Software in FP10 – New Directions for Software R&I

Introduction

Digital technologies, including hardware, software, and tools for managing data, have had a widespread and profound impact on the world, and continue to transform business and society. Global networks support pervasive connectivity and have enabled a service-based model for users to access a wide range of sophisticated functionality. Research and innovation (R&I) in recent years has been successful in advancing the state-of-the-art in software and services.

There is now an increasing focus on applications of digital technologies. The Digital Europe Programme identifies five crucial areas where there is a gap between digital technology research and market deployment: supercomputing, artificial intelligence (AI), cybersecurity, semiconductors, and advanced digital skills. Software is at the heart of most of these areas. The last few years have seen new technologies, most notably AI, starting to disrupt established ways of creating and using software. R&I in software technologies is required to complement more application-focused topics.

This paper aims to identify the context for future R&I in software and services, with a focus on technology implications and requirements. There are also significant policy implications, but these are not explicitly addressed here.

Applications of Digital Technology

Technology advances over recent decades have led to pervasive digitisation and automation of many established processes, and enabled completely new ways of working, living, and managing critical infrastructure and resources. Improved information flows and automation have delivered efficiency gains in many areas.

The result has been that many critical systems (e.g. in energy, water, industry, finance, healthcare, transport) are now dependent on digital technologies. The underlying digital infrastructure is complex, and its management is distributed across many separate organisations. The consequences of failure (whether accidental or malicious) are becoming increasingly serious.

There is an expectation that increased levels of integration, automation, and intelligence can help to address global challenges of productivity, sustainability, and quality of life. However, this may further increase the complexity and vulnerability of the systems we all rely on. Technology development and deployment need to take account of this and provide solutions that are suitable for the environment in which they are used.

Innovations in Computing Technology

We are now in a period of rapid change in computing technology. New directions in hardware and software show great promise with huge potential. Examples in hardware include:

- Heterogeneous computing, including the "cloud continuum" and the use of accelerators such as GPUs, TPUs, ASICs and FPGAs
- Quantum computers are potentially able to address problems inaccessible to current computer systems; they will need to be integrated with "classical" computer systems

- Neuromorphic computing, which may enable new approaches to AI, including improved energy efficiency

The most significant current trend in software is in AI (and large language models, LLMs, in particular), with huge resources being devoted to the development of new models and applications. The prospect is of autonomous agents and autonomous systems everywhere, developed and operated with a diminishing need for human involvement. In addition, the demands of AI are impacting the supply chain for computing hardware (notably DRAM) leading to a shift in thinking from “RAM is cheap and abundant” to “RAM is scarce and unpredictable.” This illustrates a more general emerging risk to the ability to sustain complex software applications. Design with an emphasis on efficient use of resources (RAM, accelerators, energy, water etc) is now becoming a priority.

There is strong pressure to bring these new technologies to market, and to avoid falling behind in a global race to realize their potential. However, many components are relatively immature and there are risks in adding to the complexity of systems (already complex and incompletely understood) in which they are deployed.

There is a pressing need for sound engineering practices in design, development and operations. Although the current state-of-the-art in software engineering can provide a sound starting point, new tools, techniques and processes need to be developed in parallel to ensure that the transition from prototype to production can be made safely. General techniques supporting the integration of any new computing technology into existing systems should be targeted.

Future of Software

The overall aim of R&I in digital technologies in the coming years is to develop the ability to build systems which can deliver on key priority areas - green growth and sustainability, climate resilience, healthcare, and mobility. We face significant challenges in these areas and effective use of new technologies may be essential in addressing them.

Principles associated with traditional engineering (fail-safe operation, graceful degradation etc) are often not fully embedded in software engineering for consumer and enterprise software. Established software engineering practices for developing, testing, deploying, and operating digital systems are arguably generally sufficient to meet today's requirements but we are moving towards a very different environment.

Significant investment in AI by both private and public sector is leading to rapid deployment in many areas although the details, costs and benefits are not always well understood. New software engineering solutions are urgently required to support the safe, manageable adoption of AI. This needs to cover the whole lifecycle: from application concept to operation.

AI tools are increasingly being used in the development and maintenance of software systems. There is an inherent uncertainty in their outputs which means that skilled human oversight is required. As the complexity of the tasks entrusted to AI grows, so does the challenge of assuring that software is dependable and trustworthy.

Central to this is the need to increase resilience to software errors and bugs, misconfigurations or malicious attack throughout the full software lifecycle.

Development

Rapid advances in AI are encouraging developers to replace traditional coding with the use of sophisticated tools which can generate code automatically. The capabilities of these tools are rapidly improving and for

some applications (e.g. pattern recognition, language translation) they can already produce solutions which go beyond what human developers can achieve.

AI assisted code generation (or “vibe coding”) is already empowering people to solve their own computing-related problems without having to rely on "programmers". This continues the trend exemplified by the use of spreadsheets in business and is already yielding significant benefits to individual productivity as people learn how best to use new tools. The human user needs to ensure that the generated software is fit for their own purposes. If the consequences of failure are significant, the balance of liability between human user and AI tools needs to be addressed.

The real potential value that AI can bring is in software for long-lived systems which affect many people (e.g. business systems, operation of critical infrastructure), but this inevitably comes with increased consequences of failure and more complex questions of liability. In the development phase, defining exactly how each software artefact should behave – individually and as part of a larger system – is critical.

Currently requirements in AI assisted code generation are typically specified with a mixture of prompts, instructions, and context. Implicit in this approach is the changing role of the developer. Rather than explicitly defining the behaviour of software by writing code in some programming language, the developer needs to express intent in some other way. This puts a greater emphasis on formulating precise requirements, particularly as complexity increases and manual checking becomes infeasible. One area where AI support could add significant value is in compliance-by-design: automatically ensuring that generated software follows regulatory and legal requirements and removing this responsibility from the developer.

System architectures and programming languages are very useful for people. They provide a common conceptual framework which allows different people to reason about system behaviour. This is essential for human review of code, understanding how the system works, and maintenance over time. However, it imposes constraints on the solution. AI code generation may be able to produce software that better meets the design goals by abandoning these constraints. This software may work, while not being comprehensible to a human. There is a balance between certainty and creativity which raises the question of the nature of human-AI collaboration in software development. The ideal may be for people and AI agents to work together as integral members of the development team. Achieving this requires new skills, and advances in human-computer interaction.

Testing

Quality control of AI-generated software is essential, particularly where the consequences of incorrect or unexpected behaviour are serious. Conventional techniques and processes for verification and validation will be insufficient. For example, AI-generated code may not be written for clarity (possibly not even use a human-readable programming language), making human code reviews and white-box testing difficult.

Testing based on an external view of the behaviour of the software (black-box testing) is possible if specifications and requirements are comprehensive and rigorous. This indicates a need for model-driven approaches and the integration of formal methods to assure correct behaviour.

New tools and techniques (including the use of AI in testing) and the development of skilled practitioners to use them effectively are required.

Deployment

Production of a complex software artefact involves many components, libraries, tools and processes. These may come from a variety of sources, constituting a software supply chain. Any component of a system can

have vulnerabilities. When software is deployed, a defensive approach is required, limiting the impact and propagation of any threat.

Use of AI to generate code adds complexity to the supply chain associated with the model and its training data. Techniques including attestation and comprehensive management of provenance need to be developed.

Operation

Operation of complex systems requires a high degree of automation is an obvious area where AI systems can have an impact. Pervasive instrumentation and monitoring can provide the context for AI agents to make decisions in support of specified goals. The formulation of these goals and their use in training AI models raises some important research challenges. Dynamic adaptation of models may be required to deal with environment drift.

Detection and resolution of conflicts between independent agents is important, and there is the possibility of undesirable emergent behaviour. Allowing control and management to be fully automatic requires a very high level of trust and having a skilled human-in- (or on)-the-loop, supported with appropriate management systems is required. The ability of humans to understand the context, reasoning, and behaviour of automated systems is crucial.

Recommendations

Future R&I in software needs to address several priority topics, with a particular focus on critical infrastructure and systems:

- Management of software systems as integral parts of systems in and across different domains which directly affect people (e.g. healthcare, energy, manufacturing, transport). Risks and potential harms need to be assessed and managed, especially considering the software system's interactions with, and effect on its socio-technical deployment context, intended use and reasonably foreseeable misuse. Innovation in every domain needs to fully address relevant software aspects to ensure dependability.
- AI-native software engineering techniques and tools to ensure software is robust, secure, safe and fit for purpose, and addresses the full lifecycle of systems. This must recognize both that AI is used to create software, and that software is used to create new forms of AI.
- Engineering methods and tools that assure (and evidence) secure-by-design, safe, compliant and privacy-preserving operation in many and varied environments, each with their own stakeholders, purposes, constraints, regulations, priorities and risks. Facilities for appropriate and effective human oversight and accountability are essential.
- Development of libraries and frameworks focused on significantly improving resource efficiency (computing hardware, networks, energy, water) across heterogeneous computing environments (edge to cloud). Enhanced monitoring and control are required to enable fine-grained observability and support graceful degradation in unpredictable and dynamic environments.
- End-to-end security facilities (including quantum-resistant and post-quantum cryptography) to support confidential and privacy-preserving computing on untrusted edge and cloud systems.
- Ensuring that human skills are maintained and developed so that effective application of AI technologies is broadly accessible, including to SMEs. Effective education and training of current and future practitioners is essential to achieve and maintain sovereignty in the face of the anticipated radical changes in software and services.