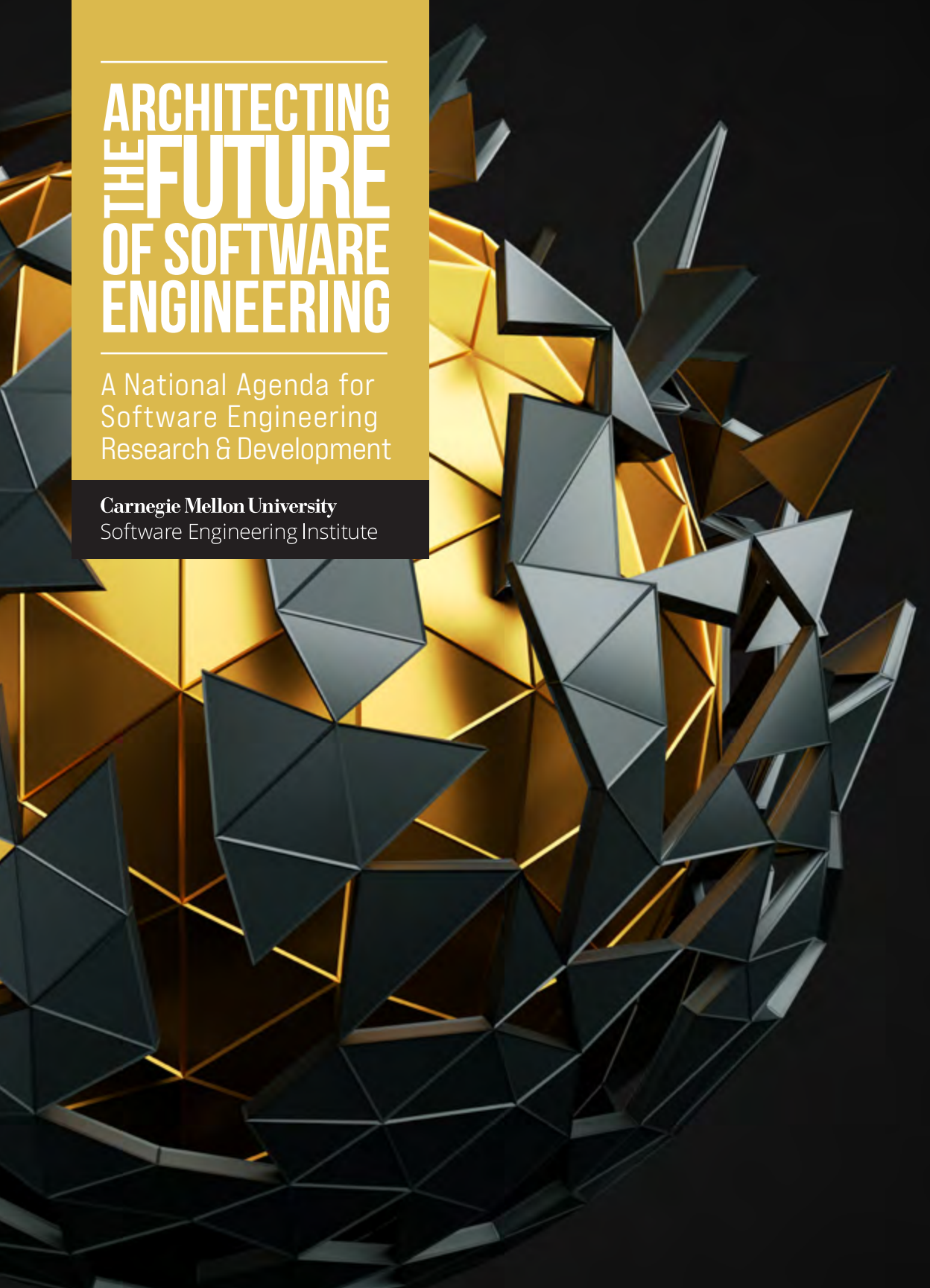

ARCHITECTING THE FUTURE OF SOFTWARE ENGINEERING

A National Agenda for
Software Engineering
Research & Development

Carnegie Mellon University
Software Engineering Institute



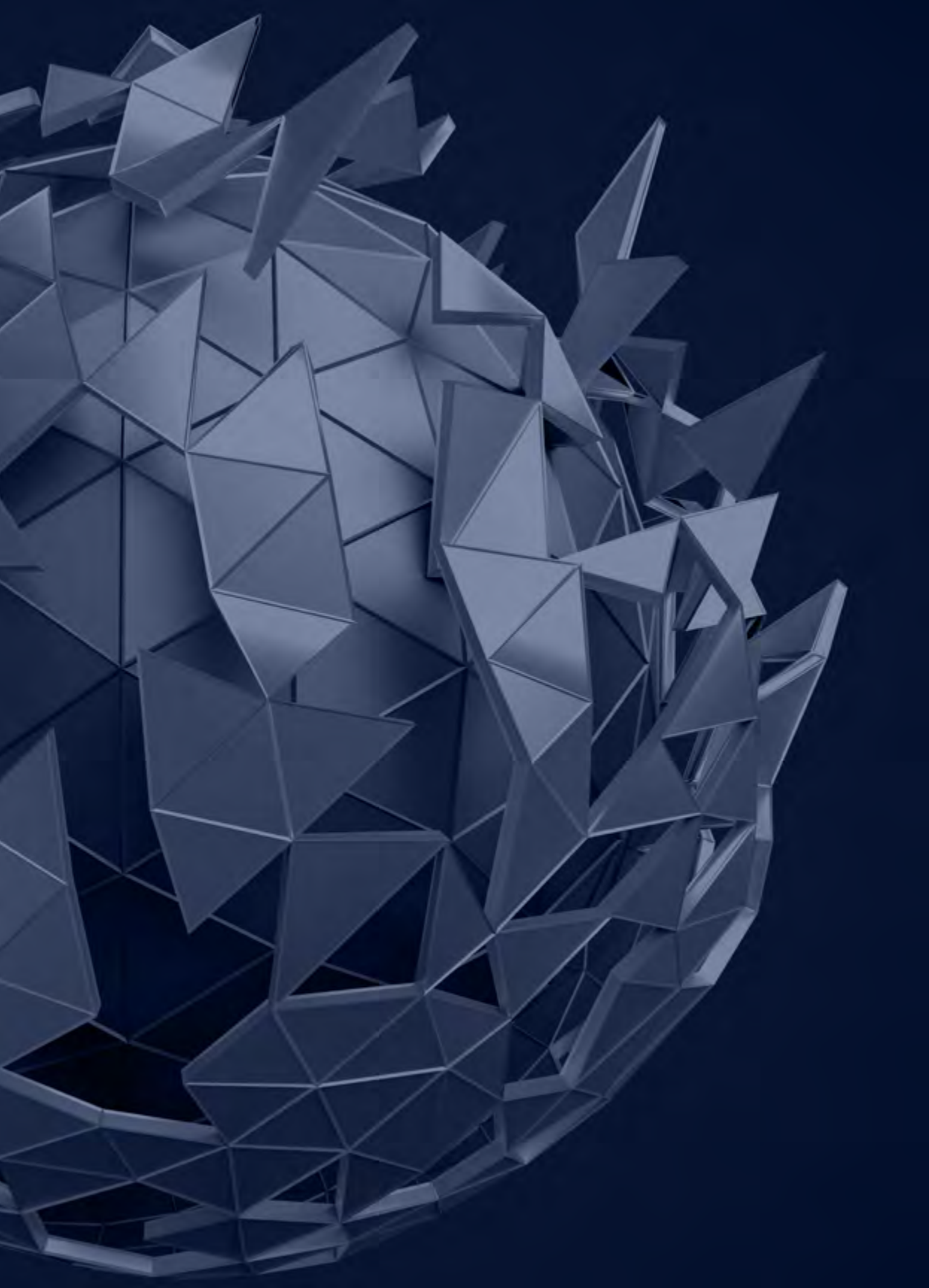


Software is vital to our country's global competitiveness, innovation, and national security. It also ensures our modern standard of living and enables continued advances in defense, infrastructure, healthcare, commerce, education, and entertainment. As part of its work as a federally funded research and development center (FFRDC) focused on applied research to improve the practice of software engineering, the Carnegie Mellon University Software Engineering Institute led the community in creating this multi-year research and development vision and roadmap for engineering next-generation software-reliant systems.



Authors and Contributors

Lead Author Team	Additional Authors (in alphabetical order)	
<p>Anita Carleton, Study Lead Director, Software Solutions Division (SSD), Carnegie Mellon University (CMU) Software Engineering Institute (SEI)</p> <p>Mark Klein Principal Technical Advisor and Principal Researcher, SSD, CMU SEI</p> <p>John Robert Deputy Director, SSD, CMU SEI</p> <p>Erin Harper Strategic Communications Manager, SSD, CMU SEI</p>	<p>Rob Cunningham Vice Chancellor for Research Infrastructure, University of Pittsburgh</p> <p>Dio De Niz Technical Director, Assuring Cyber-Physical Systems, SSD, CMU SEI</p> <p>Ed Desautels Senior Technical Writer & Content Strategist, CMU SEI</p> <p>John Foreman SEI Fellow and Principal Engineer, SSD, CMU SEI</p> <p>John Goodenough SEI Fellow, Principal Researcher, Director's Office, CMU SEI</p>	<p>James Herbsleb Director and Professor, Institute for Software Research, CMU</p> <p>Charles Holland Principal Researcher, SSD, CMU SEI</p> <p>Ipek Ozkaya Technical Director, Engineering Intelligent Software Systems, SSD, CMU SEI</p> <p>Doug Schmidt Cornelius Vanderbilt Professor of Engineering, Vanderbilt University</p> <p>Forrest Shull Lead for Defense Software Acquisition Policy Research, SSD, CMU SEI</p>
Advisory Board		
<p>Deb Frincke, Chair Associate Laboratory Director for National Security Sciences, Oak Ridge National Laboratory</p> <p>Sara Manning Dawson Chief Technology Officer, Enterprise Security, Microsoft</p> <p>Jeff Dexter Senior Director of Flight Software & Cybersecurity, SpaceX</p> <p>Yolanda Gil Director of Knowledge Technologies, Information Sciences Institute, University of Southern California</p>	<p>Vint Cerf Vice President and Chief Internet Evangelist, Google</p> <p>Penny Compton Vice President for Software Systems, Cyber, and Operations, Lockheed Martin Space</p> <p>Tim McBride President, Zoic Labs</p> <p>Michael McQuade Vice President for Research, CMU</p>	<p>Nancy Pendleton Vice President and Senior Chief Engineer for Mission Systems, Payloads and Sensors, Boeing Defense, Space and Security</p> <p>Tim Dare Defense Business Technical Director, Booz Allen Hamilton</p> <p>William Scherlis Director Information Innovation Office, Defense Advanced Research Projects Agency (DARPA)</p>



Foreword: Deb Frincke

Writing a foreword for this report has been both a privilege and a challenge. As the chair of the project's advisory board, I had the opportunity to work with some of the most knowledgeable and passionate individuals I have ever met. The resulting report is important and will be impactful on the future of software engineering. Consequently, it was a privilege to be associated with this work.

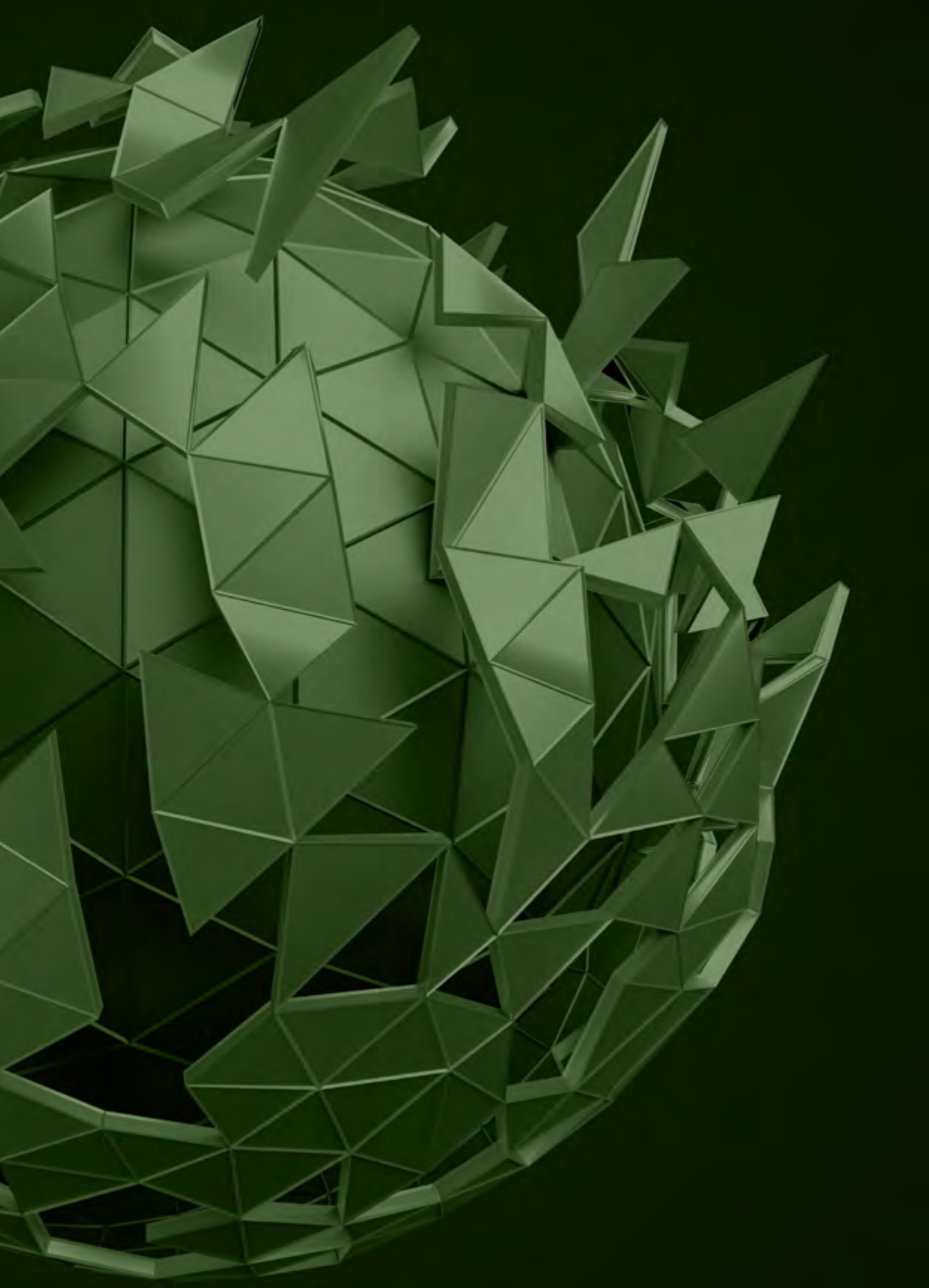
Software, and hence software engineering, problems contributed to the personal challenges I had in writing this foreword because they ate into my scheduled time to write. By chance, I was diverted three times by issues that juxtaposed humans and software-reliant systems. First, I was interrupted by technical challenges arising from ransomware in the context of critical infrastructure protection. Second, I became involved in key practical discussions about how to manage machine learning models that drive important scientific algorithms. And finally, I had to engage in a series of plaintive conversations with my air conditioning repair mechanic because of a software fault that caused my air conditioning to fail during one of the hottest weeks in the year. So while writing, I was actually experiencing the reason that motivated the need for this report: Software inadequacies resulting from inadequate software engineering are truly with us everywhere!

As you read this document, think about how software touches you, and everything around you, and what this implies for the future of software engineering. You will inevitably find that software resilience remains critical, and that software systems have become even more important to our daily lives than ever before. You'll also find that the increasing reliance on societal/global-scale systems highlights even more complexities, such as influence, social manipulation, and other challenges that emerge in these system types. All of this raises the stakes for software engineering.

My hope is that you will find ways to leverage this important report and the insights it contains, and that you will help enact its recommendations. We each have a responsibility to contribute to making software more trustworthy by advocating for investment in advancing the foundations and practice of software engineering.

*Deb Frincke, Associate Laboratory Director for National Security Sciences,
Oak Ridge National Laboratory*

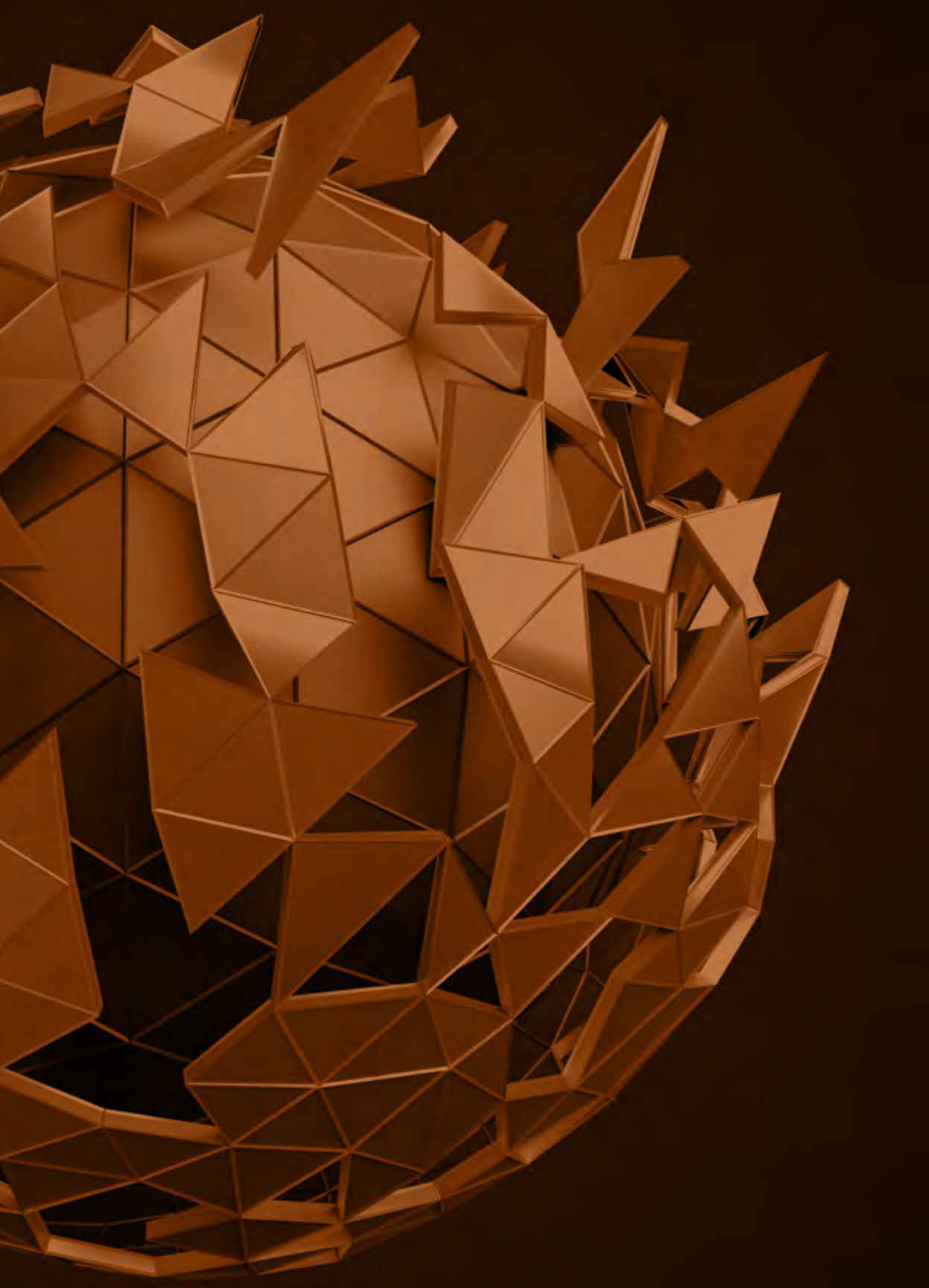
*Advisory Board Chair for the National Agenda for Software Engineering
Research and Development Study*



Foreword: The Honorable Heidi Shyu

Software is an essential, if not the central, part of every Department of Defense (DoD) system. Our hardware has become increasingly programmable, and software has become ubiquitous. Therefore, software engineering is a critical enabler for everything that we do in the DoD. To remain competitive, our weapon systems acquisition must migrate away from the linear development and test cycle and evolve into a rapid continuous update and continuous assurance environment. Consequently, this software engineering technology roadmap is a guide for our research and investment strategy that is vital for our national security. As we develop new systems, we must go beyond model-based software engineering to enable us to rapidly develop systems while reducing re-assurance and sustainment costs. In the future, we will need rapid composition of new capabilities that can operate in a highly contested and denied environment. Integrating heterogeneous systems seamlessly and rapidly will enable us to stay ahead of threats. We will need to exploit the promise of artificial intelligence to increase capability not only in our fielded systems but also in our development systems. This research roadmap should serve as the starting point for a sustained effort to improve software engineering. The DoD will continue to look to the Carnegie Mellon University Software Engineering Institute as a leader in improving the state of the art and practice in software engineering.

*The Honorable Heidi Shyu, Under Secretary of Defense
for Research and Engineering*



Acknowledgments

The world runs on software, and software engineering is the means for enabling the capabilities on which we have come to depend. Although advances in software have emerged incrementally and organically from many sectors and enabled commercial advances that were unimaginable twenty years ago, these current, fundamental piece-parts do not add up to the level of capability that future systems will require. Without a focused effort and continual investment and improvement in critical software engineering knowledge, technologies, and foundational software engineering research, next-generation applications may simply not be possible. Consequently, we felt it was imperative to orchestrate the creation of a National Agenda for Software Engineering Study to identify which technologies and areas of research are most critical for enabling future systems. The resulting roadmap is intended to guide the research efforts of the software engineering community. As we developed this roadmap, we asked ourselves, “How do we ensure that future software systems will be safe, predictable, and evolvable?”

With that brief introduction to our study as a backdrop, I would like to acknowledge the principal team of authors: Mark Klein, John Robert, Erin Harper, Rob Cunningham, Dio De Niz, Ed Desautels, John Foreman, John Goodenough, Charlie Holland, Ipek Ozkaya, and Forrest Shull, all from the Carnegie Mellon University Software Engineering Institute (SEI), along with James Herbsleb from the Carnegie Mellon University Institute for Software Research and Douglas Schmidt from Vanderbilt University. I am grateful for the opportunity to collaborate with this fabulous team who worked with passion, creativity, and determination to devise a compelling, thoughtful, and inspiring research roadmap for the future of software engineering.

It is interesting to note that this study was performed entirely during the global COVID-19 pandemic. That means that every part of the study was accomplished in a virtual environment: from designing the study and meeting with our advisory board, to the workshops we held to engage with the software engineering research communities, to working with our distributed team to assemble the study, all of it had to be done in a virtual setting. I want to thank everyone for their commitment to making time for this study and for finding creative ways to overcome the communication barriers and have meaningful conversations that contributed to this important topic.

Our team has appreciated the opportunity to work with senior thought leaders and luminaries in the field on our advisory board. It's been very helpful to have the breadth and depth of representation from different parts of the community on our board, including representatives from the Department of Defense (DoD), national labs, defense industrial base organizations, tech organizations, and academic leaders in computer science. We've been grateful for their enthusiastic participation and guidance along the way. In the early part of the study, they were instrumental in advising us on the research focus areas and helping us connect with the right people to work with on the study. Going forward, their attention has shifted to helping us think about who needs to know about this study and how we can enact our roadmap. We were most fortunate to have Dr. Deb Frincke as our advisory board chair. She demonstrated amazing leadership and commitment to this study because of her profound understanding of the critical importance of software engineering. The stellar advisory board included Vint Cerf, Penny Compton, Tim Dare, Sara Manning Dawson, Jeff Dexter, Yolanda Gil, Tim McBride, Michael McQuade, Nancy Pendleton, and William Scherlis. They were deeply committed to thinking about the future of software engineering and provided inspiring and thoughtful guidance throughout the study.

Next, our sincere thanks go to the many individuals who took time from their busy schedules to participate in or lead one of our virtual workshops. Work of this kind would be impossible without their willingness to share their experiences and ideas for the benefit of the software engineering community. We specifically want to thank Michele Falce, who provided critical ideas and contributions to enable all of the virtual workshops. We also thank the leaders and facilitators of each the workshops, including the following:

- National Agenda for Software Engineering R&D Workshop: Software Engineering Researcher Edition
Keith Webster (CMU) and Barbora Batokova
- National Agenda for Software Engineering R&D Workshop: Voice of the Customer Workshop
Harold Ennulat and Natalie Chronister
- Future Scenarios Workshop: Developing Plausible Alternative Futures
Keith Webster (CMU)
- National Agenda for Software Engineering R&D Workshop: DoD Senior Leaders Workshop
John Robert
- Software Engineering Grand Challenges and Future Visions Workshop
Forrest Shull, Sandeep Neema (Defense Acquisition Research Projects Agency), Sol Greenspan (NSF), Christopher Ré (Stanford AI Lab)

Special thanks also go to the following people who shared their deep knowledge of the field in our expert interviews: Bob Bonneau, Penny Compton, Rob Cunningham, Tim Dare, Dio De Niz, Jeff Dexter, Deb Frincke, Yolanda Gil, John Goodenough, Jim Herbsleb, James Ivers, Grace Lewis, Ruben Martins, Michael McQuade, Ipek Ozkaya, Nancy Pendleton, Dan Plakosh, Bill Scherlis, Doug Schmidt, Mary Shaw, Eileen Wrubel, Hasan Yasar, and Robin Yeman. Jennifer Hykes and Marc Novakowski were also gracious enough to share their imaginative ideas for our section on future scenarios.

We would also like to thank our SEI colleagues in the communication, design, and production teams for their important roles in writing, editing, creating graphics, and web production. We especially thank Cat Zaccardi for her design team leadership and creativity, David Biber for his gorgeous visuals and page design, Donald Kurt Hess for his beautiful graphics, and Mike Duda for his print expertise and alacrity.

And most importantly, all of the authors of this study would like to share their sincere gratitude for the critical support, sponsorship, and contributions of the SEI Director's Office, including Dr. Paul Nielsen, Director and CEO; Mr. Dave Thompson, Deputy Director and Chief Operating Officer; and Dr. Tom Longstaff, Chief Technology Officer. They understood from the very beginning how challenging this activity would be, but also recognized what a critical contribution it would make to advancing the field of software engineering.

And finally, it has been a privilege for us to talk to so many software leaders with industry, academia, and government perspectives. Without exception, the discussions reaffirmed the critical importance of advancing software engineering for national competitiveness and meeting the increasing expectations of software across the globe. Recent news headlines highlight current software engineering limitations and are early indicators that software engineering is unprepared for the even greater challenges ahead. With your help, this roadmap will bring about a new era of multidisciplinary research and new partnerships to prepare us for those challenges and enable ongoing community discussion to advance the discipline of software engineering.

*Anita Carleton, Software Solutions Division Director,
Carnegie Mellon University Software Engineering Institute*

Executive Summary

Software Engineering as a Strategic Advantage

We live in an age of software-enabled transformation. Software, and all of the software engineering processes, practices, technologies, and the scientific domains that support it, makes our world-class healthcare, defense, commerce, communication, education, and energy systems possible. It is also a key enabling component in nearly every area of research, such as smart infrastructure (nanotech), human augmentation (biotech), and autonomous transportation. Our dependence on software, however, makes us vulnerable to its weaknesses. Software weaknesses are a direct reflection of inadequacies in the state of the art and practice of software engineering, and they can affect millions of people without warning. Just recently, software issues caused the largest shut-down of an oil pipeline in U.S. history and allowed attacks that paralyzed hundreds of businesses on five continents [Satter 2021]. Software quality problems have also led to loss of life in plane and car crashes, and expensive failures in the space flight industry [Rhee 2020; CBS 2010].

Without a catalyst for investing in software engineering, the situation will worsen as we increasingly depend on ever larger and more complex software-reliant systems. This report is intended to be such a catalyst. Identifying the critical technologies and areas of research that will enable future systems and laying out a roadmap to guide research efforts is a crucial step toward making software a competitive advantage. This study outlines efforts intended to make future software systems safe, predictable, and evolvable. The Carnegie Mellon University Software Engineering Institute (CMU SEI) engaged the software engineering community and assembled an advisory board of visionaries and senior thought leaders to ensure that the views of the broad software engineering ecosystem were represented in this multi-year research and development vision and roadmap.

Findings Reflect New Learnings, Challenges, and Research Needs

Without exception, the work that we surveyed for this study points to software engineering research as a highly dynamic, fast-moving field where technologies can arise quickly and grow to become integral parts of the infrastructure of modern life. While that is perhaps unsurprising, the extent to which recent technology trends are coming together and allowing the emergence of capabilities with both speed and quality is remarkable. Many of these technologies and capabilities were unimaginable even 10 years ago.

The following findings were derived from the state of software engineering practice, new trends and emerging technologies that will help to advance the state of software engineering practice, workshops held with software engineering research communities, a literature survey, interviews with experts in the field, and input from our advisory board. They summarize key learnings, key challenges, and new research needed for the future of software engineering.

1. **Maintaining national software engineering proficiency is a strategic advantage.** Software engineering affects everything because software is everywhere, including in our nation's infrastructure, defense, financial, education, and healthcare systems. Our ever-growing dependence on software systems makes it imperative to maintain our nation's leadership and strategic advantage in software engineering. We need to raise the visibility of software engineering to the point where it receives the sustained recognition and investment commensurate with its importance to national security and competitiveness.
2. **Maintaining national software engineering proficiency requires sustained research.** New types of systems will continue to push beyond the bounds of what current software engineering theories, tools, and practices can support. Future systems and fundamental shifts in software engineering require new research focus in areas including smart automation, reassuring evolving systems, understanding composed systems, and new system types, such as AI-enabled systems, societal-scale systems, and quantum systems.
3. **Maintaining national software engineering proficiency requires fostering strategic partnerships.** We will need to enable strategic partnerships and collaborations to drive innovation in software engineering research among industry, research laboratories, academia, and government.
4. **Maintaining national software engineering proficiency requires sustained investment.** Policy makers must recognize the benefits of software engineering and make it a critical national capability. Such recognition would imply a sustained investment strategy.
5. **The vision of software engineering needs to change.** The current notion of a software development pipeline will be replaced by one where AI and humans collaborate to continuously evolve the system based on programmer intent.
6. **Focusing on re-assuring systems will enable continuous and rapid incorporation of new capability.** Because software is ubiquitous, there is an ongoing and increasing need for software to continuously evolve to incorporate new capability. We therefore need to understand how to continuously re-assure software reliant systems efficiently without doing harm to existing capability. Elevating the importance of assurance evidence and assurance arguments will be key.

7. New design principles are needed for societal-scale systems.

The growing recognition of software’s impact is generating new quality attribute requirements for which software engineers will need to develop better design approaches. In addition to the traditional ones (modifiability, reliability, performance, etc.), there is a need to add a roster of new quality attributes like transparency, influence, and so forth.

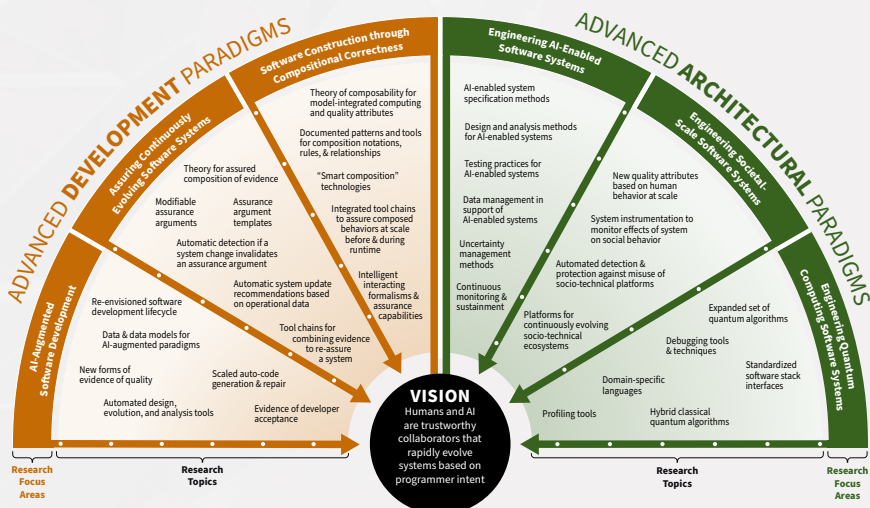
8. The software engineering workforce needs to be (re-)conceived.

Software-reliant systems are built for many different purposes by a broad collection of people with very disparate skill sets, many of whom do not have formal software engineering training. We need to better understand the nature of the needed workforce and what to do to foster its growth.

A Guiding Vision and Roadmap for the Future of Software Engineering

Our guiding vision, as described in our findings, is one in which the current notion of the software development pipeline is replaced by one where humans and software are trustworthy collaborators that rapidly evolve systems based on programmer intent. To achieve this vision, we anticipate the need for new development and architectural paradigms for engineering future systems.

Our study helped to inform new areas of research that must be met to advance software engineering for future systems. In close collaboration with our advisory board and other leaders in the software engineering research community, we developed a research roadmap with six research focus areas. The following figure shows those areas along with a list of research topics to undertake, and then short descriptions of each of the research focus areas follow. A larger version of this figure appears on the foldout after page 26.



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

AI-Augmented Software Development. At almost every stage of the software development process, AI holds the promise of assisting humans. By relieving humans of tedious tasks, they will be better able to focus on tasks that require the creativity and innovation that only humans can provide. To reach this important goal, we need to re-envision the entire software development process with increased AI and automation tool support for developers. A key challenge will be taking advantage of the data generated throughout the lifecycle. The focus of this research area is on what AI-augmented software development will look like at each stage of the development process and during continuous evolution, where AI will be particularly useful in taking on routine tasks.

Assuring Continuously Evolving Software Systems. When we consider the software-reliant systems of today, we see that they are not static (or even infrequently updated) engineering artifacts. Instead, they are fluid—meaning that they are expected to undergo almost continuous updates and improvements and be shown to still work. The goal of this research area is, therefore, to develop a theory and practice of rapid and assured software evolution that enables efficient and bounded re-assurance of continuously evolving systems.

Software Construction through Compositional Correctness. As the scope and scale of software-reliant systems continues to grow and change continuously, the complexity of these systems makes it unrealistic for any one person or group to understand the entire system. It is therefore necessary to integrate (and continually re-integrate) software-reliant systems using technologies and platforms that support the composition of modular components. This is particularly difficult since many of such components are reused from existing elements that were not designed to be integrated or evolved together. The goal of this research area is to create methods and tools that enable the specification and enforcement of composition rules that allow (1) the creation of required behaviors (both functionality and quality attributes) and (2) the assurance of these behaviors.

Engineering AI-Enabled Software Systems. AI-enabled systems, which are software-reliant systems that include AI and non-AI components, have some inherently different characteristics than those without AI. However, AI-enabled systems are, above all, a type of software system. These systems share many parallels with the development and sustainment of more conventional software-reliant 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 the specification, design, architecture, analysis, deployment, and sustainment of systems with AI components.

Engineering Socio-Technical Systems. Societal-scale software systems, such as today's commercial social media systems, are designed to keep users engaged and often to influence them. A key challenge in engineering societal-scale systems is predicting outcomes of the socially inspired quality attributes that arise when humans are integral components of the system. The goal is to leverage insights from the social sciences to build and evolve societal-scale software systems that consider these attributes.

Engineering Quantum Computing Software Systems. Advances in software engineering for quantum are as important as the hardware advances. The goals of this research area are to first enable current quantum computers to be programmed more easily and reliably, and then enable increasing abstraction as larger, fully fault-tolerant quantum computing systems become available. A key challenge is to, eventually, fully integrate these types of systems into a unified classical and quantum software development lifecycle.

Research and Enactment Recommendations Catalyze Change

Catalyzing change that advances software engineering will lead to more trustworthy and capable software-reliant systems. The research focus areas shown in the roadmap graphic previewed earlier in this section and on foldout following page 25 led to a set of research recommendations that are necessary to catalyze change, which are followed by enactment recommendations that focus on people, investment, and sustainment are needed.

The following research recommendations address challenges such as the increasing use of AI, assuring changing systems, composing and re-composing systems, and engineering socio-technical and heterogenous systems.

1. **Enable AI as a reliable system capability enhancer.** The software engineering and AI communities should join forces to develop a discipline of AI engineering. This should enable the development and evolution of AI-enabled software systems that behave as intended and enable AI to be used as a software engineering workforce multiplier.
2. **Develop a theory and practice for software evolution and re-assurance at scale.** The software engineering research community should develop a theory and associated practices for re-assuring continuously evolving software systems. A focal point for this research is an assurance argument, which should be a software engineering artifact equal in importance to a system's architecture, that ensures small system changes only require incremental re-assurance.

3. **Develop formal semantics for composition technology.** The computer science community should focus on the newest generation of composition technology to ensure that technologies such as dependency-injection frameworks preserve semantics through the various levels of abstraction that specify system behavior. This will allow us to reap the benefits of development by composition while achieving predictable runtime behavior.
4. **Mature the engineering of societal-scale socio-technical systems.** The software engineering community should collaborate with social science communities to develop engineering principles for socio-technical systems. Theories and techniques from disciplines such as sociology and psychology should be used to discover new design principles for socio-technical systems, which in turn should result in more predictable behavior from societal-scale systems.
5. **Catalyze increased attention on engineering for new computational models, with a focus on quantum-enabled software systems.** The software engineering community should collaborate with the quantum computing community to anticipate new architectural paradigms for quantum-enabled computing systems. The focus should be on understanding how the quantum computational model affects all layers of the software stack.

The above recommendations focused on scientific and engineering barriers to achieving change. The following enactment recommendations focus on institutional obstacles, including economic, human, and policy barriers.

6. **Ensure investment priority reflects the importance of software engineering as a critical national capability.** The strategic role of software engineering in national security and global market competitiveness should be reflected in national research activities, including those undertaken by the U.S. White House Office of Science and Technology Policy (OSTP) and Networking and Information Technology Research and Development (NITRD). These research activities should recognize software engineering research as an investment priority on par with chip manufacturing and AI with benefits to national competitiveness and security. Software engineering grand challenges sponsored by DARPA, the National Science Foundation (NSF), and FFRDCs are also suggested.

7. **Institutionalize ongoing advancement of software engineering research.**

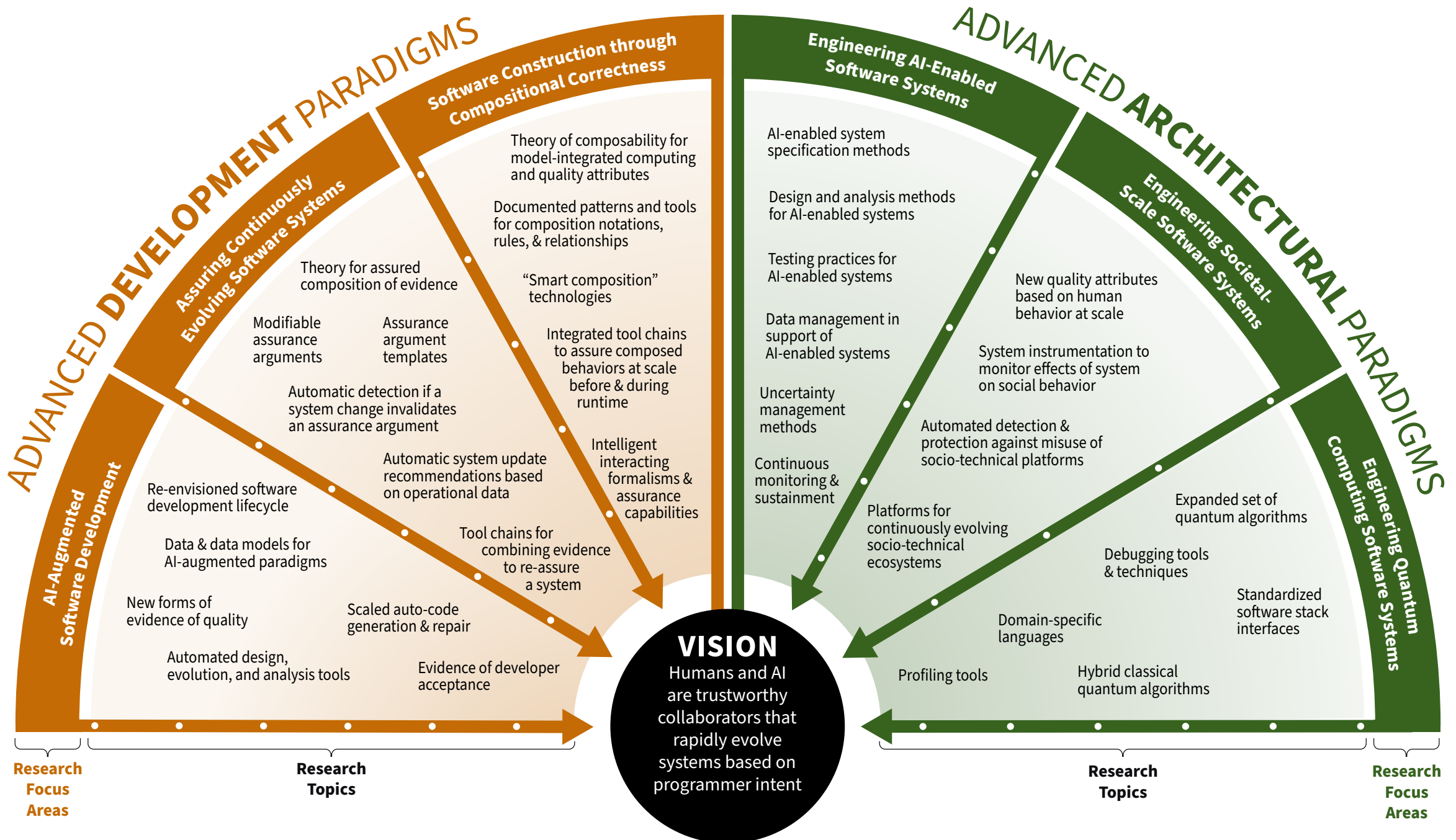
Sustained advancements in software engineering requires institutionalizing an ongoing review and reinvestment cycle for software engineering research and its impact on software engineering practice. Maintaining national software engineering proficiency requires research funding sources and institutes working with industry and government leaders in the software engineering community to periodically review the state of software engineering.

8. **Develop a strategy for ensuring an effective workforce for the future of software engineering.** Currently, software engineering is performed by a broad collection of people with an interdisciplinary skill set not always including formal training in software engineering. Moreover, the nature of software engineering seems to be changing in reaction to the fluid nature of software-reliant systems. We need to better understand the nature of the needed workforce and what to do to foster its growth. The software engineering community, software industry, and academic community should create a strategy for ensuring an effective future software engineering workforce.

Architecting Future Systems Requires Software Engineering Advances

Due to the conceptual nature of software, it continues to grow, without bounds, in capability, complexity, and interconnection. There seems to be no plateau in the advancement of software. To make future software systems safe, predictable, and evolvable, the software engineering community—with sufficient investment from private and public sources—must work together to advance the theory and practice of software engineering strategically to enable the next generation of software-reliant systems.

Software Engineering Research Roadmap with Focus Areas and Research Objectives [10–15 Year Horizon]



Copyright 2021 Carnegie Mellon University.

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.

The view, opinions, and/or findings contained in this material are those of the author(s) and should not be construed as an official Government position, policy, or decision, unless designated by other documentation.

NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN "AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

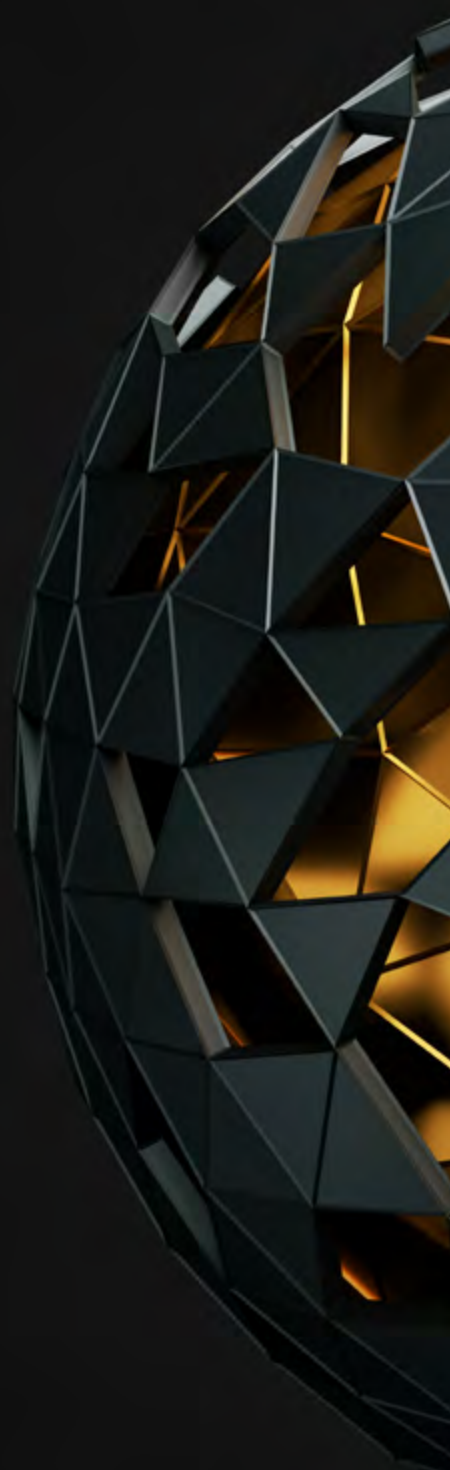
[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution. Please see Copyright notice for non-US Government use and distribution.

Internal use:* Permission to reproduce this material and to prepare derivative works from this material for internal use is granted, provided the copyright and "No Warranty" statements are included with all reproductions and derivative works.

External use:* This material may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other external and/or commercial use. Requests for permission should be directed to the Software Engineering Institute at permission@sei.cmu.edu.

* These restrictions do not apply to U.S. government entities.

DM21-0889



About Us

The Software Engineering Institute (SEI) at Carnegie Mellon University is a Federally Funded Research and Development Center (FFRDC)—a nonprofit, public–private partnership that conducts research for the United States government. One of only 10 FFRDCs sponsored by the U.S. Department of Defense (DoD), the SEI conducts R&D in software engineering, systems engineering, cybersecurity, and many other areas of computing, working to introduce private-sector innovations into government.

As the only FFRDC sponsored by the DoD that is also authorized to work with organizations outside of the DoD, the SEI is unique. We work with partners throughout the U.S. government, the private sector, and academia. These partnerships enable us to take innovations from concept to practice, closing the gap between research and use.

Contact Us

Carnegie Mellon University
Software Engineering Institute
4500 Fifth Avenue
Pittsburgh, PA 15213-2612

412.268.5800 | 888.201.4479
sei.cmu.edu | info@sei.cmu.edu