Software Engineering: An Unconsummated Marriage
(Extended Abstract)

David Lorge Parnas, P.Eng

Software Engineering Research Group
Communications Research Laboratory
Department of Electrical and Computer Engineering
McMaster University, Hamilton, Ontario, Canada L8S 4K1

When the first conference on "Software Engineering" was held, under NATO sponsorship three decades ago in Munich, the vast majority of Engineers ignored it. Electrical Engineers were obviously interested in building computers, but they regarded programming those computers as something to be done by others, often scientists who wanted the numerical results, or mathematicians who were interested in numerical methods. Programming was not viewed as engineering, but as a trivial task, akin to using a calculator. An engineer might have to perform such a task in order to get numerical results needed for some other task, but their real job was the other task.

The organisers of the first Software Engineering conferences saw things differently. Knowing that the engineering profession has always been very protective of its legal right to control the use of the word "engineer", they chose the title of the conference to provoke the interest of engineers. Those who organised and attended the conference had recognised four important facts:

1. The mathematicians recognised that programming wasn't really mathematics. They were not adding to our understanding of the properties of mathematical structures; they were building useful products, many of which would be used by others.

2. The scientists recognised that programming itself wasn't really science. They were not adding to mankind's knowledge of the world. Although the programs that they wrote might be used for scientific purposes, writing the programs was building a product, often one that would be used by others.

3. Building products to be used by others was engineering. The profession of engineering was invented, and given legal standing as a self-regulating profession, so that "customers" could know who was qualified to build technical products.

4. The software being built was not very good; it was becoming a major source of problems for those who owned and used it. This was the beginning of the "software crisis", a silly phrase that we still hear today. The problems were exactly those that you would expect if you allow products to be built by people who think that building products is not their "real job" and who were not prepared for this job by a professional education.

The organisers did not succeed in provoking the interest of engineers. Communication between those who study software and those who work as engineers has not been effective. Today, the majority of Engineers understand very little about the science of programming or the mathematics that one needs to analyse a program.

On the other hand, the scientists who study programming understand very little about what it means to be an engineer, why we have such a profession, how the profession is organized, or the things engineers learn during their education. In spite of this mutual ignorance, many of today's engineers spend much of their time writing and using software, and an increasing number of people trained in computer science or mathematics pontificate about "what engineers do". The purpose of this talk is to discuss both fields and attempt explain each to the other.

1 Why is engineering an organised profession?

Bridges can collapse and engines can explode. In the past, many people presented themselves as qualified to design, and direct the construction of, those products, but *did not have the requisite knowledge and mathematical ability.* The public, governments, and other potential customers wanted to be able to assess the qualifications of those offering their services. Potential customers did not have the knowledge necessary to make the judgement. The solution was to establish an association of Engineers with the power to license their colleagues. Each newly recognised Engineer could become a member of the association and participate in the evaluate of future candidates.

This solution has been adopted in many jurisdictions. Generally, legislation states that nobody may practice engineering or claim to be an engineer unless they have been recognised as qualified by becoming a member of the association of "Professional Engineers". The associations are obligated, by the same legislation, to make sure that all of its members are qualified to practice, that they are aware of their professional responsibilities, and that others who are qualified to practice will be able to enter the association. This is difficult task has been taken seriously by the associations with which I am familiar.

These associations have set up:

- a registration system to verify the qualifications of individuals,
- an accreditation system for institutions to certify appropriate programmes,
- a way of keeping Engineers conscious of their great responsibilities, by discussing difficult cases and providing expert ethical advice.

2 Why do we need "Software Engineering"?

Today, where bridges, engines, aircraft, power plants and medical devices are designed and/or controlled by software, the problems that led to the establishment of the engineering profession are now found in the field of software design. Many people present themselves as qualified to build software, but their products are full of problems. Few people receive an education that prepares them to develop robust and reliable software, and the general public has no way to evaluate the qualifications of those who present themselves as qualified experts.

Just as Chemical Engineering is a marriage of the science of chemistry with a lot of Engineering areas such as thermodynamics, mechanics, and fluid dynamics, the

Software Engineering field should be a marriage of the science of software with the older knowledge of the engineering profession.

The members of the Software Engineering profession, should know that subset of Computer Science that is relevant to software design, but they must also share the knowledge about design, mathematics, and other sciences that are traditionally known by Engineers. Over the years Engineering has split into a number of distinct specialities, each characterised by a distinct area of engineering science, but all sharing certain fundamental knowledge that is useful in all areas of Engineering. It is time that another such speciality, Software Engineering be identified and defined.

3 Issues worth discussing

The talk will discuss:

- Why there are no software engineers today,
- Why we have a software science and what it includes,
- The difference between an Engineering style of education and common Computer Science programmes,
- The differences between the Computer Science viewpoint and the Engineering viewpoint,
- The role of mathematics in Software Engineering,
- Some contrasts between Engineering Mathematics and Mathematics in Computer Science,
- Important parts of Computer Science that Engineers don't know,
- Engineering concepts outside conventional Computer Science that should be known to Software Engineers,
- What Engineering tells us about how to document software systems,
- Dilemmas faced by the Professional Engineer who uses software in professional practice,
- The "Professional Practice Exam" for the year 2001,
- Replacing disclaimers by warranties.

4 How to consummate the marriage

There are many important differences between the classical fields of Engineering and the new one that we are meeting to discuss. However, we will argue that there are more similarities than differences, and that Software Engineering is better understood as a branch of Engineering than as a branch of Computer Science. It is essential that those in "Software Engineering" learn more about classical Engineering and that those in classical Engineering recognise Software Engineering as a new branch of their profession.