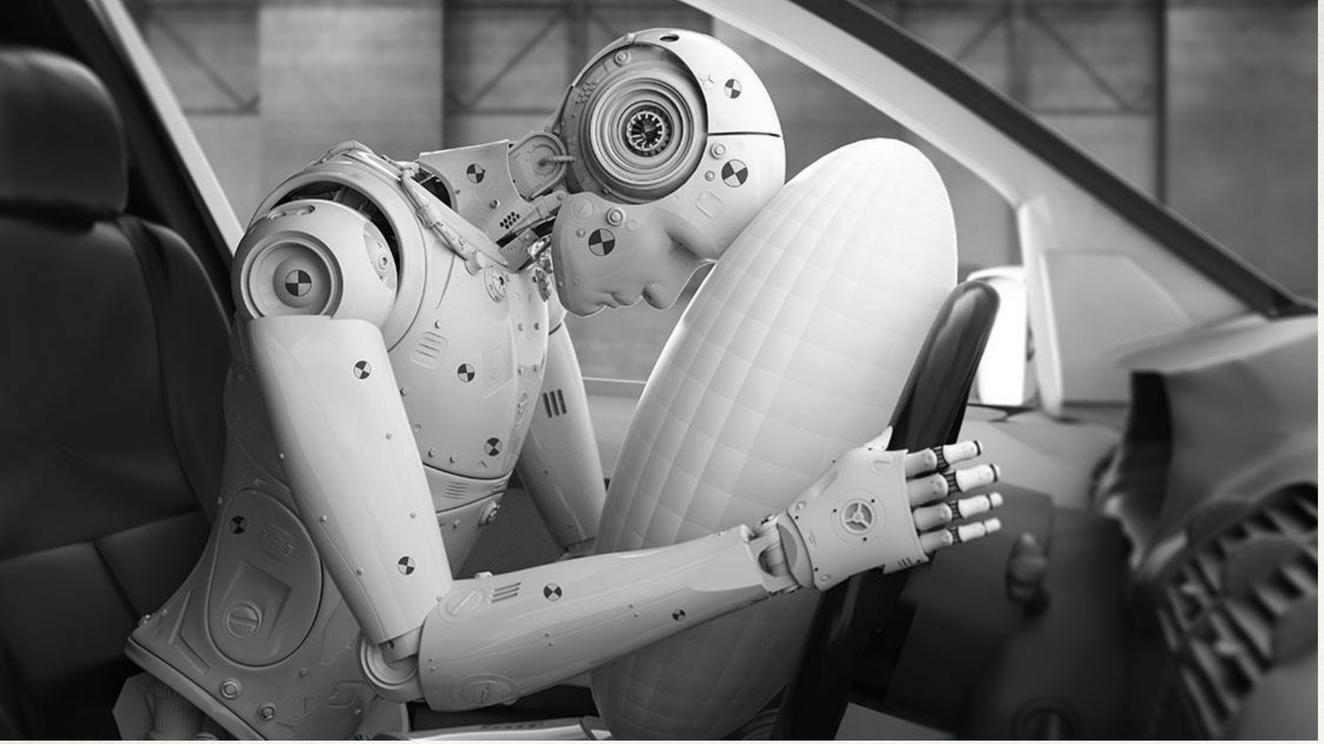
Source: Diversity Agenda 2024 report. Accessed from https://diversityagenda.org/wp-content/uploads/2024/06/Diversity_Agenda_2024_insights.pdf (August 2024)

Benefits of diversity

- Improved productivity
- Increased innovation
- More usable products

Vasilescu et al., CHI 2015; Østergaard et al., Research Policy 2011; Burnett et al., CHI 2016

Image: http://www.theinclusionsolution.me/what-is-diversity-part-2-diversity-of-thought/



First name:

François

Your first name must have at least two letters and no unusual characters

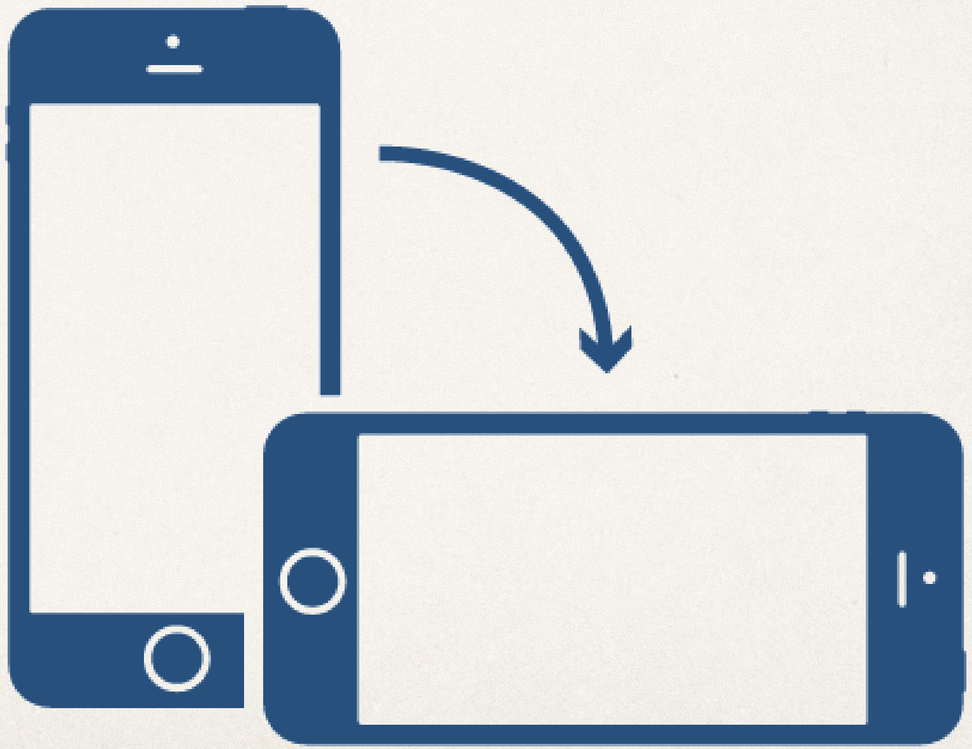


Image source: https://www.safeatworkca.com/safety-articles/air-bag-safety/

Image: The Noun Project; Created by Kelig Le Luron

Cognitive style – five facets

Motivation	Why using the software. Task Completion vs. trying out new features.
Self-Efficacy	Confidence using the software. Blame self vs. blame tool.
Information Processing	How information is gathered. Comprehensive vs. selective.
Learning Style	How new features are learned. Process orientated vs. tinkering.
Risk Attitude	Willingness to try unknown features. Risk-averse vs. risk-taker.

M. Burnett, S. Stumpf, J. Macbeth, S. Makri, L. Beckwith, I. Kwan, A. Peters, W. Jernigan, Gendermag: A method for evaluating software's gender inclusiveness, Interacting with Computers 28 (6), (2016) 760–787

Cognitive style – five facets

Motivation

For task completion — To learn new features

Self-Efficacy

Lower, blames self — Higher, blames tech

Information Processing

Comprehensive — Selective

Learning Style

Process oriented — Tinkering

Risk Attitude

Risk-averse — Risk-taker

M. Burnett, S. Stumpf, J. Macbeth, S. Makri, L. Beckwith, I. Kwan, A. Peters, W. Jernigan, Gendermag: A method for evaluating software's gender inclusiveness, Interacting with Computers 28 (6), (2016) 760–787

What is your style?

Cognitive style – five facets

Motivation

For task completion — To learn new features

Self-Efficacy

Lower, blames self — Higher, blames tech

Information Processing

Comprehensive — Selective

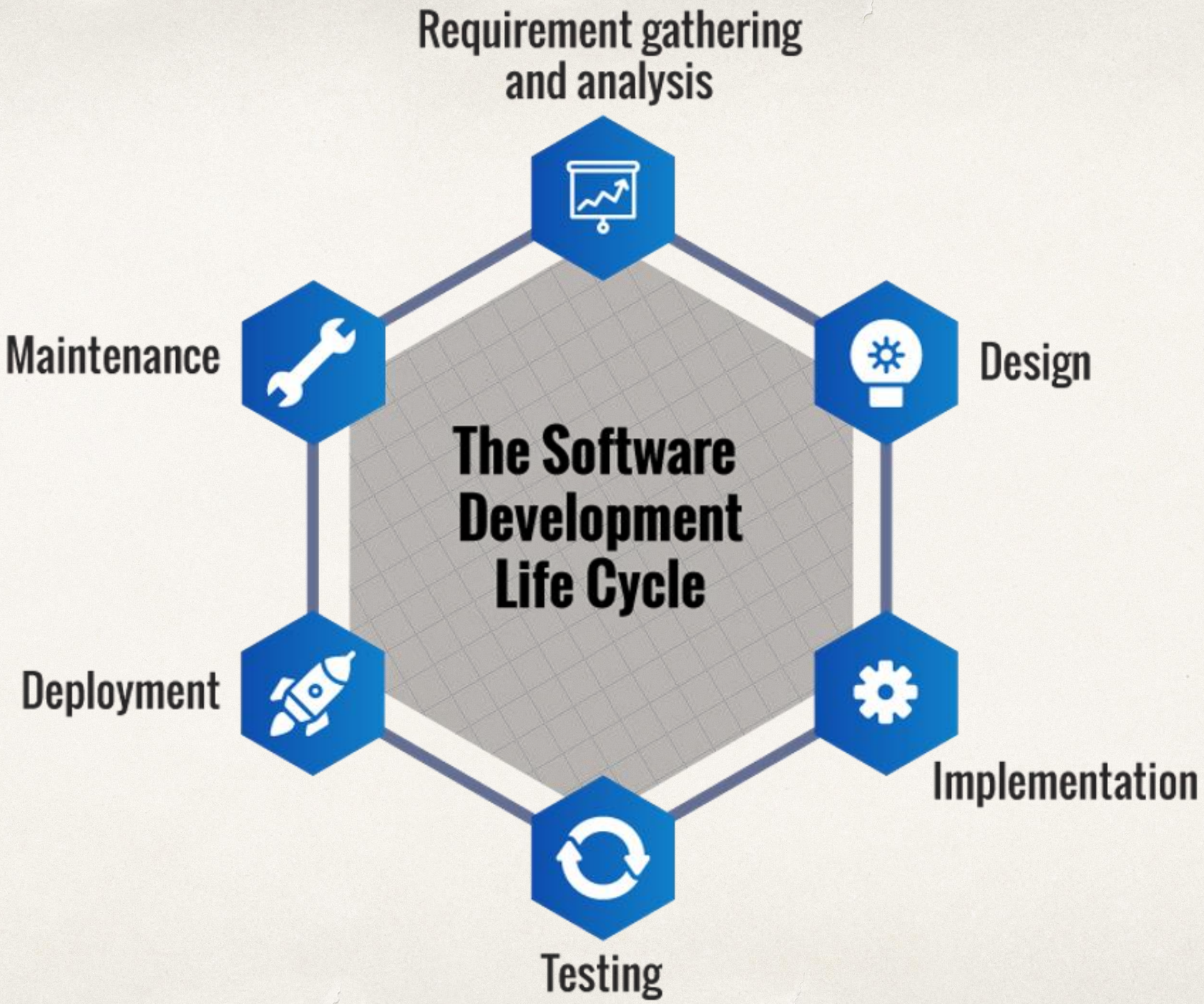
Learning Style

Process oriented — Tinkering

Risk Attitude

Risk-averse — Risk-taker

M. Burnett, S. Stumpf, J. Macbeth, S. Makri, L. Beckwith, I. Kwan, A. Peters, W. Jernigan, Gendermag: A method for evaluating software's gender inclusiveness, Interacting with Computers 28 (6), (2016) 760–787


Requirement gathering and analysis
Design
Implementation
Testing
Deployment
Maintenance
The Software Development Life Cycle

Building inclusive software with GenderMag

- Three personas based on the cognitive facets (Abi, Pat, and Tim)
- Teams do a cognitive walkthrough role playing using their software to complete certain tasks using the personas
- Inclusivity bug: can't complete the task or face disproportionate barriers along the way
- 17 software teams using GenderMag teams found inclusivity bugs in 12%-100% of their software (average 32%).

M. Burnett, S. Stumpf, J. Macbeth, S. Makri, L. Beckwith, I. Kwan, A. Peters, W. Jernigan, Gendermag: A method for evaluating software's gender inclusiveness, Interacting with Computers 28 (6), (2016) 760–787

Building inclusive software with GenderMag

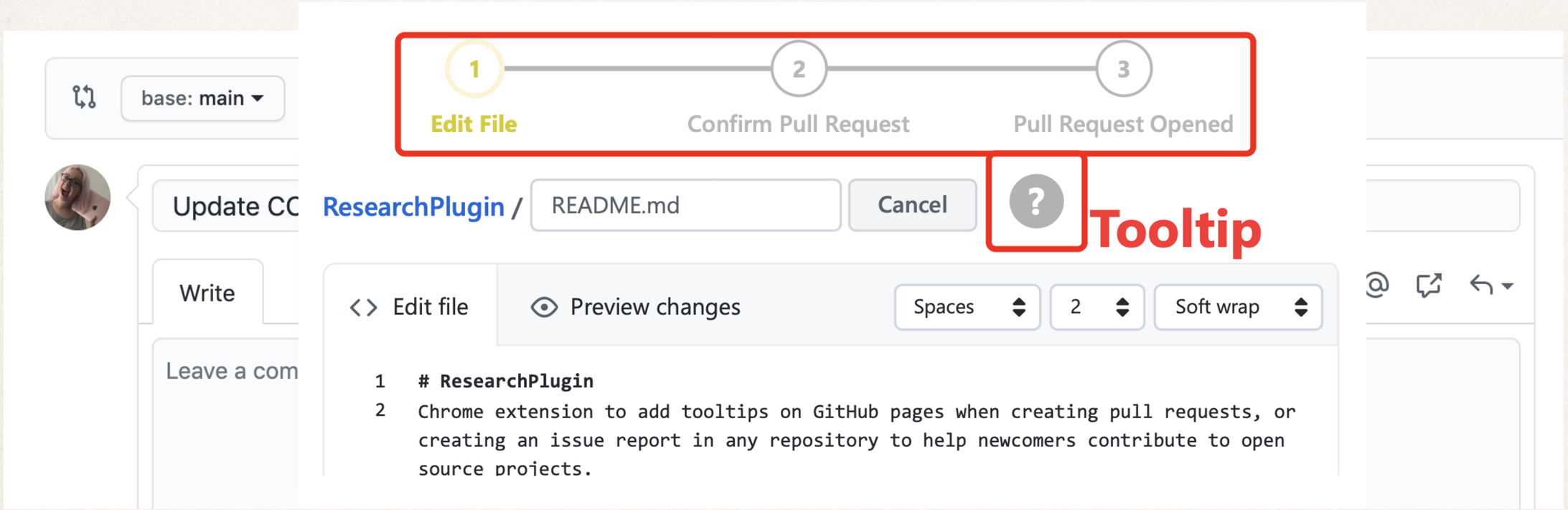
Abi (Abigail/Abishek)

Motivation: Uses technology to accomplish their tasks.
Computer self-efficacy: Lower self-confidence than their peers about doing unfamiliar computing tasks. Blames themselves for problems.
Attitude toward risk: Risk-averse about using unfamiliar technologies that might require a lot of time
Information processing style: Comprehensive
Learning style: Process-orientated learning

M. Burnett, S. Stumpf, J. Macbeth, S. Makri, L. Beckwith, I. Kwan, A. Peters, W. Jernigan, Gendermag: A method for evaluating software's gender inclusiveness, Interacting with Computers 28 (6), (2016) 760–787

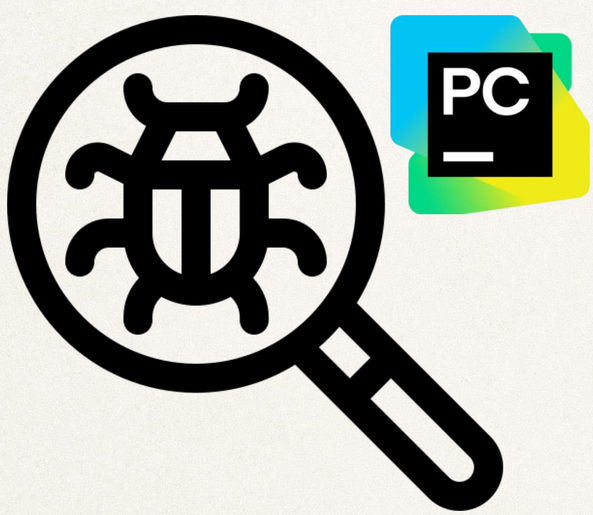
Software tools

- Software engineers use software to create and maintain software
- Inclusivity bugs also found in SE tools like GitHub and code review tools



I. Santos, J. F. Pimentel, I. Wiese, I. Steinmacher, A. Sarma and M. A. Gerosa, "Designing for Cognitive Diversity: Improving the GitHub Experience for Newcomers," 2023 IEEE/ACM 45th International Conference on Software Engineering: Software Engineering in Society (ICSE-SEIS), Melbourne, Australia, 2023, pp. 1-12

Our recent studies



Lizzie Matusov · 2nd
Co-founder/CEO at Quotient | Research...
Visit my website
1d ·

+ Follow

The AI honeymoon phase in engineering is over—and that's good news.

Stack Overflow's 2025 Developer Survey polled 49,000+ developers and revealed a fascinating tension: AI usage hit 51% daily adoption among professional developers, yet favorable sentiment dropped from 70% to 60% in just one year.

The reality check? 66% of developers are frustrated with "almost right" solutions that require time-consuming debugging. It basically means we're moving from hype to realism.

Where AI actually works: ✅ Searching and learning new codebases ✅ Documentation and boilerplate generation ✅ Exploratory work

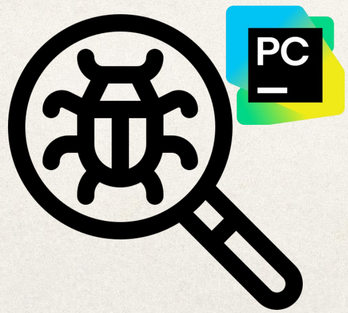
Where developers still don't trust it: ❌ Deployment decisions ❌ Architecture planning ❌ High-stakes production code

And maybe my favorite... 78% of engineers say "vibe coding" isn't how they work. Despite the narrative, professional developers aren't blindly accepting AI suggestions—they're evaluating, verifying, and maintaining human oversight.

The takeaway for engineering leaders: AI tools are here to stay, but success requires setting realistic expectations, reserving AI for appropriate use cases, and investing in teams' ability to critically evaluate AI output.

What we did

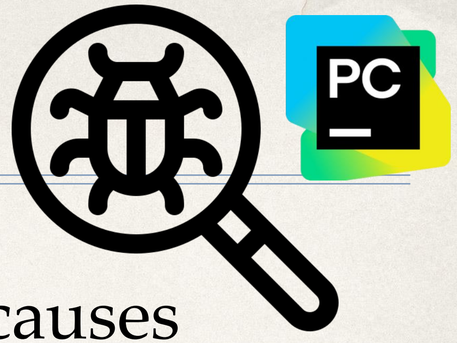
- **Lab experiments:** one-hour, individual session for each participant.
- **Think-aloud protocol:** Participants verbalized their thought processes while performing tasks of increasing complexity.
- **Data Capture:** audio and screen recordings were collected and transcribed
- **Analysis:** We used reflexive thematic analysis to identify inclusivity bugs and the GenderMag facet questionnaire to analyze results through a cognitive inclusivity lens.



24 Participants

20 Participants

What we found

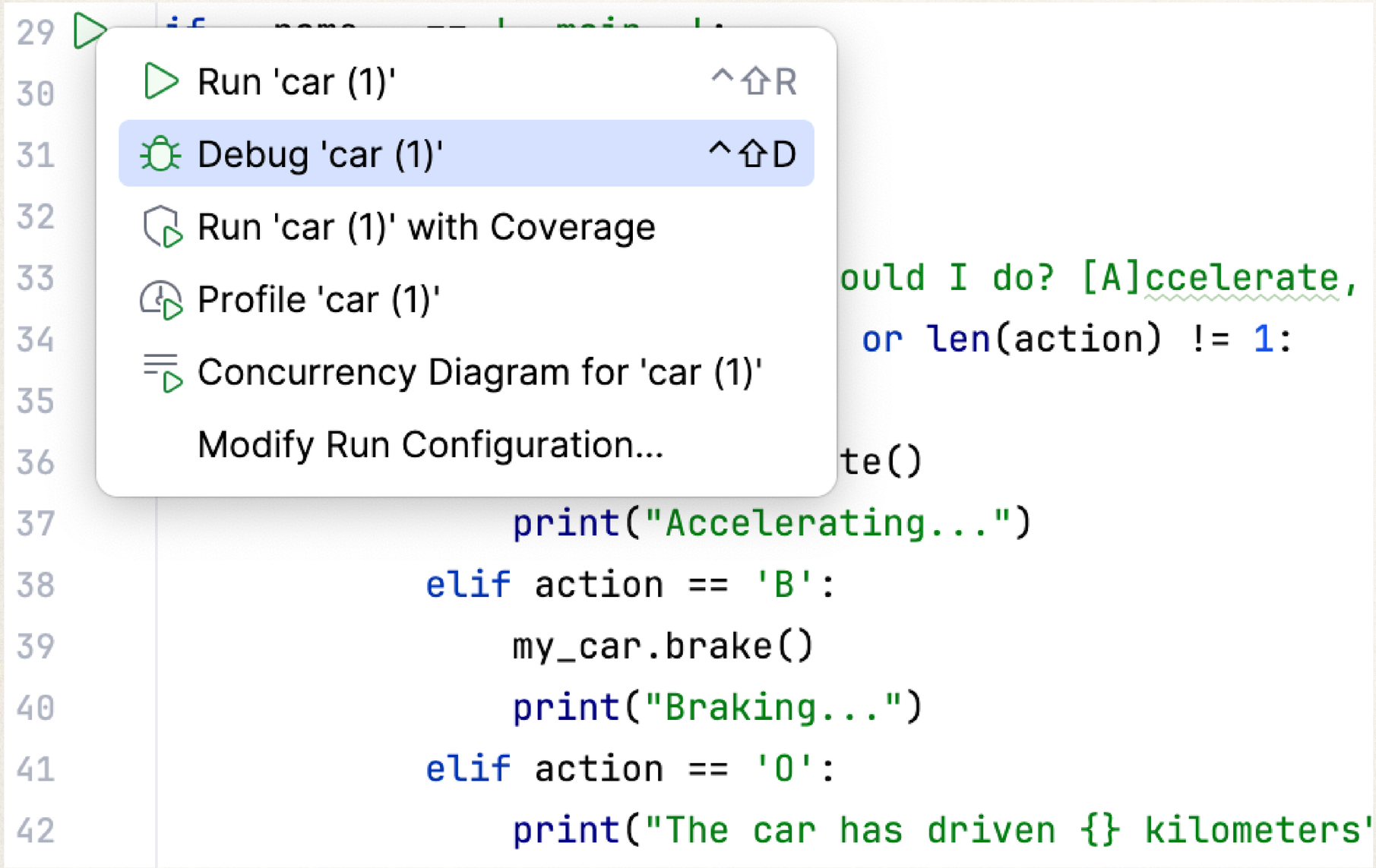


- 21 inclusivity bugs across 13 different features with two main causes
- **Discoverability**: *"I can't find it"*
  - The degree to which users can independently locate features.
  - ***Caused by:*** Cluttered interfaces, poor placement, lack of visual cues, hidden elements, and poor labeling.
- **Learnability:** *"I found it, but I don't understand how to use it"*
  - The degree to which users can understand and effectively use a new feature.
  - ***Caused by:*** Insufficient or unclear feedback from the tool.

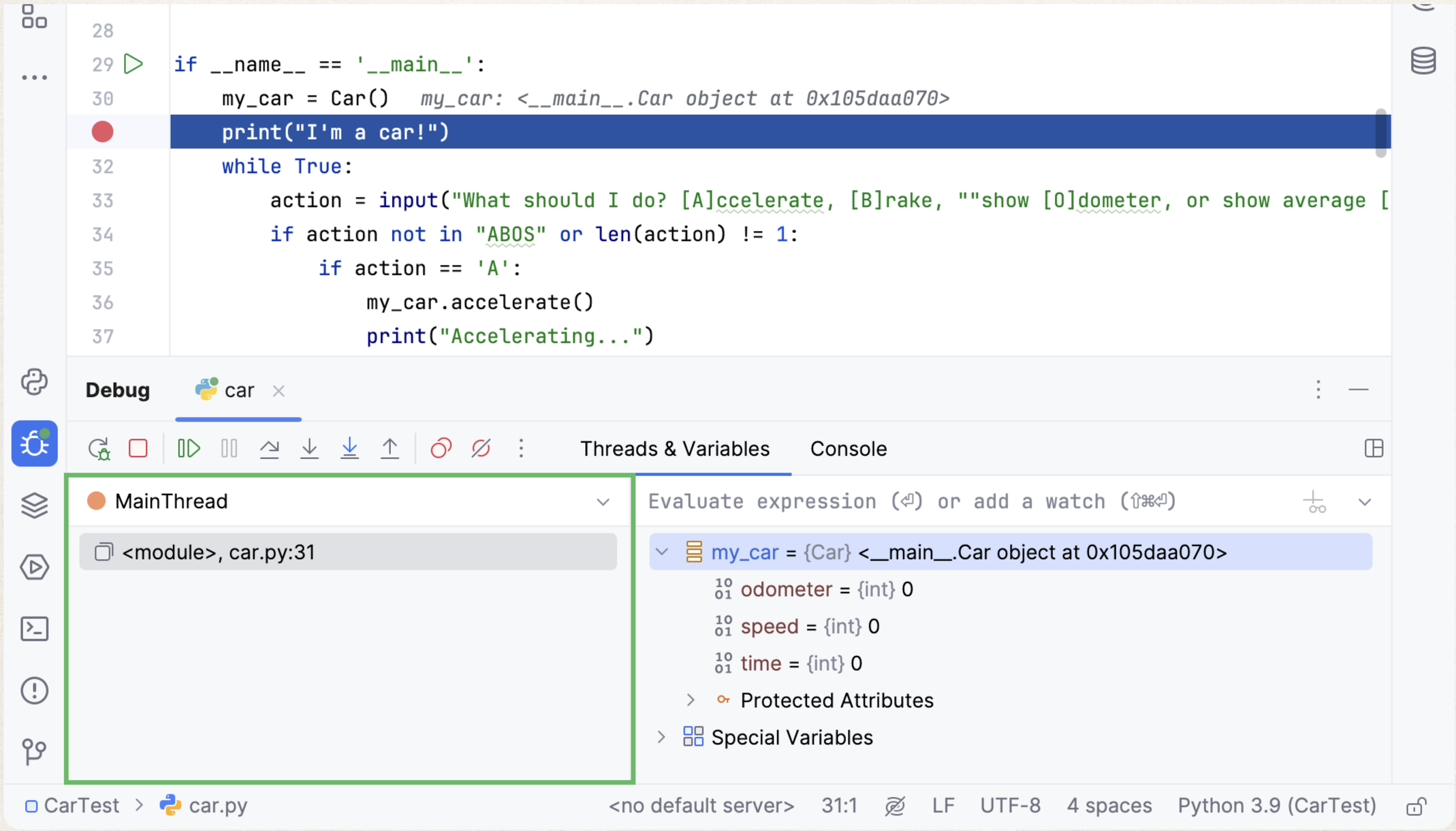
Setting breakpoints

```
3       def __init__(self, speed=0):
4           self.speed = speed
5           self.odometer = 0
6           self.time=0
7
8       def accelerate(self):
9           self.speed += 5
10
11      def brake(self):
12          if self.speed >= 5:
13              self.speed -= 5
14          else:
15              self.speed = 0
```

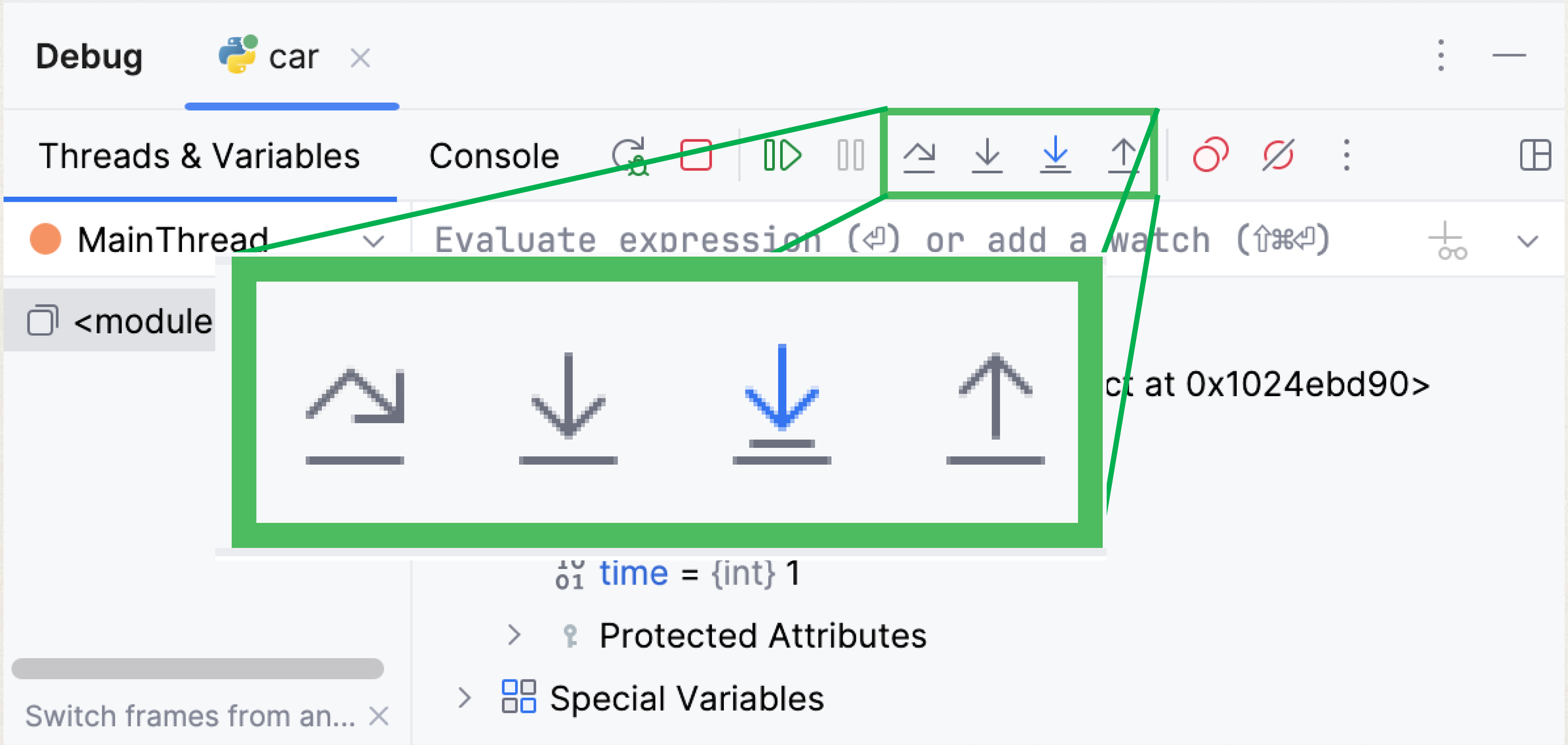
Starting the debugger



Examining suspended program



Stepping through the program



Stepping through a program

- Stepping is the process of controlling step-by-step execution of the program.

Step over: goes to next line and skips method calls

Step into: goes to called methods (even library methods)

Step into my code: goes to called methods in your code only

Step out: goes out of the current method and back to the caller method

Table 4
Number of participants who encountered inclusivity bugs categorised by debugger feature and cause (discoverability and learnability). - indicates no inclusivity bug

	Feature description	Discoverability bug	Learnability bug
1	Setting breakpoint and starting debugger session	5	2
2	Finding the debugger icon to start the debugger	3	-
3	Stopping debugger session	1	-
4	Setting breakpoint at the correct line	-	7
5	Following the execution point	1	2
6	Stepping through program	3	21
7	Examining variables	4	4
8	Managing breakpoints in the middle of a debug session	-	7
9	Evaluating expressions	6	7
10	Resuming program	4	5
11	Exploring test results	13	1
12	Running or debugging tests	14	-
13	Changing run configurations	6	3
	Total instances of inclusivity bugs	60	59

Culas, F., Singh, A., Arankalle, A., Dhopade, P., & Blincoe, K. (2025). Newcomers' experiences during debugging: A cognitive inclusivity perspective using GenderMag. Information and Software Technology, 107932.

Who faced the most inclusivity bugs

Motivation

For task completion — To learn new features

Self-Efficacy

Lower, blames self — Higher, blames tech

Information Processing

Comprehensive — Selective

Learning Style

Process oriented — Tinkering

Risk Attitude

Risk-averse — Risk-taker

M. Burnett, S. Stumpf, J. Macbeth, S. Makri, L. Beckwith, I. Kwan, A. Peters, W. Jernigan, Gendermag: A method for evaluating software's gender inclusiveness, Interacting with Computers 28 (6), (2016) 760–787

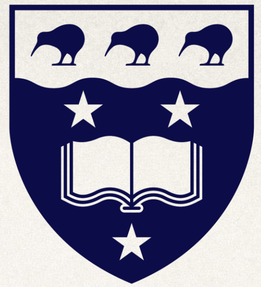
What we found

- 10 inclusivity bugs
  - **Autocomplete-Style Suggestion Mode:** prioritises rapid acceptance, lacks explanations, can be disruptive, and limits creative control

  - **Chat Mode:** verbose chat responses

Conclusion

- Many different ways of thinking – no one right way
- Software engineers *should* design for inclusion
- Software often has inclusivity bugs
- It's not you – it's the software
- We need more diversity in software engineering so we can build better software



Waipapa
Taumata Rau
University
of Auckland

Thanks

Questions?

Kelly Blincoe
k.blincoe@auckland.ac.nz