How can we best support the modeling experience of software developers?

Timothy C. Lethbridge

University of Ottawa, Canada

timothy,lethbridge@uottawa.ca

http://www.umple.org

Objective of this talk

To discuss

- Challenges to uptake of modeling among SE practitioners
- How we have attempted to overcome these using Umple (https://www.umple.org)

Working definition of Model in this talk

- —Any representation to enable design or understanding of software that is distinct from or more abstract than traditional programming languages
- —Often multiple special-purpose views, best if connected
- —Can be textual/diagram (or both)
- —May be prescriptive or descriptive

What have been the greatest recent advances in SE for mainstream practitioners? (1) My opinion

Automation improvements

- Continuous testing and integration
- Git/github/gitlab for version control, reviewing, issue tracking.
- Package/dependency management
  - homebrew, apt, pip(Python), npm(Node.js), cargo(Rust)
- Build tools (e.g. Gradle)
- Code analysis (e.g. spotbugs, SonarQube, linters)

Other agile methods

- Short sprints, user focus

Question answering and tutorial sites: Stack overflow, many others

What have been the greatest recent advances in SE for mainstream practitioners? (2) My opinion

Advances in languages (e.g. Dart, Rust, R, Scala, Kotlin, Elixir, Python …)

Frameworks

- Node.js, Angular, Django, React, Ruby on Rails, etc.

Encapsulation of algorithms

- Incorporation of machine learning, data analysis into languages and libraries (Particularly in Python)

Simpler, free IDEs with rapid update cycles, plugins, searching

- E.g. Visual Studio Code

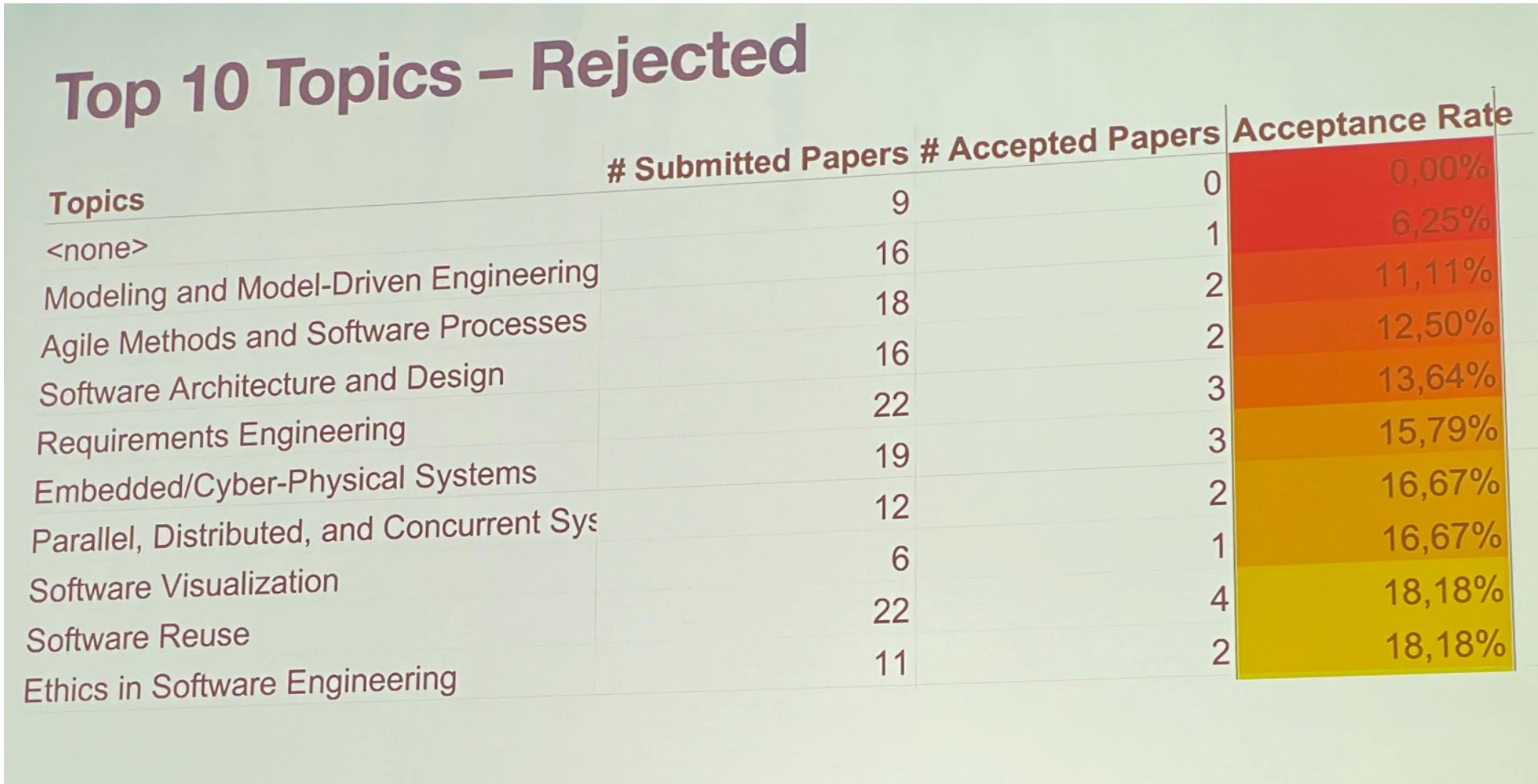
Where is modeling among this list of advances?

Modeling is not on the list!

- Advances in modeling tend to be very specialized
  - E.g. formal transformations
- Poor utility (feature set), usability of tools -- more later
- Adoption of tools is weak
  - Taught in courses, but then not used much except for safety-critical systems
- Much usage is just on whiteboards

- Other advances lead people to feel they don’t need modeling ??

Papers getting rejected: Stats from ICSE last week



Top 10 Topics – Rejected

Topics	# Submitted Papers	# Accepted Papers	Acceptance Rate
<none>	9	0	0,00%
Modeling and Model-Driven Engineering	16	1	6,25%
Agile Methods and Software Processes	18	2	11,11%
Software Architecture and Design	16	2	12,50%
Requirements Engineering	22	3	13,64%
Embedded/Cyber-Physical Systems	19	3	15,79%
Parallel, Distributed, and Concurrent Sys	12	2	16,67%
Software Visualization	6	1	16,67%
Software Reuse	22	4	18,18%
Ethics in Software Engineering	11	2	18,18%

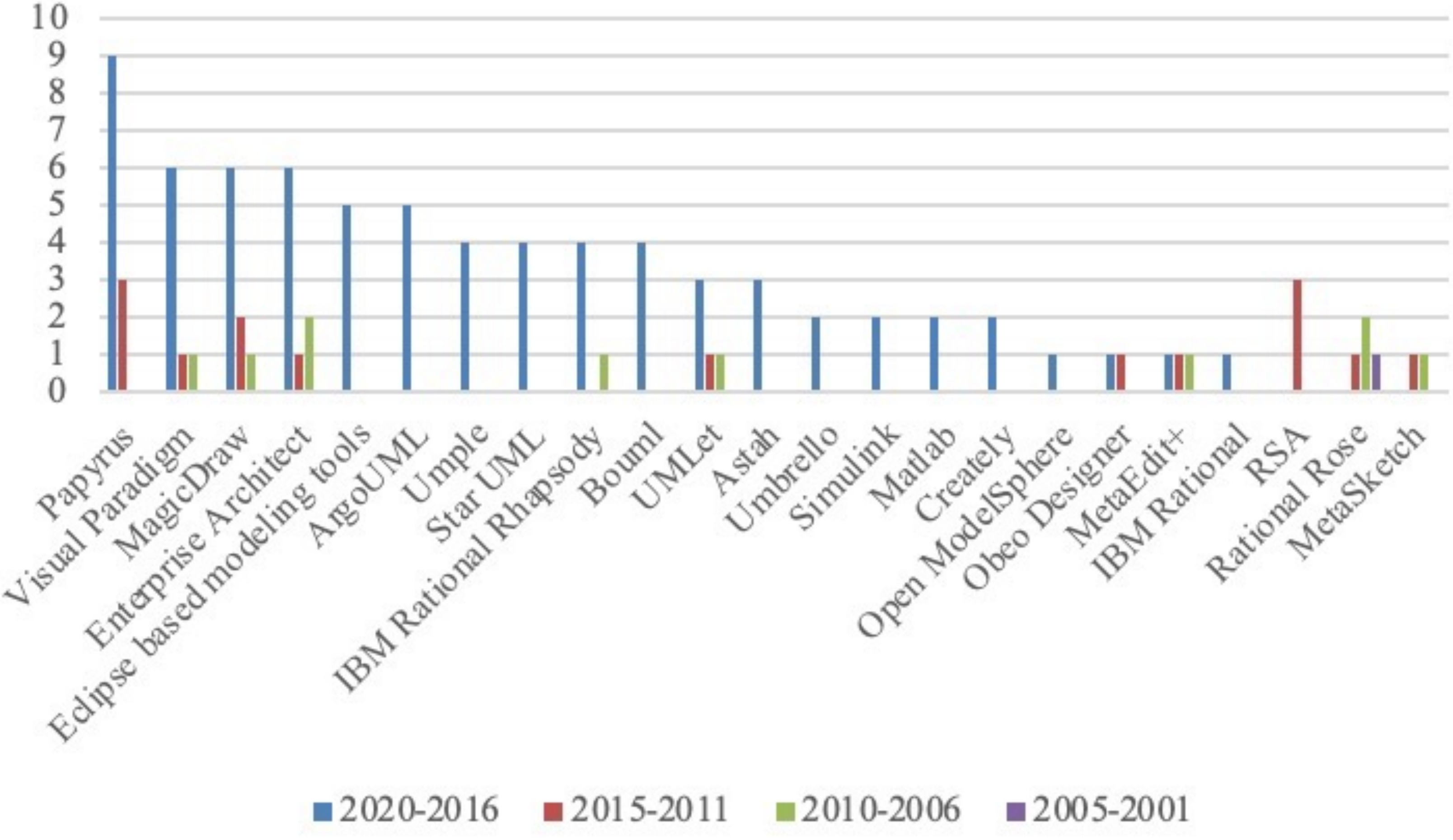
Modeling Experience (MX) definition:

User experience for software modeling

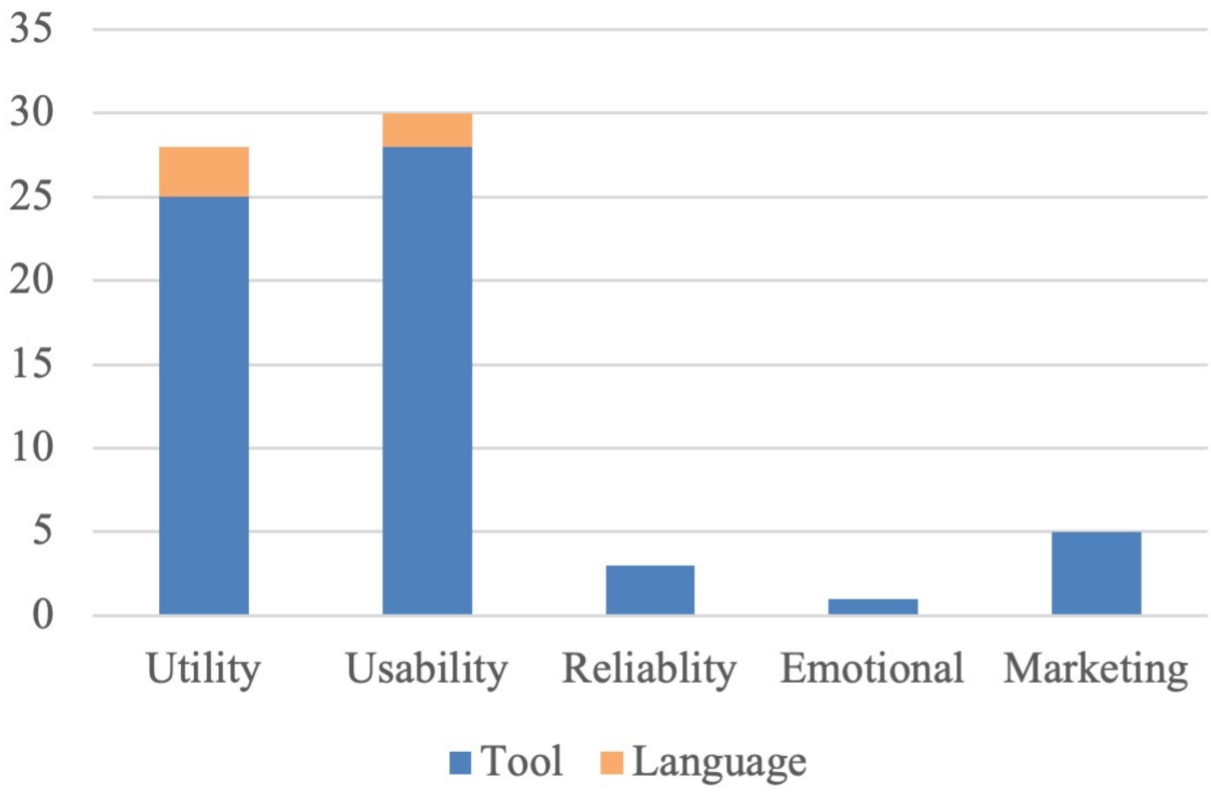
- Introduced by Abrahao et al

Tools mentioned in our literature review of MX studies

From submitted paper led by my PhD student Reyhaneh Kalantari, n=41 relevant papers



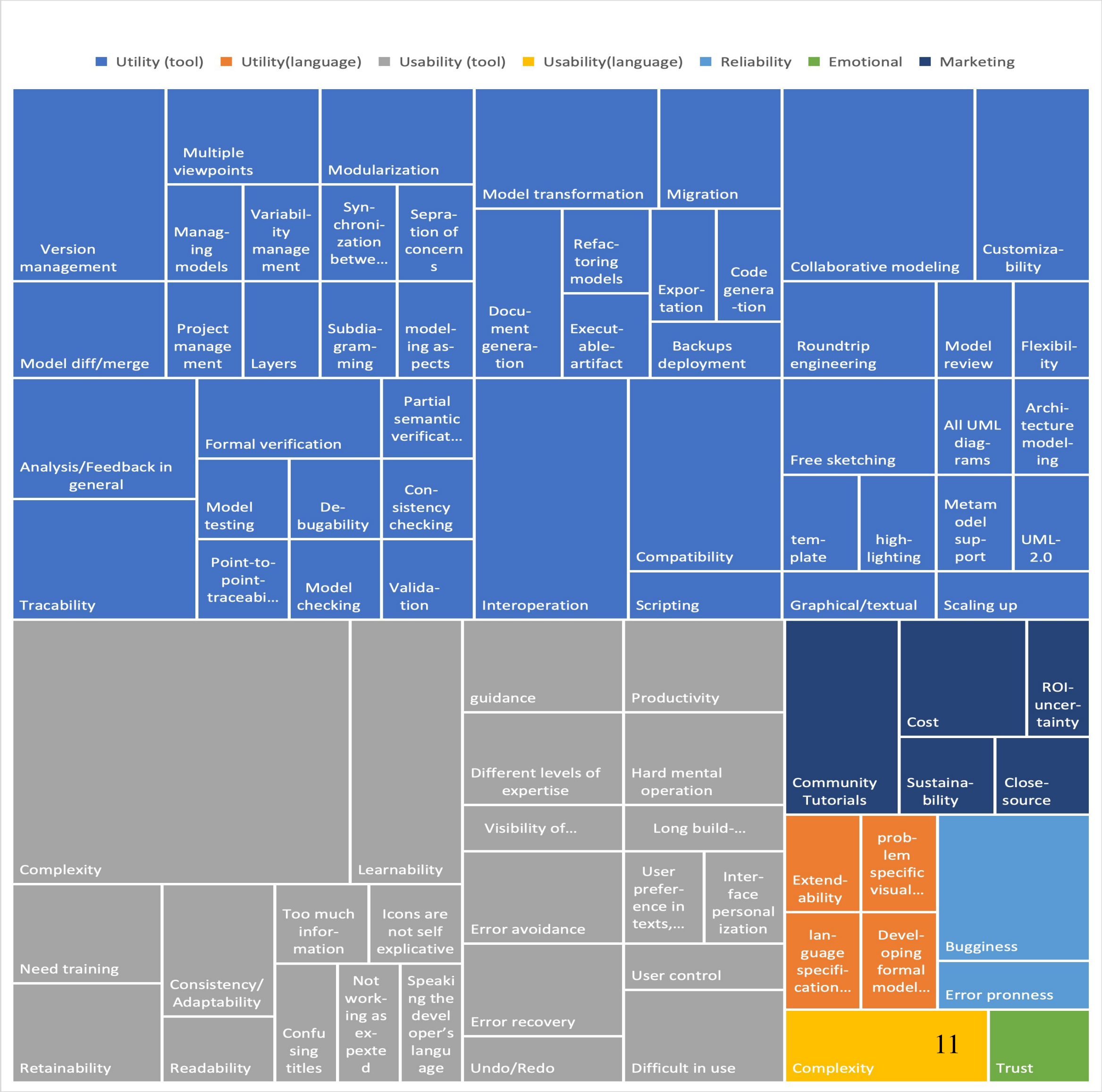
Number of publications mentioning 5 categories of MX issues



Treemap of issues

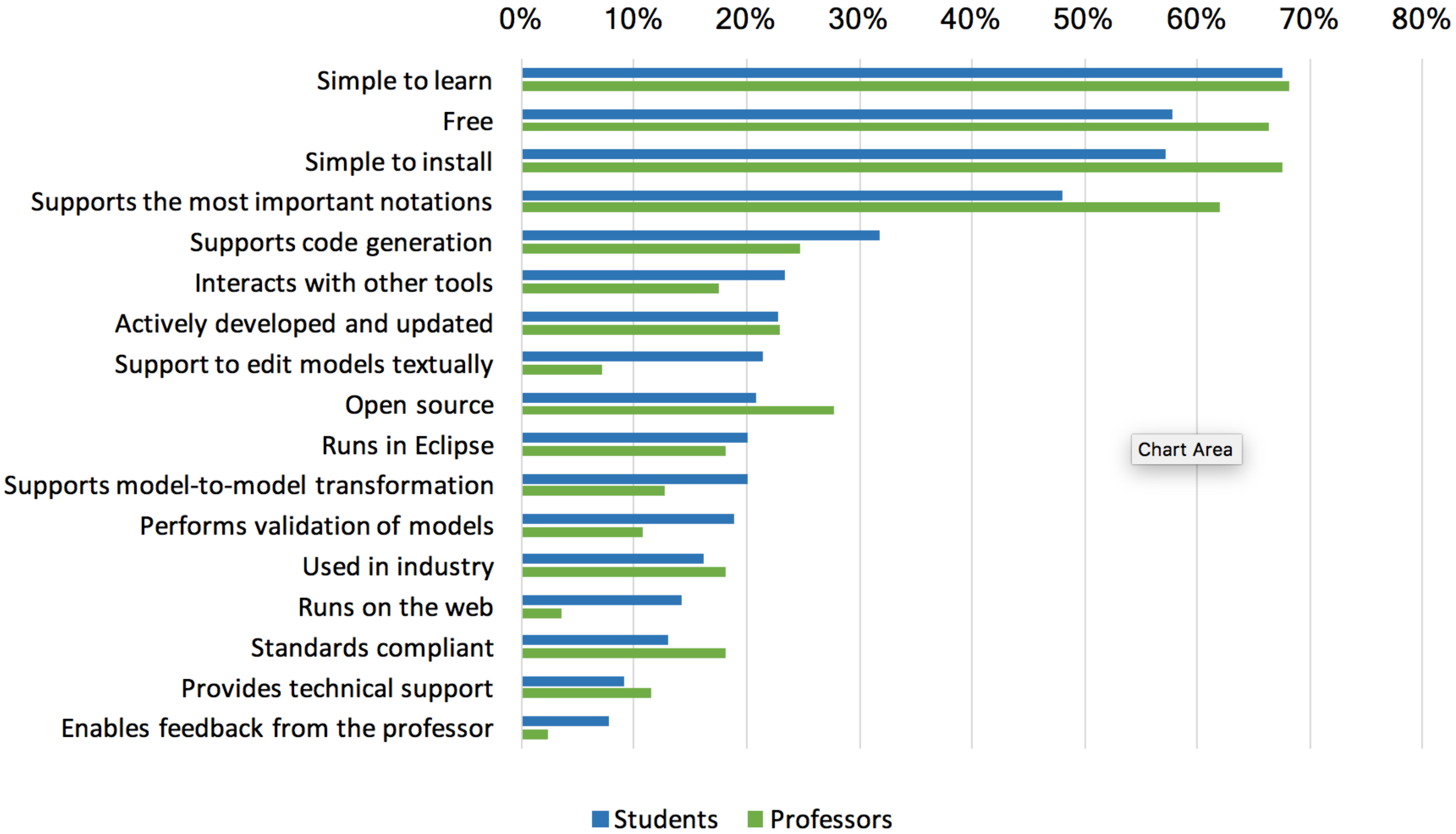
Categories:

- Utility
  - —Tool (top)
  - —Language
- Usability
  - —Tool (bottom left)
  - —Language
- Reliability (right near bottom)
- Emotional (bottom left)
- Marketing (centre right)

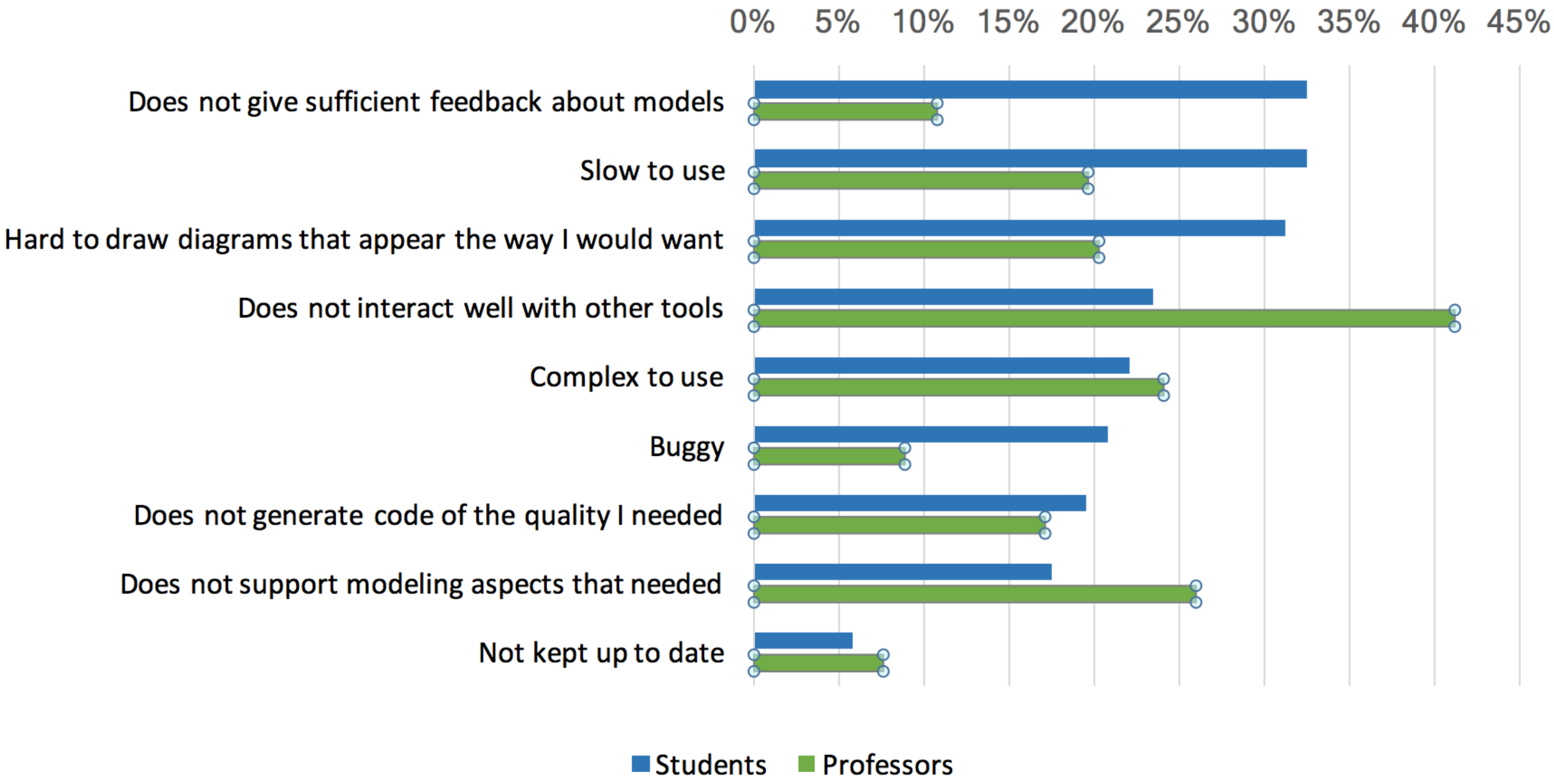


Desired MX attributes our 2017 study of Modeling tools

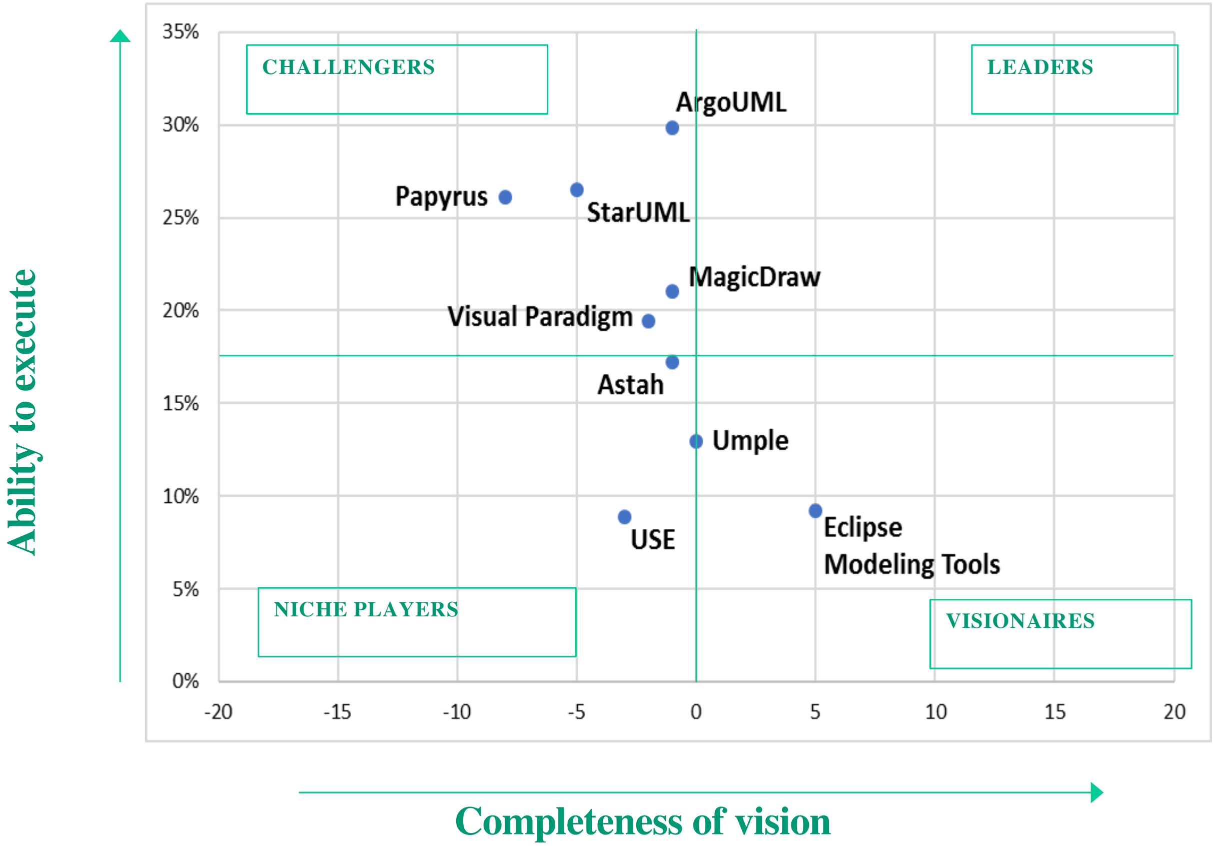
(n=117 students; n=134 professors; papers with Luciane Agner, Models / SoSyM)



Key MX complaints from our 2017 study



MX: Magic Quadrant – Nobody at top right quadrant in 2017



The way forward

Out with the old modeling tech?

- Is Eclipse too long in the tooth?

Out with the complex

First make it executable/compilable

- (i.e. code generation for real systems)
- Doesn't preclude using models purely for informal design
  - —Just like circuit models can be used for simulation, analysis, and generation of ICs

Then, integrate with other software engineering advances

Integrating modeling with the greatest recent advances in SE (1)

Integrate with automation

- Git and related tools: Implies it needs to have a textual concrete syntax
- Package management of model libraries
- Building of systems with models using build tools like Gradle

With the above, model-driven development can embrace agility!

Rich info: MDE tools need a presence on Stack Overflow and have extensive and deep manuals and tutorials

Integrating modeling with the greatest recent advances in SE (2)

Embrace integration of modeling with:

- Many programming languages
- Frameworks
- Algorithm libraries
- Multiple IDEs + command line

Umple *does* support many recent advances (demos 1)

Main link: https://www.umple.org

Fully compilable

- Model-driven: Written in itself

Textual, embracing Git, build tools, and package managers for development

- E.g for Mac/Linux. `brew install umple`

Developed using agile methods (demonstrating agile MDE)

- Test-driven
- Continuous integration

Umple *does* support the recent advances (demos 2)

Extensive assistance: Analysis, Manual, Stack overflow, self-documenting

- https://manual.umple.org
- https://cruise.umple.org/umple/umple-core-classDiagram.shtml

Multi-language support

- Java, Php, Ruby, C++ (beta), Python (coming)

IDE support: Web, VS-Code, Command Line, Eclipse

- Now executable on the web as well as the command line
- Docker image of web version available for local use

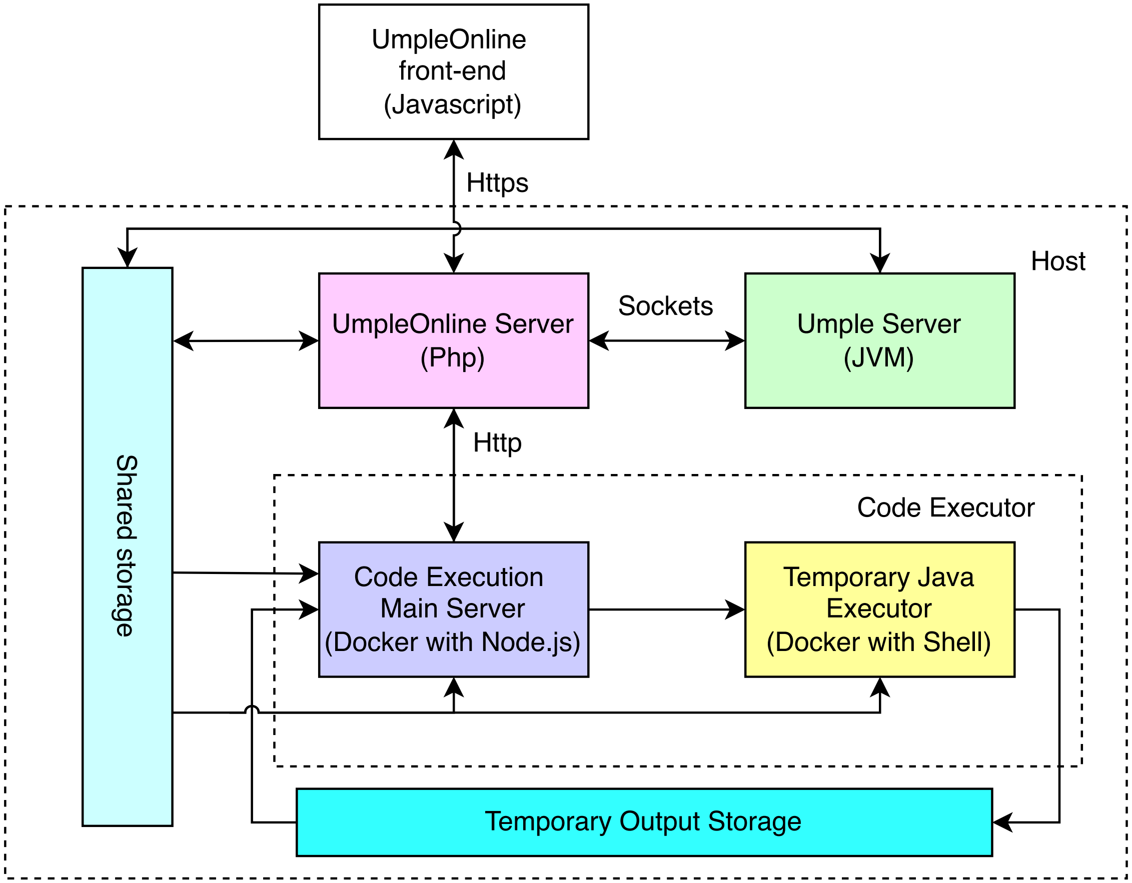
Other Umple highlights (demos 3)

Developed by almost 70 students and professionals over 15 years

- 200,000+ user sessions per year of the web version

Speedy to install and execute (we fixed pre-2017 problems)

- UmpleOnline runs compiler in server mode with micro-services architecture (green at right)
- Server mode can also be used on command line


UmpleOnline front-end (Javascript)
Https
Host
Shared storage
UmpleOnline Server (Php)
Sockets
Umple Server (JVM)
Http
Code Executor
Code Execution Main Server (Docker with Node.js)
Temporary Java Executor (Docker with Shell)
Temporary Output Storage

Other Umple highlights (demos 4)

Wide array of input and generation targets (model transformations)

- **In**: Plain Umple, Edits of diagrams, XMI from Papyrus, Reverse engineered Java
- **Out**: Code, Tables, Diagrams, Formal methods

Low bug rate due to extensive testing and in-practice use

Enhanced through multiple usability studies, and extensive user-feedback

So why add modeling (in Umple) to your toolchain?

Much less code to write and maintain

Numerous features available in models *and* multiple programming languages

- Mixsets: Product line and feature-oriented development
  - ‘Feature toggles’ as per earlier talk today
- State machines
- Aspects
- Traits with model elements

Multiple views (diagrams, tables, analysis)

- Improved understandability, maintainability etc.

Thanks to

Open source contributors

- Andrew Forward, Omar Badreddin, Dusan Brestovansky, Julie Filion, Miguel Garzon, Hamoud Aljamaan, Ali Fatolahi, Julian Solano, Joshua Horacsek, Joel Hobson, Alvina Lee, Sultan Eid, Jordan Johns, Sonya Adams, James Zhao, Adam Dzialoszynski, Luna Lu, Song Bae Choi, Thomas Morrison, Sacha Bagasan, Andrew Paugh, Stuart Erskine, Russell Staughton, Christopher Hogan, Geoffrey Guest, Gabriel Blais Bourget, Robin Jastrzebski, Quinlan Jung, Blakeley Quebec Desloges, Jesus Zambrano, Ahmed Orabi, Mahmoud Orabi, Tonio Resende, Vahdat Abdelzad, Opeyemi Adesina, Aliaa Alghamdi, Tiago Nascimento, Tianyuan Chu, Fiodar Kazhamiaka, Greg Hysen, Jean-Christophe Charbonneau, Kenan Kigunda, Adriaan Cody Schuffelen, Marc Antoine Gosselin-Lavigne, Pedro Augusto Vincente, Jason Canto, Ellen Arteca, Alexi Turcotte, Karin Ng, Mark Galloway, Alexander Ringeri, Antonio Maria Pereria de Resende, Craig Bryan, Eric Telmer, Charles Wang, Chan Chun Kit, Nabil Maadarani, John Zweip, Kevin Brightwell, Warren Marivel, Ashley Merman, Xinxin Kou, Aymen Ben Rkhis, Curtis Meerkerk, Adam Kereliuk, Matthew Fritze, Michael Mkicik, Victoria Lacroix, Morgan Redshaw, Matthew Rodusek, Shikib Mehri, Amid Zakariapour, Marc de Niverville, Alex Hochheiden, Noah Murad, Katharine Cavers, Jackie Lang, Adam Bolding Jones, Chang Ding, Joshua McManus, Balaji Venkatesh, Runqing Zhang, Finn Hackett, Daniel Mitchell, Richard Hugessen, Bowei (Bernard) Yuan, Abdulaziz Algablan, Zainab Al Showely, Gloria Law, Yiran Shu, Evgeniya Vashkevich, Paul Wang, Firas Jribi, Jingyi Pan, Haowen Shi, Ralph Ngassa, Svetlana Esina, Samuel Labonté, Thierry Laprade, Yanic Mainville, RJ Stead

Core funding: NSERC, ORF

Current Host: Digital Research Alliance of Canada, formerly Compute Canada

Early corporate support: Google, Facebook, IBM

See you all at ICSE 2025 in OTTAWA, Canada

April 26-May 3, 2025

(Main conference April 29-May 2)

To volunteer or provide your ideas:
https://bit.ly/2Swzi6Y

uOttawa

Carleton UNIVERSITY

Thank-you!
These slides will be in
https://cruise.umple.org/presentations/

To Cite Umple: Timothy C. Lethbridge, Andrew Forward, Omar Badreddin, Dusan Brestovansky, Miguel Garzon, Hamoud Aljamaan, Sultan Eid, Ahmed Husseini Orabi, Mahmoud Husseini Orabi, Vahdat Abdelzad, Opeyemi Adesina, Aliaa Alghamdi, Abdulaziz Algablan, Amid Zakariapour, "**Umple: Model-Driven Development for Open Source and Education**", *Science of Computer Programming*, 2021, https://doi.org/10.1016/j.scico.2021.102665.

Cite latest software release as: University of Ottawa, Umple, https://doi.org/10.5281/zenodo.4677562