*IRACST – International Journal of Computer Networks and Wireless Communications (IJCNWC), ISSN: 2250-3501*
*Vol.12, No 1, Jan– Mar 2022*

# Changes and Future Directions in Software Engineering Education

Nancy R. Mead
*Software Engineering Institute*

***Abstract***
In this talk we'll trace the history of software engineering education and focus on some of the key players. Although the time has gone by quickly, it is more than 20 years since the first CSEE was held, and there have been many changes. We'll highlight the work that we have accomplished in the areas of degree programs and curricula, conferences and working groups, professionalism, certification, and industry-university collaboration.
We'll also look at the challenges that lie ahead—the global reach of education, new delivery mechanisms, new professional efforts, and the need for us to engage in leadership in software engineering education. What new approaches should we be considering? How can we maintain our vitality? How can we best nurture new educators and encourage others to join our profession?

## Introduction

This paper is reminiscent of *A Christmas Carol*. I feel like I am writing about the ghost of Christmas past, present, and future! In some regards it's hard to believe what has been accomplished in a relatively short time. On the other hand, it feels like we have barely scratched the surface in terms of what can be done to further software engineering education.
We'll start by taking a look at our past accomplishments. Then we'll take a look at the exciting developments taking place right now. Finally we'll get out the crystal ball and try to guess what lies in the future. As we are doing this, we'll remember the many contributions made by leaders in the past, take a look at what the current leadership is accomplishing, and guess at the makeup of our leaders in the future. Needless to say, so much has happened that this short paper is far from complete. I am just scratching the surface in this list of accomplishments and have mentioned only a few of the people who have been so influential in the field. I hope to fill in some of these gaps in my talk.

**The early days**

I was first exposed to software engineering education when I was a practitioner at IBM in the 1970s. (Of course at that time, the term software engineering was still not in common use.) Harlan Mills had convinced the then President of IBM Federal Systems Division, John Jackson, to offer a series of courses entitled Structured Programming Workshop, Structured Design Workshop, and eventually Advanced Design Workshop, fondly known as SPW, SDW, and ADW. Every programmer in the Division was to take SPW, lead programmers were to take SDW, and the elite would eventually take ADW. Eventually these course offerings were supplemented by university short courses. Although it doesn't seem like much now, taking programmers off the job for two to three weeks was a big commitment at the time. Workshop completions were tracked at the highest levels of management. The workshops resulted in the Linger-Mills-Witt book on structured programming [1], which is still in use. Many of the early pioneers from that era are still working, such as Rick Linger, but others such as Mills and Dijkstra have passed on, having revolutionized the field.

### 1.1. MSE programs, conferences, and workshops

In the late 1970s, Master's degree programs in software engineering started to emerge. The workshop series that was so successful in IBM's Federal Systems Division grew into a corporate-level commitment, and IBM's Software Engineering Institute was founded. The corporate model was slightly different, but many of the same courses were offered across the company, and at the corporate

level, in-house instructors were certified to teach them via a formal instructor training program.

During the early 80s, these educational activities continued, and a study was undertaken that resulted in the current Software Engineering Institute at Carnegie Mellon University. One of the key elements of the original SEI charter was to advance software engineering education. Norm Gibbs was hired as Director of Education. Under his leadership, the first Conference on Software Engineering Education (CSEE) took place. Many of our current leaders were involved in these early activities, both inside and outside the SEI. Long before I thought of leaving IBM and joining the SEI, I co-authored a paper on software engineering education at IBM and submitted it to one of these early conferences, where it was published as part of the *Issues in Software Engineering Education* [2]. Unfortunately, I did not have the opportunity to present the paper, as the conference was too close to my wedding date!

As a consequence of these early workshops, and work at the SEI, the initial MSE model curriculum was developed, and an MSE program was started at Carnegie Mellon University in 1990 that continues to this day. Needless to say, the program has undergone changes in content and faculty since those early offerings. The initial faculty included many educators who are still teaching, some who have retired, and sadly, some who have passed on. The latter group includes Norm Gibbs and Jim Tomayko.

The number of MSE programs grew rapidly, and during these years the SEI developed and distributed instructional materials in the form of videotaped courses, curriculum modules, and educational materials. Many universities used these early materials as a springboard, whereas others developed programs that were uniquely their own. Over time, the various programs came to reflect the specialties and emphases of their universities [4].

The number of conferences and conference tracks also grew. The ACM Special Interest Group on Computer Science Education has long sponsored a conference series (SIGCSE) that has contained software engineering content from time to time. The series of Conferences on the Teaching of Computing (CTC) in Ireland had a considerable amount of software engineering content. More recently, the ICSE conference has included a software engineering education track. The CSEE conference series also broadened its focus to include training and is now known as the Conference on Software Engineering Education and Training (CSEET).

In the early 90s, the CSEE became the catalyst for a newsletter, the Forum for the Advancement of Software Engineering Education (FASE). Over the years it provided informal information to software engineering educators on a periodic basis. It continues to provide announcements and articles to its audience, which is worldwide. The editors included Keith Pierce, Don Bagert, and the current editors, Barrie Thompson and Helen Edwards.

Industry/university collaboration and the WGSEET

One of the hallmarks of the MSE program is that it was aimed at software practitioners who were already working in the field. As a consequence, the SEI started a continuing education group to focus on the needs of practitioners. This group initially worked on developing a set of courses that were parallel to the MSE courses [6] but later branched out into courses that were unique to practitioners in the field but not necessarily part of a degree program. Some of these courses were also videotaped and distributed. Train-the-trainer courses were offered so that in- house trainers could deliver the courses to their practitioners.

Some years later, a working group for software engineering education and training (WGSEET) was formed under my leadership. This was an ad hoc group whose goal was to advance software engineering education. The WGSEET provided a forum for advancing the field and for tackling controversial issues in a friendly environment. It continued for a number of years as a standalone group, publishing many useful papers and reports. Eventually it was disbanded in favor of work under the umbrella of IEEE CS.

It became clear that part of the agenda was to collaborate with industry as well as influence it.

The early MSE programs were among the roots of work in industry/university collaboration. Many universities established industry advisory boards so that their degree programs could remain relevant to their

industry partners. These collaborations took many forms—joint research, regular work sessions or meetings, and the like. This provided an opportunity to study these collaborations and try to figure out what made them tick [3], [5], [7], [10], [14].

However, industry/university collaboration was never a controversial issue. It took the discussion of undergraduate degree programs in software engineering and the idea of licensing software professionals to generate true controversy! A joint committee was formed by ACM and IEEE Computer Society in 1993, to promote software engineering as a profession. This group was known as the Software Engineering Coordinating Committee [8], which sponsored development of the SWEBOK [9]. Once the SWEBOK was completed, some professionals saw it as the basis for certification programs, but others feared that it would become the basis for licensing.

Licensing generated serious controversy among thoughtful people that continues to this day.

Papers about licensing generated a level of rhetoric that had not been seen before [11], [12]. More scholarly articles received fewer slings and arrows, but were nevertheless controversial [13]. Don Bagert was the subject of much discussion when he became the first licensed software engineer in the state of Texas. The discussion continues, and the future for licensing is unclear [20].

About the same time, undergraduate degree programs in software engineering education started to emerge, and a merger of the accrediting bodies, CSAB and ABET, made it possible for them to be accredited. New curricula were developed to support a variety of computing degree programs [18]. Universities such as RIT, under the leadership of Mike Lutz, were pioneers in offering the first undergraduate degrees in software engineering [21].

IEEE CS started to offer a certification program that was initially based on the SWEBOK and has evolved to include several levels of certification [19].

## Delivery mechanisms

Distance learning became a popular concept [15], as well as the establishment of branch locations among universities. Globalization became a buzzword as universities tried to outdo each other in the number of farflung locations that they supported.

## The present

The CSEET has evolved to include the ASEET, the Academy for Software Engineering Educators & Trainers. This is a wonderful addition to the conference, and supports the idea of mentoring as a way of growing the skills that are needed for professional software engineering education. It is truly an exciting development.

The conference itself has evolved from an SEI offering to an IEEE-sponsored conference. One bold move in the early days was to move the conference out of Pittsburgh. After many years in Pittsburgh, our first venture to another location was the conference in San Diego.

Another bold move was to start to hold the conference in international locations. Many of the early pioneers on the CSEET Steering Committee have been succeeded by dynamic young educators. This is all to the good.

New degree programs are springing up all the time. For example, at Carnegie Mellon, there are now a number of international locations that are offering unique degree programs, including Australia, Greece, Japan, Portugal, and Qatar [17]. We continue to define new degree programs, such as the new Master's degree program in Software Engineering Management, the MSIT- SEM.

Distance delivery via a variety of mechanisms is routine. Creative programs allow working students almost anywhere to complete many, and in some cases all, of their degree requirements at a distance.

## The future

Although it's interesting to reflect on the past, we don't want to get stuck there. Nor do we want to become complacent in the present. Change is happening all around us and we need to adjust to and motivate change. Here are some possible indicators of future trends:

At our last conference, Dave Parnas [16] said that he has come to think that maybe
there are three types of degree programs: computer science for researchers, software engineering for engineers developing mission-critical systems, and software professional degrees for practitioners developing non-critical systems.

The trend toward globalization is here to stay. We will continue to see creative
evolution of delivery mechanisms. We can also expect to see multicultural teams and offshoring. Of course the meaning of the term offshoring depends on where you are located.

Offerings such as the ASEET will help to develop a new generation of software engineering educators. Conferences, working groups, and committees will evolve. Electronic publishing is becoming much more common, as are video conferences and webcasts.

Professionalism continues to be a hot topic, and it's still unpredictable whether
licensing or certification will become a major trend in the future.

New specializations will emerge. For example, software security and software assurance have become big concerns for many of us, and these as well as other new directions will result in change.

Our challenge is to be change agents and to motivate the next generation of software engineers *and* software engineering educators. It won't be easy, but it will be fun!

## References

R. C. Linger, H. D. Mills, and B. I. Witt, *Structured Programming: Theory and Practice*. Reading, MA: Addison-Wesley, 1979.

N. Hall and J. Miklos, "Formal education within the software life cycle," in *Issues in Software Eng. Ed.* (Proc. 1987 SEI Conf. on Software Eng. Ed.), Monroeville, PA, 1987. New York: Springer-Verlag, 1989.

N. R. Mead, "Industry-university partnerships: The wave of the future?" in *Proc. 4th Annu. Conf. Teaching of Computing*, Dublin, 1996.

http://ieeexplore.ieee.org/iel5/10731/33854/01612026.pdf

N. R. Mead, K. Beckman, N. Coulter, and S. Khajenoori, "Industry/university collaboration: Closing the gap between industry and academia," *IEEE Software*, vol. 14, no. 6, pp. 49–57, Nov.-Dec. 1997.

N. R. Mead and P. Lawlis, "Software engineering: Graduate-level courses for AFIT professional continuing education," in *5th SEI Conf. Software Eng. Ed.* New York: Springer-Verlag, 1991.

K. Beckman, J. Lawrence, G. O'Mary, C. Parish, P. Unpingco, and H. Walker, "Industry/university collaborations: different perspectives heighten mutual opportunities, " *CrossTalk*, vol. 13, no. 3, pp. 10–15, Mar. 2000.

http://computer.org/tab/swecc.htm

http://www.swebok.org/overview/

N.R. Mead, H. J. C. Ellis, A. Moreno, and P. MacNeil, "Can industry and academia collaborate to meet the need for software engineers?" *Cutter IT Journal*, vol. 14, no. 6, pp. 32–39, June 2001.

N. R. Mead, "Issues in licensing and certification of software engineers," in *Proc. 10th Conf. Software Eng. Ed. and Training*, Virginia Beach, VA, 1997, pp. 150–160.

N. R. Mead, "Are we going to fish or cut bait? Licensing and certification of software professionals," *Cutter IT J.*, vol. 11, no. 5, pp. 4-8, May 1998.

N. R. Mead and A. J. Turner, "Current accreditation, certification, and licensure activities related to software engineering," *Annals of Software Eng.*, vol. 6, pp. 167–180, 1998.

"Industrial Input to the Computing Curriculum," in *Effective Learning and Teaching in Computing*, A. Irons and S. Alexander, Eds. New York, NY: Routledge Falmer, 2004, pp. 123–135.

N. R. Mead, "Distance learning: Lessons learned," in *5th Annu. Conf. Teaching of Computing*, Dublin, 1997.

http://csdl2.computer.org/persagen/DLAbsToc.jsp?resourcePath=/dl/proceedings/&toc=comp/proceedings
/cseet/2007/2893/00/2893toc.xml&DOI=10.1109/CSEET.2007.43

http://www.cmu.edu/global/flash/index.html

Joint Taskforce for Computing Curricula (2004), *Software engineering 2004, curricular guidelines for undergraduate programs in software engineering*, New York, NY: ACM/IEEE, 2004.

http://www2.computer.org/portal/web/certification/about

http://www.nspe.org/PEmagazine/pe_1207_Software_License.html

http://tab.computer.org/fase/fase-archive/v6/v6n14.txt