Another email from SEN editor Will Tracz, my third SEEed column is due tomorrow (joking, really in two weeks). What a slave driver, and now he tells me that he wishes it was longer. I wish I could write about 285 words to get my key points across. Why 285 words? The number of words in Abraham Lincoln’s Gettysburg Address (19 Nov 1863)¹.

In the past two columns I have devoted more than 285 words to the ongoing CCSE (Computing Curricula for Software Engineering) efforts [4]. Time to move on in this column.

**SEEed Home Page:** Current and past SEEed columns can be found at the following URL http://blue.butler.edu/~phenders/SEEed

**NSEFS 03 – The New Software Engineering Faculty Symposium** [3], organized by Leon Osterweil had about 25 new SE faculty attend the one day symposium that addressed both SE research and education issues. One participant I talked with felt this workshop was very beneficial. To help new software engineering faculty the organizers will be developing a web site, which should be up by the time you read this column. Please check it out, especially your new SE faculty. It will be accessible from the NSEFS 03 link [3].

**Quote of the Month?** “I wish I could find a good quote for this column!” by Peter Henderson who is looking for that perfect quote. If you have found one published somewhere or heard a quote you like, please send it to me.

**ICSE 2003 : Panel Undergraduate SE Degree: Pros and Cons.** Prior to this panel there was a lively email exchange amongst software engineering educators worldwide initiated by Don Bagert from questions posed by Dewayne Perry, the panel moderator, for the panelist. This exchange was very useful, so I would like to summarize key points here. This will be followed by a capsule overview of the actual panel session by Don.

What are[should] the differences between computer science, software engineering and computer engineering [be]? Mark Ardis response says it well. “Software engineering should be to computer science as electrical engineering is to physics. That is, software engineering is about the application of knowledge and techniques for developing software that are studied and developed in computer science. Computer engineering is about hardware, software engineering is about software.” This is probably too simplistic because of all the potential interrelationships between the three disciplines, but does provide a concise distinction.

What are the primary impediments to an undergraduate SE major? There seemed to be general agreement that two of the major impediments are the lack of qualified software engineering faculty and lack of maturity of the discipline (some felt there was sufficient knowledge whereas others argued about how well this knowledge was organized). Tim Lethbridge comments further: “First and foremost, lack of understanding of SE from both CS and EE/CE {traditional engineering} people.” Of course, there are always the political and resource issues pitting faculty in the various disciplines against each other. New SE textbooks, courses and course materials will come, but are sufficient to meet current needs. Another point made by Tom Hilburn relates to “marketing” software engineering programs to potential students and industry. With the proliferation of so many different computer based undergraduate programs of study, it is hard for students to select one that matches their interests and needs. And what about the corporate world trying to make sense of so many varied programs – varied with respect to emphasis and quality – to meet their needs?

**What are the arguments in favor of an undergraduate SE major?** Mark Sebern responds “SE practitioners (who can apply engineering processes and methods to software development) are sorely needed by industry and society.” Mark continues: “… current MSSE programs are really “entry level” (due to the lack of BSSE graduates) and often cover a subset of the material now being incorporated in BSSE curricula. Thus one additional reason for having a BSSE is so that we can have more advanced MSSE programs …” Tim Lethbridge notes that: “Having undergraduate {SE} programs will help accelerate maturing the discipline.”

There was general agreement that one primary reason is the requirement for better approaches to developing and maintaining software systems.

**Where should SE be housed?** David Parnas argued that SE should be housed with engineering [2] primarily to reinforce and maintain the engineering mindset. However, there was general agreement in the discussion that the two viable alternatives are: a combined CS/SE department or a separate SE department – these may be administered by a college of computing. The best fit should be a local decision, highly dependent upon local politics, administration, resource allocation, etc.

**Can the current apathy/antagonism towards SE in CS departments be overcome?** This derives from understanding, lack of maturity of SE, and contention for resources. What is SE? How is it different from CS? Why change to SE since CS has “sufficiently” provided software practitioners to date? Fear of change! Recall my prediction in the first SEEed column “Within 50 years, software engineering will supplant computer science as the educational discipline for professional software developers.” Some in the discussion felt that CCSE, the software engineering curriculum effort would lend credence to SE as a recognized and viable discipline.

---

¹ Check out Lincoln’s powerpoint version for a good laugh http://www.norvig.com/Gettysburg/index.htm
Time will tell. The capsule overview of the actual panel session by Don Don Bagert follows:

Panel Chair: Dwayne E. Perry (University of Texas at Austin)
Panelists: Don Bagert (Rose-Hulman Institute of Technology), Rich LeBlanc (Georgia Institute of Technology), Hausi Muller (University of Victoria), Wilhelm Schaefer (University of Paderborn), Michael Young (University of Oregon)

Dewayne was impartial but mentioned that he is trying to start a BSSE at UT-Austin. Rich, Hausi and I were pro-undergraduate SE, while Wilhelm and Michal were against it. The two that were the most forceful in their views were Hausi and Michal. Hausi felt that a SE degree should be developed, implemented and housed as independently from CS as possible. Michal felt that there was no way to have the flexibility to do what is necessary to achieve the proper background for an SE degree without a four-year CS degree with various options which would allow for a variety of backgrounds for software developers.

Wilhelm maintained that it would be difficult to achieve a baccalaureate software engineering degree in many European countries due to their educational systems. (From the audience, Barrie Thompson noted that such degrees have existed in the UK for about 15 years, and mentioned the Bologna Declaration on higher education in the European Union, which hopes to bring the systems in the 59 countries that signed the declaration closer to together in future evolution [5]).

Rich discussed the CCSE activities [4], and pointed out that accreditation of undergraduate SE programs has gone on for many years in the UK, Australia, Canada and (starting this year) in the U.S. I tried to provide a proof-of-concept (using the proposed Rose-Hulman degree) that it was possible to have a BSSE which included the standard CS background and yet had sufficient SE topics plus enough flexibility to do (some) different things.

There were only about 40 people at the panel, since there was a mini-tutorial by Mary Shaw on how to write a research paper opposite it, taking away many of the young researchers. David Notkin (another opponent of undergraduate programs) did not attend, as well [editors note: David will be expressing his views in a future SEEd column].

Overall, I think few (if any) minds were changed, and to me that’s a plus for the baccalaureate programs, which will continue to grow and thrive absent the presence of some organized, well-placed opposition. [End of Don’s report]

Definition of Engineering

"Engineering is the profession in which knowledge of the mathematical and natural sciences, gained by study, experience, and practice, is applied with judgment to develop ways to utilize, economically, the materials and forces of nature for the benefit of mankind."

Does this definition of engineering match your views of software engineering? In what ways is it consistent? Inconsistent?

Here are some of my views. What are the “natural sciences” of computing? Perhaps “natural laws” or “natural theories” might be more appropriate for SE. Indeed, I think of computer science as the foundational science for SE – similar to the way chemistry is a foundational science for chemical engineering. OK, what are the “materials and forces of nature” impacting computing? Most of the “material” is conceptual rather than physical, and the “forces” are really “laws.”

So, it appears that the ABET definition may be too square for the round hole called software engineering (i.e., does not fit). Then, let’s refine it! How about:

"Engineering is the profession in which knowledge of the foundational mathematics and sciences, gained by study, experience, and practice, are applied with judgment to develop ways to utilize, economically, the materials, concepts and laws of nature for the benefit of mankind."

Not perfect, but perhaps a better fit. We could go on and on refining, but …

But, there is still one important point I would like to make. Notice my placement of the word “foundational” and how it relates to the ‘M’ word in this definition (not “materials” or “mankind”). Look at your favorite definition of software engineering to see if the ‘M’ word appears.

Mathematics, maths, math, etc. is a required foundation for any “true” engineering discipline – in my opinion. Otherwise, I consider them to be “pseudo” engineering disciplines. Using my viewpoint is software engineering a “true” or “pseudo” engineering discipline? If you said “true,” then what is the foundational mathematics? How about your point of view, “true” or “pseudo”? Send me email with three lines, one answer for each of these questions. I will publish the results in the next column.

For those of you interested in mathematics in computing education there will be a special issue of the Communications of the ACM devoted to this topic published later this year. This issue is edited by Keith Devlin, author of numerous mathematics for the general public and the 2001 CACM viewpoint article “The Real Reason Why Software Engineers Need Math” [1].

Growth of BSSE programs worldwide: The following chart illustrates the growth of undergraduate software engineering programs (BSSE) worldwide that have been tracked since 1990. The shorter bars represent new starts that year. There are 15 existing programs whose starting date is unknown. These are not included in this chart. So there are currently about 65 known BSSE programs worldwide. However, there are numerous such programs

---

3 Skip to the next section if you don’t like reading views or opinions, especially mine!!
4 Try “google” if you don’t have a definition in mind.
5 Let’s watch the peanut butter hit the fan blades on this one.
popping up around the world, and primarily in India, China, and throughout south east Asia that are not shown. At last count there were about 22 BSSE programs in the US, but that number is growing. In the next column I will present a chart illustrating the distribution of BSSE programs worldwide.

![BSSE Programs Tracked](chart.png)

**Acknowledgements**

I would like to thank Steven Frezza and Joe Clifton who are tracking BSSE programs and provided the information for the Growth of BSSE programs worldwide section. They plan to provide a web site making this information available sometime soon. Also, I would like to thank Don Bagert for providing his first hand report on the ICSE panel Undergraduate SE Degree: Pros and Cons.

**References**

[4] [http://sites.computer.org/ccse/](http://sites.computer.org/ccse/)
[5] [europa.eu.int/comm/education/socrates/erasmus/bologna.pdf](europa.eu.int/comm/education/socrates/erasmus/bologna.pdf)