AI, Education, and the Future of Software Expertise

USA:
Anshu Dubey - adubey@anl.gov / anshu.uchicago@gmail.com
France:
Roberto Di Cosmo (INRIA) - roberto@dicosmo.org
Raymond Namyst (INRIA) - raymond.namyst@u-bordeaux.fr
Japan:
Jens Domke (RIKEN) - jens.domke@riken.jp
LAC:
Alfredo Goldman (USP) - gold@ime.usp.br
Antonio Tadeu Gomes (LNCC) - atagomes@lncc.br

Core Tension

- AI increases productivity and lowers barriers to coding
- It may also reduce opportunities to build deep expertise
- Challenge: gain AI benefits without losing human capability

Possible Approaches

- Manual coding may become a specialized skill
- Emphasis shifts to specifying, validating, and adapting solutions

Education must Change

- Assignments rewarding only final answers are no longer enough
- Assessment must focus on visible understanding, not output alone
  - Explain approach and tradeoffs
  - Trace execution and predict behavior
  - Debug broken solutions
  - Adapt solutions under new constraints
  - Explain AI-generated code
- Teach prompting, critique, refinement, and verification
- Students should prove understanding even with AI assistance
- Teach creativity alongside algorithms.
- Avoid convergence to standard solutions.
- Use multiple paradigms: coding, block programming, visual dataflow.

Why fundamentals still matter

- Algorithms, debugging, data structures, and systems remain essential
- Engineers need mental models for behavior, edge cases, and performance
- Without fundamentals, users may overtrust outputs

Expertise at risk

- Loss of low-level debugging ability
- Reduced skill in tracing execution and interactions
- Weaker independent judgment
- Fewer pathways for training future experts

New risks

- Rapid artifact generation can outpace review processes
- Skipping disciplined workflows
- AI verifying AI can reinforce errors
- Common models may reduce originality

Scientific software has extra complexity

- Requires computational, numerical, and domain expertise
- Developers rely on trusted libraries/frameworks
- LLM quality depends on ecosystem representation in training data

Opportunities from AI

- Broader participation in computing and science
- Faster prototyping and experimentation
- Searchable expertise systems
- AI assistants that explain reasoning

New roles for research software engineers

- Validate AI-generated systems
- Maintain infrastructure
- Optimize performance
- Assure safety/correctness
- Translate domain intent into AI workflows

Final Takeaways

- AI will change software development
- Human expertise remains essential
- Education should shift toward reasoning and validation
- Goal: humans with AI who remain capable, critical, and creative