World Transactions on Engineering and Technology Education
Vol.2, No.3, 2003

Pervasive computing reforming software engineering education

Gilda Pour

San José State University
San José, United States of America

ABSTRACT: Pervasive computing has emerged as the next generation of computing. It envisions computing environments focusing on people, rather than machines, as it has been the case for over 40 years. It is envisaged that the boundaries between hardware and software will disappear. Today's distributed and mobile computing's goal of *anytime, anywhere* connectivity will be extended to the pervasive computing's goal of *all the time, everywhere* connectivity. Pervasive computing encompasses many areas of computer science and engineering, such as intelligent systems, agent technologies, mobile computing, wireless networks, distributed systems, middleware, wearable and context-aware computing, and device technology. The article discusses the necessity of reform in software engineering education to facilitate moving pervasive computing from the laboratories into the real world. The article presents key strategies for the education reform efforts. These strategies include infusing the software engineering curriculum with a set of new topics, integrating research into education, building a solid foundation for both multidisciplinary work and life-long learning, and establishing industry-academic partnerships in research and education.

INTRODUCTION

In 1991, Mark Weiser, then chief of technology officer for Xerox's Palo Alto Research Centre, described a vision for 21st Century computing: ubiquitous computing (a.k.a. pervasive computing). The opening statement in his seminal paper is: *The most profound technologies are those that disappear. They weave themselves into the fabric of everyday life until they are indistinguishable from it* [1]. He articulated his vision further:

> *There is more information available at our fingertips during a walk in the woods than in any computer system, yet people find a walk among trees relaxing and computers frustrating. Machines that fit the human environment instead of forcing humans to enter theirs will make using a computer as refreshing as taking a walk in the woods* [1].

Don Norman also emphasised this concept in his 1998 book entitled *The Invisible Computer* [2].

Michael Dertouzos, director of the MIT's Laboratory for Computer Science for more than 25 years, described the ultimate goal as *breaking away from our 40-year machine preoccupation to a new era of people-oriented computing.* The MIT's Oxygen project aims to help people do more by doing less [9]. This ongoing project has focused on the development of a new infrastructure for information technologies (the Oxygen system) to bring abundant communication and computation, as pervasive and free as air naturally into people's lives. Dertouzos wrote:

> *The first three socioeconomic revolutions were all based on things: the plow for the agrarian revolution, the motor for the industrial revolution, and the computer for the information revolution. Perhaps the time has come for the world to consider a fourth revolution – aimed no longer at objects but at understanding the most precious resource on earth – ourselves* [3].

MOVING TOWARDS PERVASIVE COMPUTING

Pervasive computing presumes a vision that is based upon disappearing computing hardware and software into the background, becoming totally transparent to users. Pervasive computing environments will no longer be virtual environments for storing and running software systems as do today's computing environment. In the pervasive computing world, computing systems will no longer be machines that run programs in virtual environments as do today's computing systems. Software applications will no longer be written to exploit devices' capabilities as do today's software applications. Devices will no longer be reactive, and managed by users as do today's devices.

The distributed and mobile computing's goal of *anytime, anywhere* connectivity will be extended to the pervasive computing's goal of *all the time, everywhere* connectivity. This calls for revamping and integrating several technologies, such as device, network, software, user and perceptual technologies in order to support abundant services that are proactive, invisible, interoperable, intelligent, mobile and secure.

To advance pervasive computing vision towards technical and economic viability, a *multidisciplinary approach* has to be adopted to address the issue of proliferation of computational resources in the physical world. The successful development and deployment of pervasive systems require the collaboration of software engineers and developers, hardware designers, wireless engineers, and human-computer interaction. Pervasive computing encompasses many areas of computer science and engineering, such as intelligent systems, agent technologies, mobile computing, wireless networks, distributed systems, middleware, wearable and context-aware computing, and device technology.

STRATEGIES FOR REFORMING SOFTWARE ENGINEERING EDUCATION

In order to address the sheer scale and complexity of the issues faced in pervasive computing, it is necessary that software engineers achieve new competences that are beyond those required in distributed and mobile computing. This calls for a reform of software engineering education, which is currently focused on distributed and mobile computing.

The software engineering education reform must address both technical and non-technical challenges. Table 1 presents a list of topics recommended for integration in the software engineering curriculum as a part of this education reform effort. Integrating research into education; and partnering with industry in research and education, are necessary elements of this reform effort. It is also critical to provide opportunities for software engineering students to achieve the competences required for adopting a multidisciplinary approach to real-world problem solving, and for engaging in life-long learning.

Table 1: List of new topics for integration into the software engineering curriculum.

Part I	Introduction to Pervasive Computing
Part II	Technical Challenges
Part III	Multidisciplinary Approach
Part IV	Integrating Research into Education
Part V	Case Studies
Part VI	Experimental Projects

Part I: Introduction to Pervasive Computing

The motivation and the historical perspective are described through Weiser's papers from the early 1990s and publications from other researchers that give a historical viewpoint about the evolution of pervasive computing research since Weiser's early work in the 1990s. An overview of the hardware, software and networking advances, which have brought Weiser's vision close to reality, is also covered. The most remarkable advances include the following:

- The creation of the World Wide Web;
- The advances of agent component technologies and agent-oriented software engineering [4];
- The widespread adoption of digital mobile computing.

The Web has provided access to information and services from any computer, including PDAs and low-power mobile devices. Using computers as portals to the Web, the view of the digital world, as offered by a particular system, becomes more important than the computers themselves. This has helped create a culture that is considerably more amenable to pervasive computing.

In addition, the recent advances in agent component technologies and agent-oriented software engineering have made it possible to develop enterprise systems that offer greater adaptability and flexibility than conventional component-based systems [4-8]. Agent components are used to support context-aware computing, both individual and teamwork, vigilantly monitoring of business processes and system status, and flexibly adapting system response and reconfiguration to circumstances. Agent-based systems offer salient strengths, such as the following:

- Highly customisable: having the ability to adapt to user's goals and tasks.
- Autonomy: having goals, proactively monitoring the relevant situations (or events) and trying to perform actions based on an assessment of the situation.
- Context-aware: detecting situations (or events) and adapting to behaviour.
- Collaborative: interacting with other agents to perform sub-goals or provide information [4][9].

Furthermore, the widespread adoption of viable commercial mobile products, like wireless LANs, PDAs, wearable computers and devices to sense and control appliances, affects the ability to develop and deploy pervasive systems. Other major technological advances that help in implementing the pervasive computing vision include processing capability, storage capacity and high-quality displays [10].

Part II: Technical Challenges

The key technical challenge is to develop and deploy proactive systems that are capable of monitoring, predicting and reacting to physical world conditions. A proactive system must closely and reliably integrate sensors and actuators with the physical world. This necessitates greater complexity in the system components to make the system process real-world data efficiently, robustly, accurately and securely; and affect the physical world as needed. In order to support highly dynamic and varied human activities in a pervasive computing environment, systems must be:

- Pervasive: being everywhere and reaching into the same information base by every portal;
- Embedded: sensing and affecting the physical world;
- Nomadic: allowing users and computations to move around freely according to their needs;
- Adaptable: providing flexibility and spontaneity in response to changes in user's requirements and operating conditions;
- Powerful yet efficient: freeing itself from constraints imposed by bounded hardware resources, addressing system constraints imposed by user demands and available power or communication bandwidth;
- Intentional: enabling people to name services and software objects by intent;
- Eternal: never requiring shut down or reboot while components are added or removed in response to demands, errors and upgrades [9].

Part III: Multidisciplinary Approach

Pervasive computing environments will face a proliferation of users, applications, a variety of devices and intelligent spaces that interact on a scale far beyond what is experienced today. Thus, many current technologies must be revamped and combined, and the existing backbone IT infrastructure has to be extended so as to meet the anticipated demand. Existing applications for global networks (eg the Internet) must be modified to completely integrate pervasive computing devices into the existing social systems. To implement the pervasive computing vision, technologies such as the following must be revamped and integrated:

- Device technologies;
- Network technologies;
- Software technologies.
- User technologies;
- Perceptual technologies [9].

In a pervasive computing environment, device and network technologies will create a seemingly uniform computing space in a heterogeneous and mobile computing environment; connecting dynamically changing configurations of self-identifying mobile and stationary devices to form collaborative areas. Pervasive computing should encompass every device that has built-in active and passive intelligence. Networks will need to configure and reconfigure themselves automatically, as nodes appear, migrate or disappear.

Software technologies, such as agent component technologies, will support adaptability of software systems to the needs and requirements of different users and environments, and also to the various changes and failures in the system with minimal user interventions and without the interruption of the services they provide. Agent components will work on behalf of users, devices and applications in order to effectively provide transparent interfaces between different entities in the environment, thus enhancing the invisibility envisioned in pervasive computing [4].

In a pervasive computing environment, user technologies will directly address human needs, and consist of the following:

- Collaboration technologies to enable the formation of spontaneous collaborative regions that accommodate the needs of mobile people and computations, and also for providing support for recording and archiving speech and video fragments from a variety of sources and/or events.
- Knowledge access technologies to offer vastly improved access to information, customised to the needs of users (ie people, applications and software systems).
- Automation technologies to offer natural, easy-to-use, customisable and adaptive mechanisms for automating and tuning repetitive information and control tasks [9].

Perceptual technologies (eg speech and vision technologies) will enable communication with device, network and software to extend the range of user technologies delivered to all places.

Part IV: Integrating Research into Education

In order to increase the effectiveness of students' learning experiences, and to stimulate innovative educational activities in the software engineering discipline, it is critical to integrate a large and growing body of research on pervasive computing research into software engineering courses and curricula. The research may be an ongoing or a completed research, and also from projects led by the professor or by others [11]. A list of research issues includes:

- Development of a robust agent-based system architecture that enables the software to adapt to changes in location and/or the needs of a user to respond both to component failures and newly available resources, and also maintains the continuity of service even when the set of available resources changes as a result of a resource failure or a resource addition.
- Development of a middleware that interfaces between the networking and the end-user applications running on pervasive devices, mediates interactions with the network on the user's behalf, and provides users the services and information offered by pervasive computing environments. The pervasive middleware will consist of firmware and software bundles that run in either client-server or peer-to-peer mode.
- Development of a middleware that provides transparent service to users by addressing the issue of heterogeneity and interoperability in pervasive computing environments. It is critical to find ways to mask the heterogeneity between intelligent spaces that offer different levels of infrastructural intelligence.
- Development of pervasive computing applications that run across all platforms. Today's applications are typically developed for specific device classes or system platforms; leading to separate versions of the same application for handhelds, desktops and cluster-based servers.
- Coordination and integration of pervasive computing components in a way that address the issues of reliability, quality of service, invisibility and security in pervasive networking.
- Explicit distribution and installation of applications for each class and family across a wide geographic area as the number of devices grows, and the distribution and installation of applications become unmanageable.
- Development of pervasive computing applications involving multiple components, which are required to manage their configuration changes, for example by developing and using reflective middleware platforms. Although reflection has been used to support adaptation in mobile computing environments, its application to pervasive computing systems requires more extensive research.
- Development of systems and devices that perceive context to provide service in a pervasive computing environment. This is more complex than it is in mobile computing. Mobile computing addresses location- and mobility-management issues in a reactive context mode (responding to discrete events). Many computing systems and devices today cannot sense their environments and, therefore, cannot make timely, context-sensitive decisions.
- Development of techniques and technologies to address the security and privacy issues in pervasive computing. Just as users must be confident of their computing environment's trustworthiness, the infrastructure must be confident of a user's identity and authorisation level before responding to the user's requests. It is difficult to establish this mutual trust in a manner that is minimally intrusive [12]. An inherent contradiction lies at the heart of the pervasive computing vision. On the one hand, a computing environment must be highly knowledgeable about a user to conform to his/her needs and desires without explicit interaction – almost reading the user's mind. On the other hand, a system that is truly ubiquitous will encompass numerous users, physical areas, and service providers. At such a large scale, perfect trust among all parties is an impractical ideal. Trust boundaries then represent seams of discontinuity in the fabric of pervasive computing [12]. The research issues include how to achieve a balance between a conflicting criteria, such as proactive actions and privacy; security and cost-efficiency; and security, privacy and quality. The

approaches based upon increasing awareness, maintaining an audit trail, and creating a *sixth sense* are promising [12].
- Development of new techniques for a user interface to remind users that their locations are being monitored and also alert them when the trustworthiness of the entity performing that monitoring changes. Thus, more research on protocols and mechanisms for authenticating and certifying an individual's location at any given time is required.

Part V: Case Studies

Discussing ongoing major projects, such as the following, can offer valuable information and insight:

- MIT's Oxygen Project [9];
- Hewlett-Packard Labs' Cooltown (www.cooltown.com);
- The University of California at Berkeley's Endeavour Project (endeavour.cs.berkeley.edu/);
- CMU's Aura Project (www-2.cs.cmu.edu/~aura/).

Part VI: Experimental Projects

Many experimental projects in this new field require the use of state-of-the-art software that is not yet well tested, supported or stable. Facing these practical issues in the course of projects provide a great educational opportunity for software engineering students to gain valuable experience that cannot be acquired by reading technical papers. Experimental projects help students learn to appreciate applied research and the work that is behind research papers that present the development and deployment of software prototypes. They also provide opportunities for innovative engineering solutions. Hence, experimental projects help students become prepared for their challenging software engineering career, particularly in the R&D sector.

Effective and Efficient Collaboration and Communication across Fields

Due to the multidisciplinary nature of pervasive computing, it is essential to pull diverse expertise and experiences together to develop practical solutions to real-world problems. This necessitates providing opportunities for software engineering students to acquire skills for effective and efficient collaboration and communication across fields and the profession. This requires multidisciplinary projects in academia.

Life-Long Learning

The rapidly evolving and highly diversified world of pervasive computing requires software engineers to be life-long learners. Hence, it is essential to provide opportunities for software engineering students to:

- Recognise the crucial need for life-long learning;
- Acquire skills to actively engage in life-long learning [13].

Partnership with Industry in Research and Education

A strong and sustained industry-academic partnership significantly helps provide students with the best of the two worlds. More specifically, it helps develop more effective software engineering education programmes and ensure that university research will have greater access to, and influence on, industrial-scale development. It is crucial to differentiate a partnership in education and research from a training exercise [14][15].

CONCLUDING REMARKS

Reform in software engineering education is inevitable in order to move pervasive computing from the laboratories into the real world. In order to facilitate the software engineering education reform efforts, this article has presented key strategies that are confirmed to be effective in a similar effort led by the author. These strategies include integrating research into education, building a solid foundation for both multidisciplinary work and life-long learning, and partnering with industry in research and education.

REFERENCES

1. Weiser, M., The computer for the twenty-first century. *Scientific American*, 265, **3**, 94-104 (1991).
2. Norman, D.A., *The Invisible Computer*. Cambridge: MIT Press (1998).
3. Dertouzos, M.L., The future of computing, *Scientific American*, 281, 2, 52-55 (1999).
4. Griss, M. and Pour, G., Accelerating development with agent components. *IEEE Computer*, 34, **5**, 37-43 (2001).
5. Pour, G., *Web-Based Multi-Agent Architecture for Software Development Formal Peer Inspection*. In: Mohammadian, M. (Ed.), Computational Intelligence for Modelling & Control Series (2003).
6. Pour, G. and Hong, A., *Internet-Based Multi-Agent Framework for Software Service Retrieval & Delivery*. In Mohammadian, M. (Ed.), Computational Intelligence for Modelling & Control Series (2003).
7. Griss, M., Letsinger, R., Cowan, D., Sayers, C., VanHilst, M., and Kessler, R., CoolAgent: Intelligent Digital Assistants for Mobile Professionals - Phase 1 Retrospective. Hewlett-Packard (HP) Laboratories report HPL-2002-55(R) (2002).
8. Jennings, N.R. and Wooldridge, M.J.., *Agent Technology*. Springer (1998).
9. MIT Project Oxygen, http://www.oxygen.lcs.mit.edu/
10. Want, R., Pering, T., Borriello, G., and Farkas, K., Disappearing hardware. *IEEE Pervasive Computing*, 1, **1**, 36-47 (2002).
11. Pour, G., An innovative approach to integrating research into education in Component-Based Software Engineering (CBSE). *Proc. 6th UICEE Annual Conf. on Engng. Educ.*, Cairns, Australia, 105-108 (2003).
12. Satyanarayanan, M., Privacy: the Achilles heel of pervasive computing? *IEEE Pervasive Computing*, 2, **1**, 2-3 (2003).
13. Pour, G., Agent-Oriented Software Engineering (AOSE): its emergence as a cornerstone of enterprise software engineering education. *World Trans. on Engng. and Technology Educ.*, 2, **2**, 225-228 (2003).
14. Pour, G., Component-based development refining the blueprint of software engineering education. *World Trans. on Engng. and Technology Educ.*, 2, **1**, 45-48 (2003).
15. Pour, G., Griss, M., and Lutz, M., The push to make software engineering respectable. *IEEE Computer*, 33, **5**, 35-43 (2000).