

Engineering, Software Engineering, Computer Engineering, Computer Science: What are they?

1 Seminar objective

To gain a better understanding of what *Engineering* is and what *Software Engineering*, *Computer Engineering* are, might be, or maybe aren't. Also, to get a handle on *Computer Science*. What is it? In particular, is it *Science*?

1.1 Strategy

- To discuss a number of issues relating to Engineering/Software Engineering/Computer Engineering/Computer Science.
- To base the discussion on both personal perceptions and documented statements by others.
- To recognise, but not necessarily to resolve conflicting statements.
- To document the outcomes of the discussion.

Ad hoc subgroups will be formed and discussion will alternate between small groups discussing and then the whole class discussing using Edward de Bono's *Six Thinking Hats* methodology:

White Hat:

what are the facts?

Red Hat:

what are the feelings?

Black Hat:

caution: what are the negative aspects?

Yellow Hat:

positive: what are the good aspects?

Green Hat:

creative: what new ideas can we think of?

Blue Hat:

control: what do we need to think of? what have we discovered?

Please see the summary of the [six thinking hats](#).

1.2 Issues to be discussed

Engineering: What is it?

How is *Engineering* described? What criteria are used? Anything about what is *not* Engineering?

What other professions are there? What's is it that makes a profession "engineering".

Software Engineering?

How is *Software Engineering* described? Why is it Engineering? Why is it not Engineering? How could it be Engineering?

Computer Engineering?

What is it? Does it differ from *Software Engineering*? How?

Computer Engineering?

Computer Engineering is just Electrical Engineering with an IT label?

Computer Science?

What is it? How does it relate to Software/Computer engineering? Can the differences be resolved?

What's licencing all about?

Why might licencing be required? What are the advantages? What are the disadvantages?

Why would you want SE/CE to be E?

Why would you *not* want SE/CE to be E?

Who cares?

Does it matter if SE/CE is not E?

What is SE/CE if it's not E?

Is CS Science?

Why would you want CS to be Science?

Does it matter?

What do you think you are?

What would you like to be?

Where to from here?

2 Sources for discussion

There are a number of sources that you are encouraged to consult. Perhaps the best, because it discusses the prime question for us, is the chapter, [Are "Software Engineers" Engineers](#) in the book, *Engineering*, by Davis [1]

Please read that carefully. It contains many interesting observations, for example:

Let me answer the last question first: Defining a field is more than semantics. How we define a field can affect how it develops. Software engineering may be a field whose progress is threatened by the analogy with engineering, a field pushed toward an unnecessarily rigid curriculum. That is the first reason our questions about software engineering are worth answering. Second is that trying to answer them will help us understand engineering. What are its boundaries? What is at stake when we draw such boundaries? A third is that trying to answer such questions tests the utility of our history of engineering. What insight can this history give us?

Nothing said here is meant to raise questions about the status of software engineering as a discipline, an occupation, or even a profession. My concern is how to conceptualize this new but already respectable occupation. Perhaps we would understand it better if we stopped trying to borrow concepts from engineering and instead borrowed them from architecture or industrial design, areas in which chemistry, physics, and mathematics are less important, pure invention more so, and codes of ethics less detailed. Or, perhaps we should borrow concepts from construction management. Software engineering may be more like overseeing the building of a great public work (a bridge, skyscraper, or power plant) than like doing the engineering for it. Construction managers are at least as good as engineers at delivering on time, within budget, and to the customer's satisfaction. Or, perhaps software engineering is more like what lawyers do when they create new negotiable instruments or complex land-use agreements.

The question to be asked, then, is not whether software engineers are engineers. Clearly, while some are, most are not. The question is, rather, whether (or when) they should be.

My conclusion is that there is no fact of the matter here, only a complex of social decisions about standards of training and conduct in need of attention. Like engineering, software engineering is a social project, not a natural species.

2.1 Etymology

engine any mechanism or machine designed to convert energy into mechanical work. From Latin *ingenium* nature, invention

engineering the art or science of making practical application of the knowledge of pure science, such as physics, chemistry, biology, etc.

Macquarie dictionary

Of course the English language is a living thing, but you will probably spot some problems in applying the above to software engineering?

But ...see the OED entry for *engineering* below.

engine

History

The word *engine* entered Middle English in the sense "ingenuity, cunning" coming via Old French *engin* from Latin *ingenium* "talent, device" (the root also of *ingenious*) ... from which in the mid 17th century arose the idea of a machine.

engineering · *n* **1** the branch of science and technology concerned with the design, building, and use of engines, machines, and structures; the practical application of scientific ideas and principles. **2** a field of study or activity concerned with modification or development in a particular area: *software engineering*.

Concise Oxford English Dictionary

2.2 Perhaps Engineering is not Based on Science?

Ferguson in *Engineering in the Mind's Eye* [2] gives a quote of Richard Feynman (a famous physicist) that sums up what his book is about:

One time, we were discussing something —we must have been eleven or twelve at the time— and I said, "But thinking is nothing but talking to yourself."

"Oh yeah?" Bennie said. "Do you know the crazy shape of the crankshaft in a car?"

"Yeah, what of it?"

"Good. Now tell me: how did you describe it when you were talking to yourself?"

So I learned from Bennie that thoughts can be visual as well as verbal.

2.3 Wikipedia

Consult the [Wikipedia](#) . in particular,

- [Engineering:](#)

Engineering is the application of *scientific* and *technical knowledge* to solve human problems. Engineers use imagination, judgment, reasoning and experience to apply *science*, *technology*, *mathematics*, and practical *experience*. The result is the *design*, *production*, and operation of useful *objects* or *processes*.

- [Software Engineering:](#)

Software engineering (SE) is a *profession* whose members create and maintain *software applications* by applying *technologies* and *practices* from *computer science*, *project management*, *engineering*, *application domains* and other fields.

Software engineering, like traditional engineering disciplines, deals with issues of cost and reliability. Some software applications contain millions of lines of code that are expected to perform properly in the face of changing conditions, making them comparable in complexity to the most complex modern machines. (The Boeing 777-200 has about 132,500 engineered and unique parts. Including rivets, bolts and other fasteners, the airplane has more than 3 million parts. Included is approximately 1400 data processing units and 5 million lines of code.)

- [Computer Engineering](#)

Computer Engineering (also called Electronic and Computer Engineering or Computer Systems Engineering) is a discipline that combines elements of both Electrical Engineering and Computer Science.[1] Computer engineers are electrical engineers that have additional training in the areas of software

design and hardware-software integration.[citation needed] In turn, they focus less on power electronics and physics. Computer engineers are involved in many aspects of computing, from the design of individual microprocessors, personal computers, and supercomputers, to circuit design. This engineering monitors the many subsystems in motor vehicles.

- [Computer Science](#)

Computer science (or computing science) is the study of the theoretical foundations of information and computation, and of practical techniques for their implementation and application in computer systems.[1][2][3] It is frequently described as the systematic study of algorithmic processes that describe and transform information; the fundamental question underlying computer science is, "What can be (efficiently) automated?"[4] Computer science has many sub-fields; some, such as computer graphics, emphasize the computation of specific results, while others, such as computational complexity theory, study the properties of computational problems. Still others focus on the challenges in implementing computations. For example, programming language theory studies approaches to describing computations, while computer programming applies specific programming languages to solve specific computational problems, and human-computer interaction focuses on the challenges in making computers and computations useful, usable, and universally accessible to people.

The general public sometimes confuses computer science with vocational areas that deal with computers (such as information technology), or think that it relates to their own experience of computers, which typically involves activities such as gaming, web-browsing, and word-processing. However, the focus of computer science is more on understanding the properties of the programs used to implement software such as games and web-browsers, and using that understanding to create new programs or improve existing ones.

2.4 CACM papers

There are a number of papers from the May and November 2002 issues of the Communications of the ACM. Here is a list:

[Searching for the Holy Grail of Software Engineering](#)
[Beware the Engineering Metaphor](#)
[The Missing Customer](#)
[Sorting Out Software Complexity](#)
[A Rice University Perspective on Software Engineering Licensing](#)
[Texas Licensing of Software Engineers](#)
[Licensing Software Engineers in Canada](#)
[Software Engineering Requires Individual Professionalism](#)
[Software Engineering Considered Harmful](#)
[ACM's Position on the Licensing of Software Engineers](#)
[Should Software Engineers be Licensed?](#)
[What UML Should Be](#)

2.5 Proceedings of NATO Science conferences

There is also the proceedings from the conference that started it all:

[SOFTWARE ENGINEERING, Report on a conference sponsored by the NATO SCIENCE COMMITTEE Garmisch, Germany, 7th to 11th October 1968](#)
[SOFTWARE ENGINEERING TECHNIQUES](#)

2.6 ABET: Accreditation Board for Engineering and Technology

ABET, the body responsible for engineering accreditation in the USA, gives the following definition of engineering:

Engineering is the profession in which a 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.

2.7 Quality Assurance Agency for Higher Education

The Quality Assurance Agency for Higher Education is a UK education accreditation organisation. The following is extracted from a document of theirs on Engineering.

What is Engineering? The Engineering Profession is a diverse discipline which has made a major positive impact on society, yet it is seldom defined in a manner which covers all contributing activities. Alongside the founding disciplines of Civil, Mechanical, Electrical and Chemical Engineering we have many other engineering disciplines that rightly claim a seat at the engineering table. What then is at the heart of all their activities, which makes the practitioners Engineers and their activities Engineering? Moreover what are the standards associated with those essential characteristics which may enable one to benchmark the knowledge, understanding, skills and know-how which should be attained by graduates in an honours degree programme for Professional Engineers?

The Engineering Council, in Part 2 of SARTOR 3rd Edition (Ref.2), sets out its definition of Engineering:

“Engineering is a profession directed towards the skilled application of a distinctive body of knowledge based on mathematics, science and technology, integrated with business and management, which is acquired through education and professional formation in a particular engineering discipline. Engineering is directed to developing, providing and maintaining infrastructure, goods and services for industry and the community.”

The outcome of Engineering is a product, or perhaps a process or service; it is this that distinguishes it from Science (which is about observing and explaining natural phenomena), and Mathematics (which is about describing and exploring relationships). Moreover, Engineering is not just Applied Science and Mathematics. Nor is it merely Applied Technology. Rather, Engineering is about the application of the understanding, knowledge, skills and know-how (as appropriate) of scientific, mathematical and technological principles in a business context to achieve an economic solution. This context is one of constraints and disciplines including those imposed by finance, legislation, ethics, and people. At a professional level, furthermore, Engineering requires innovation, creativity and flair focused in a design process.

See appendix A for more detailed description of generic activity of Professional Engineering.

2.8 Software Engineering program proposal

The need for the course

The software industry is one of the fastest growing industries in the world. Even companies that have been associated largely with hardware in the past are estimating that 80–90% of their engineers are involved, or will be involved by the year 2000, in software development. Accompanying this expansion there is a serious problem finding staff who are able to deal with the complexity of developing large software systems.

Paralleling the growth in the software industry is the increasing dependence of society on software. In many cases that dependence is critical, and failure of the software threatens the fabric of our society, and even the lives of citizens. Currently, very little software comes with a guarantee of any sort of fitness for its application, and software does not have a particularly good name for reliability. Society needs software to be produced to the same high standards required of any other engineering profession. Software engineers need to be able to produce verifiable, reliable software. In general, graduates in Computer Engineering, Computer Science, Information Systems and Electrical Engineering are employed to undertake software engineering work. Specifically from this University, the undergraduate courses in Computer Engineering, Computer Science, Business Information Technology, and Information Systems are the courses that are most relevant to Software Engineering. However, none of those courses are designed to give intensive treatment of Software Engineering.

Appendix A

What is Engineering? The generic activity of Professional Engineering may be represented by the diagram shown in the accompanying figure (Figure One). This shows all the activities (in the shaded portion) needed to satisfy a customer's requirements, but within the constraints of the business environment. Note that the customer may be an individual, a company or organisation, or a perceived market. Whoever the customer may be, the process proceeds as follows.

Customer requirements are those captured when the requirements of the customer are explored, developed and specified in relation to technical needs, legislative and statutory constraints (for example those relating to health and safety, equal opportunities, data protection, quality standards, and so on), and also to competitive and environmental factors imposed by political, economical, sociological, ethical, and similar considerations. The availability of Suppliers for components, materials and services is an important enabling requirement at this stage.

Technical specification is developed from the Requirements Capture, and represents a translation of those requirements into the language of the engineering disciplines involved in order to specify the performance of the product, process or service which is to be designed.

Functional design The functional design represents one stage of the design process whereby the product design is created to satisfy the strictly functional aspects required. The non-availability of practical design solutions at this stage will require iteration of the Technical Specification stage, and indeed Requirements Capture in order to establish a satisfactory functional design.

Design for test, maintenance, production, etc., is a further step in the design process, which reflects that functional design is but one aspect of the design process. Products, processes and services must be capable of being tested and maintained; they must be compatible with their environment (both technical and aesthetic), and above all, it must be possible to produce or manufacture them economically and with due concern for the environment.

Analysis of design is the means by which the design may be qualified or validated. The means by which this is done could simply be hand calculation, or computer simulation, or construction and evaluation of a model or prototype. This stage complements the synthesis of the design process.

Product approval is the stage at which the engineer draws conclusions regarding the design process by checking the performance achieved through design against the technical specification, the customer requirements, and hence the Customer. Approval from the latter can then signal the start of the Production or Manufacture process using the Suppliers' components, materials and services, and delivering to the Customer.

Operation, maintenance and disposal are the final elements for design; here the arrangements for operation of the product, process or service must be considered, together with those for maintenance, and finally disposal as the product life-end is reached. This completes the engineering cycle, which can start again for different, improved or updated products for the same or other customers. The stages in the design cycle define certain abilities and attributes on the part of the engineer working in a particular engineering discipline, many of which are clearly technical. These abilities and attributes in turn define the standards required in the education and continuing professional development of those engineers. Alongside the technical standards, however, there must be standards for those other abilities and attributes essential in engaging in the engineering cycle. These are shown on the diagram.

There is a requirement for Key Skills (often referred to as generic or general transferable skills)—skills in communication, using information technology, team working, time management, and appropriate understanding of the legislative and environmental framework in which the engineering process is executed.

Business Management ability includes the use and management of resources, particularly people;

Accounting and **Financial** ability represents another important aspect, whilst successful **Project Management** requires the disciplined organisation of activities and resources. All of the activities in the engineering cycle must be performed with due regard to **Quality**—ranging from the Quality Management

System in place, and the arrangements for Quality Assurance and Control, to the Quality Standards used in commerce and industry.

Finally, **Marketing**—Marketing is about finding out what the customer wants, and then providing it. It is thus inextricably linked to Engineering.

References

- [1] Michael Davis. *Thinking like an engineer: studies in the ethics of a profession*. Oxford University Press, 1998.
- [2] Eugene S Ferguson. *Engineering and the Mind's Eye*. MIT, 1992.