Envisioning the Future of Software Engineering

July 2021

Dr. Forrest Shull

Lead for Defense Software Acquisition Policy Research,
Carnegie Mellon University / Software Engineering Institute

President,
IEEE Computer Society

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

DM21-0632

Outline

CMU/SEI National Agenda for Software Engineering Study:

- Motivation
- Methodology
- 10-15 Year Research Roadmap
- Recommendations

CMU SEI - a US Federally Funded Research and Development Center

Our mission: Engineering and securing software

Established in 1984 at Carnegie Mellon University

US FFRDC charged to improve the state of the practice of software engineering, cyber security, and AI engineering.

Focus of National Agenda for Software Engineering

Software is vital to America's **global competitiveness, innovation**, and **national security**. The economy, the nation's infrastructure, education, and healthcare all depend on software.

Lead a community effort to:

1. **Identify future challenges** in engineering software-reliant systems.
2. Develop a **research roadmap** that will drive advances in **foundational software engineering principles** across system types such as intelligent, autonomous, safety-critical, and data intensive systems.
3. Raise the **visibility** of software to the point where it receives sustained recognition commensurate with its importance to national security and competitiveness.
4. Enable strategic partnerships and collaborations to **drive innovation among industry, academia, and government**.

Progress in 10-year increments: 2001 Workshop

New Visions for Software Design & Productivity: Research & Applications (Vanderbilt, NITRD, NSF)

Progress cited:

- Productivity gains: "…small teams can now create high-quality software systems ***up to 100,000+ LOC (lines of code)*** in a fraction of the time it used to take"
- New technologies having an impact: user interface generators, middleware, APIs, MBE, application frameworks
- Maturation and adoption of lifecycle process models like Rational Unified Process, eXtreme Programming

Recommendations – Increased efforts in 3 broad research areas:

- Better ways to specify / manage requirements – especially for networked / embedded systems
- Better software development environments
- Sophisticated testbeds that can simulate realistic operational situations

https://www.nitrd.gov/pubs/sdp_wrkshp_final.pdf

Progress in 10-year increments: 2010 Workshop

Future of Software Engineering Research (FoSER) (NITRD, NSF)

Progress cited:

- Modern programming mechanisms & software development methods;
- Mathematical techniques for verifying software behavior & finding faults;
- Methods for structuring software artifacts, ***sometimes comprising tens of MLOCs*** for human understanding and maintainability;
- Globally distributed teams.
- *"Google now dwarfs all of the greatest libraries in human history… More than one in ten people of the entire human population is registered with Facebook"*

Recommendations included:

- Democratize and broaden participation in production of software
- Address societal grand challenge problems – address unprecedented complexity of modern systems
- Design for dependability and usability
- Automate software evolution
- Strengthen empirical research foundations
- Expand formal methods research
- Incorporate social science research

https://www.nitrd.gov/pubs/FOSER_report_2011.pdf

2020-21 Study: New System Types Require New R&D

New types of systems

- very adaptive defense mission systems
- systems that perform data fusion at a huge scale
- highly engineered business enterprises
- personal digital assistants—that really assist
- dynamically integrated healthcare
- smart cities, buildings, roads, cars, and transport

Trends

- scale motivating the need for safe and resilient software composition
- rapid and assured continuous software evolution
- addressing workforce gaps in software talent
- artificial intelligence (AI)-inspired automation
- evidence-based assurance that a system behaves as intended
- impact of cyber-social platforms on social behavior, creating societal scale impact
- primacy of data (data has become as important as code)

Approach

- Advisory Board
- Computing Landscape
- Emerging Technologies
- Literature Review
- Expert Interviews
- Workshops
- Future Scenarios

Cast a wide net with input from many communities

National Agenda Study: Roadmap / Outcome

Codify findings in an actionable way

Diverse Ecosystem

Ecosystem acts on findings

A Small Subset of Example Participants

Academia / Govt Research

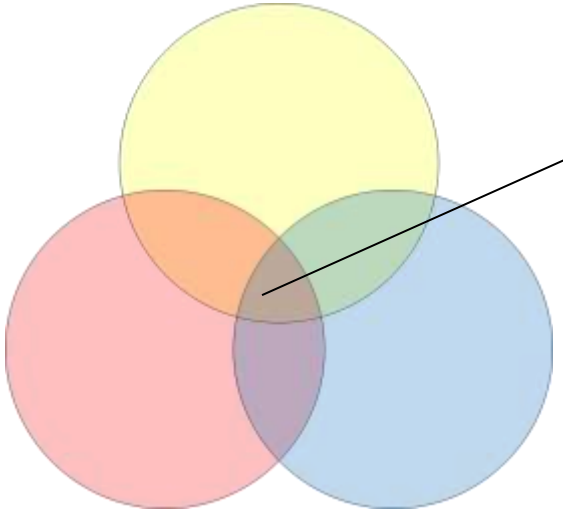
- **Dr. Sergey Bratus,** DARPA PM, Information Innovation Office; winner of 2013 BlackHat Pwnie award for Most Innovative Research
- **Dr. Hal Finkel,** Program Manager in DOE's Office of Science, Advanced Scientific Computing Research (ASCR)
- **Dr. Sol Greenspan,** NSF Program Director for Software and Hardware Foundations (SHF), Cyberinfrastructure for Sustained Scientific Innovation (CSSI), and others
- **Dr. Jim Herbsleb,** Director, Institute for Software Research, CMU
- **Dr. Laurie Williams,** Distinguished Professor, North Carolina State University (NCSU); Winner of the ACM SIGSOFT Influential Educator Award

Industry

- **Mr. Mark Boyd,** Senior Technical Fellow in Software and Chief Software Engineer in Boeing AvionX
- **Mr. Patrick Lardieri,** Lockheed Martin Fellow for Cyber
- **Dr. John Launchbury,** Chief Scientist, Galois, Inc
- **Mr. Jacob Torrey,** Manager, Secure Hardware and Foundational Technologies, Amazon Web Services (AWS)
- **Robin Yeman,** Lockheed Martin Space Senior Fellow

Mission Experts

- **Dr. Jared Dunnmon,** Technical Director, AI/ML, at Defense Innovation Unit (DIU)
- **Dr. Amy Henninger,** Senior Advisor for Software and Cybersecurity at DOT&E
- **Ms. Hannah Hunt,** Chief Product and Innovation Officer at Army Futures Command Software Factory
- **Dr. David Martinez,** Laboratory Fellow, MIT Lincoln Labs



These groups, and especially their intersection, help to identify gaps and provide important insights

Emerging Vision of the Future of Software Engineering

The current notion of software development will be replaced by one where **the software pipeline consists of humans and AI as trustworthy collaborators that rapidly evolve systems based on user intent.**

Advanced development paradigms lead to efficiency and trust at scale.

- Humans leverage trusted AI as a workforce multiplier for all aspects of software creation.
- Formal assurance arguments are evolved to assure and efficiently re-assure continuously evolving software.
- Advanced software composition mechanisms enable predictable construction of systems at increasingly large scale.

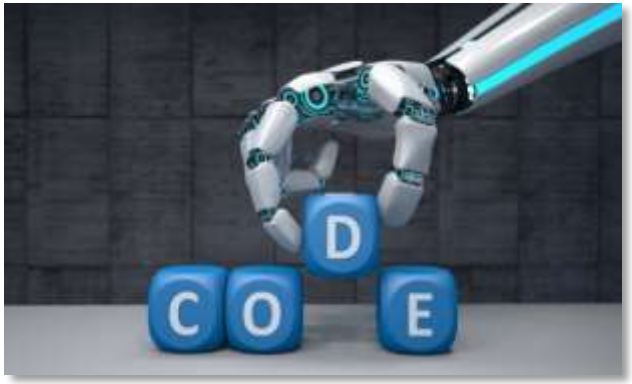
Advanced architectural paradigms enable the predictable use of new computational models.

- Theories and techniques drawn from the behavioral sciences are used to design large-scale socio-technical systems, leading to predictable social outcomes.
- New analysis and design methods facilitate the development of quantum-enabled systems.
- AI and non-AI components interact in predictable ways to achieve enhanced mission, societal, and business goals.

AI Augmented Software Development - 1

Leverage advances in deep learning and search-based algorithms to support development of next generation software design, evolution, and conformance tools, to enforce sound software engineering principles into system development.

- Augment each stage of software development with AI to orchestrate rapid development, continuous systems evolution, and rapid deployment of software-intensive systems.
- Orchestrate continuous evolution by distributing peer roles and responsibilities appropriately between human-led tasks and AI.
- Develop techniques to understand human expression of intent.



AI Augmented Software Development - 2

Open research challenges for AI-Augmented Software Development:

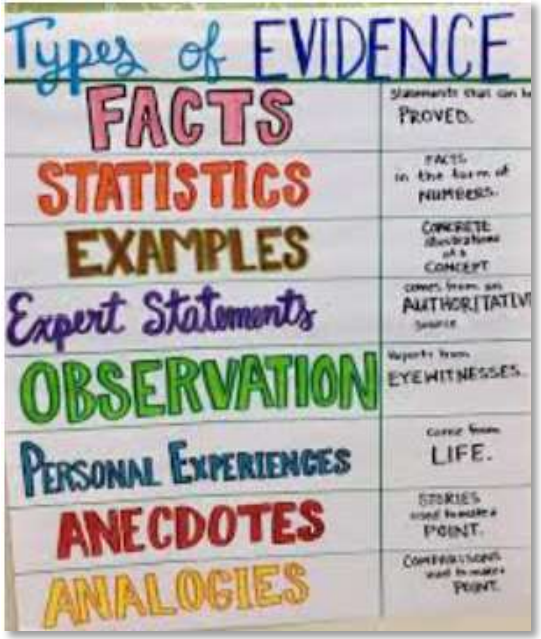
- Designing new phases or activities, or re-design the existing activities, to re-think the software development life-cycle in an AI-Augmented paradigm.
- Incorporating the elicitation of user intent.
- Obtaining data to model each stage/work flow of an AI-Augmented paradigm.
- Identifying roles humans and AI can perform effectively so AI is a trustworthy peer.
- Automatically accumulating and carrying along evidence of quality; Verifying results are correct—use AI to generate meta-data needed to efficiently verify or validate code; generate proof with code.
- Determining how AI will orchestrate continuous systems evolution.

Assuring Continuously Evolving Systems - 1

This research area focuses on providing **evidence-based** assurance arguments that a system will behave as intended, considering both desired functionality and quality attributes, as it **evolves** continuously to incorporate new capabilities and dynamically self-adapts its operating configuration(s) at runtime in response to changing mission demands and environmental conditions.

This requires

- "Theory of evidence"
- "Theory of evolution"



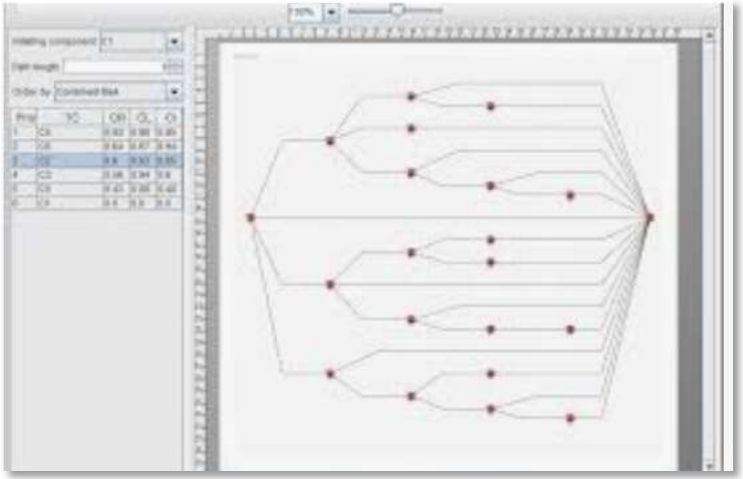
Assuring Continuously Evolving Systems - 2

"Theory of evidence" for

- combining different types of evidence and …
- automatically creating (semi-)formal arguments or assurance cases

"Theory of evolution" for

- understanding how potential system changes propagate and possibly lead to new failure modes or vulnerabilities
- automatically generating and analyzing runtime data and generate system improvement suggestions
- combining human expression of intent and with operation data to generate and assure change requests
- using AI to automatically generate evolution trajectories



Software Construction Through Compositional Correctness -1

This research area focuses on the development of software -reliant systems (and systems of systems) constructed from modular components, where behaviors and quality attributes of the compositions of the components are more robust and resilient than the component parts in isolation.

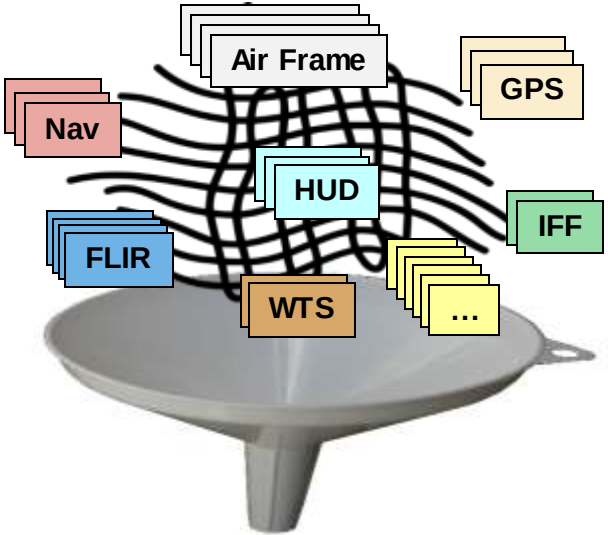
Achieving success in this area requires a new theory of composition and associated patterns and integrated tool chains that can assure the following properties.

- The effort required to (re)assure and (re)certify the entire system as it evolves should be minimized, for instance by bounding the effort required to (re)assure/(re)certify individual components and subsets of composed components.
- The complexity and cost of assurance should grow no more than linearly with the size of the system and the scope of the changes.

Software Construction Through Compositional Correctness - 2

Capabilities needed to support compositional correctness at scale.

- A theory of composability that supports cross-cutting concerns and quality attributes.
- Documented patterns and tools that enable the specification and enforcement of composition rules.
- Integrated tool chains that create required behaviors and assure these behaviors at scale.
- Formalized interactions and non-siloed assurance capabilities supported by integrated tool chains.



Research Focus Areas: Architectural Paradigms

Engineering Societal-Scale Software Systems

This research area leverages the social sciences to develop new software engineering approaches that enable predictable behavior of software systems consisting of people as system components.

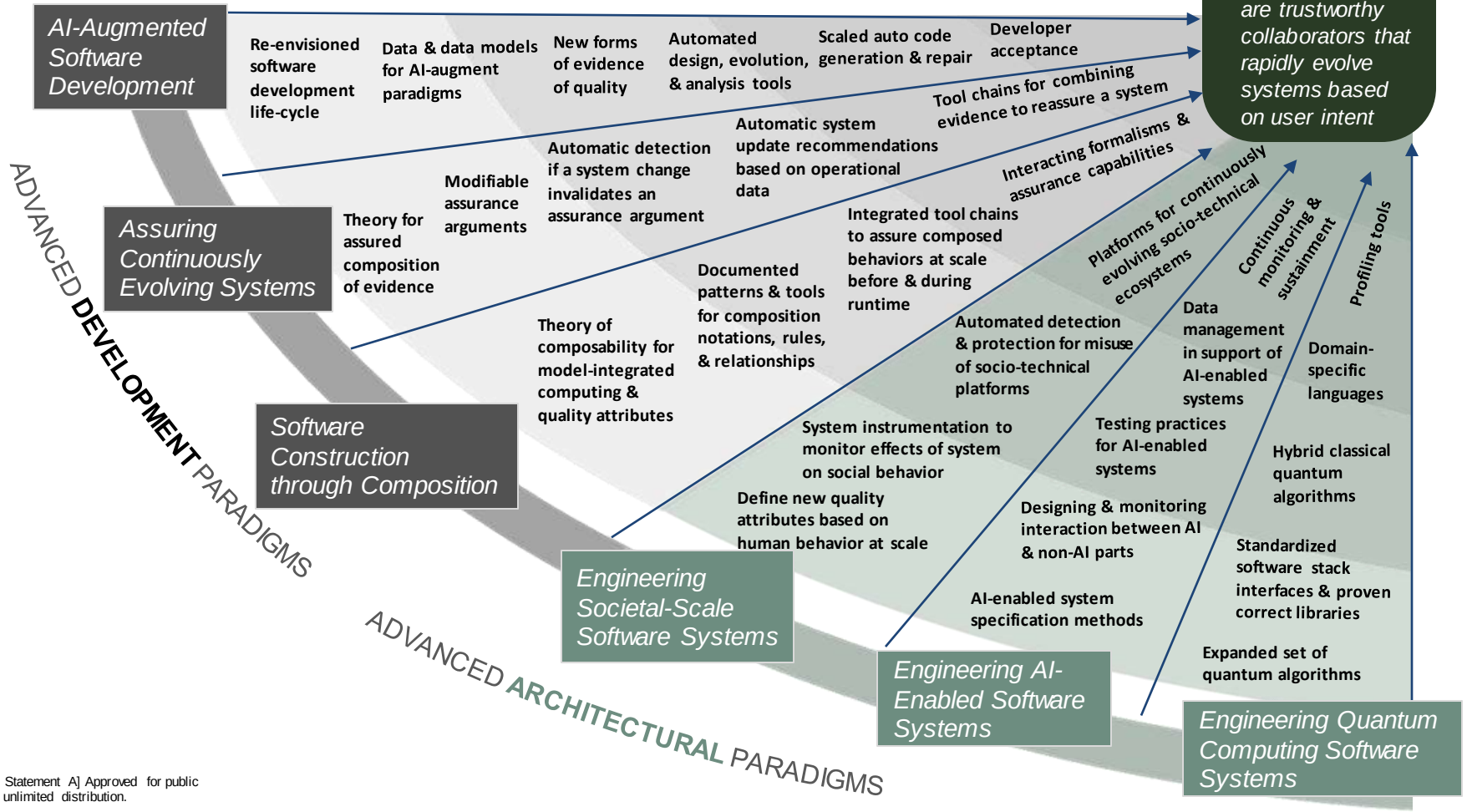
Engineering AI-Enabled Software Systems

This research area focuses on exploring which existing software engineering practices can reliably support the development of AI systems, as well as identifying and augmenting software engineering techniques for systems with AI components.

Engineering Quantum Computing Software Systems

Goals in this research area are to first enable current quantum computers to be easily programmed, and then enable increasing abstraction as larger, fully fault-tolerant quantum computing systems become available.

Software Engineering Research Roadmap with Research Focus Areas and Research Objectives (10-15 Year Horizon)



Findings

1. Software engineering **profoundly impacts all aspects of society** as we increasingly rely on it to provide complex and critical functionality.

2. **Key issues** impacting future directions in software engineering include **smart automation, reassuring evolving systems, understanding composed systems**, and **new system types.**

3. Research focused on **integrating heterogenous systems at ever-larger scales** is needed to support the emergence of new system types.

4. Beyond scale, tomorrow's software challenges will include the need to address **social software considerations** such as transparency, freedom from bias, and privacy.

5. **Academia, government, industry, and research labs** will have critically important and **interacting roles** for the future of software engineering.

6. Significant **research opportunities lie at the intersection** of software engineering and other fields.

7. Software engineering is a distributed activity, taking place worldwide. Technologies are needed to support **seamless integration across computing environments and teams.**

8. **New approaches** are needed to meet growing needs in the **workforce.**

Selected Recommendations

Our goal is to catalyze change that advances software engineering, which in turn will lead to more trustworthy and capable software-reliant systems in the future. Advancing software engineering requires supportive policy, research funding, researchers, practitioners, and cross-fertilization with other research communities.

Accordingly, we make the following recommendations:

1. **Increase investment in software engineering research.**
2. **Initiate software engineering grand challenges.**
3. **Increase software engineering workforce quantity and effectiveness.**

1 **Recommendation:** Increase Investment in software engineering research.

A. Long-term investment to fund core programs in software engineering research is needed to enable new partnerships and advance software engineering technologies to build the systems of the future. Software engineering research is foundational to build the software supply chain including the tools, techniques and software infrastructure to build future software-reliant systems.

i. Funding for software engineering research seems out of step with the criticality of software and with the research challenges to position for the future.

ii. Software industry partnerships with national labs and academic research communities sponsored though government investment enable effective collaboration to solve new problems and provide the long-term progress needed to meet the future needs of software engineering

iii. Research sponsors prioritize funding of software engineering research, including the technical focus areas identified in this study. The technical focus areas are foundational areas that amplify industry research activities, enabling broad advancement in software engineering.

2 **Recommendation:** Initiate software engineering grand challenges.

A. Government sponsorship of 2-3 software engineering grand challenges will further technological innovation and inspire new partnerships to advance software engineering.

   Grand challenges have become an effective way to quickly mobilize existing capability on critical issues while enabling new partnerships across academia and industry.

3 **Recommendation:** Increase software engineering workforce quantity and effectiveness.

A. New approaches are needed for talent and workforce development to significantly expand the scale of software engineering talent required to meet future needs. Multiple actions are necessary to continue to build software engineering capacity, including:

i. continuous education, including some training capability built into systems through digital assistants

ii. research and training on how to maximize automation in software engineering activities to best utilize the time of people

iii. incentives for individuals and organizations to support training and education at all levels, from high school to undergraduate and graduate education

iv. emphasis on creating effective tools for non-software engineering professionals to create automated solutions that are safe, reliable, secure, and resilient

Summary

- New types of systems will continue to push beyond the bounds of what current software engineering theories, tools, and practices can support, therefore **CMU SEI has been leading a study that aims to capture emerging shifts in how software is developed.**
- **Future systems** and **fundamental shifts** in software engineering **require new research focus**.
  - Rapidly deploying software with confidence will be critical for addressing needs and expectations
  - Greater **speed in translating technology into fielded capability** is vital for achieving and maintaining **technological edge.**
- Tomorrow's software challenges will include the need to address new "-ilities": social software considerations such as transparency, freedom from bias, and privacy.
- New approaches are needed for **talent and workforce development** to significantly expand the scale of software engineering talent required to meet future needs.

Discussion