

# Debates within software engineering

From Wikipedia, the free encyclopedia.

This article needs to be **cleaned up** to conform to a higher standard of quality.  
See How to Edit and Style and How-to for help, or this article's talk page.

Many debates are raging within the software engineering community. As software becomes more pervasive, most recognize the need for better software, but few agree on how to obtain it.

With about 612,000 software engineers in the U.S., and 1,400,000 more around the world, there should be room for many different opinions and approaches. A consensus has yet to emerge.

## Contents

- 1 Right to use the word engineering
- 2 Substance versus metaphor
- 3 Meanings of terms
- 4 Fighting over priorities

## Right to use the word engineering

The word *engineering* within the term software engineering causes a lot of confusion.

The wrangling over the status of software engineering (between traditional engineers and computer scientists) can be interpreted as a fight over control of the word engineering. Traditional engineers question whether software engineers can legally use the term.

Traditional engineers (especially civil engineers and the NSPE) claim that they have special rights over the term *engineering*, and for anyone else to use it requires their approval. In the mid-1990s, the NSPE sued to prevent anyone from using the job title *software engineering*. The NSPE won their lawsuit in 48 states.

However, SE practitioners, educators, and researchers ignored the lawsuits and called themselves software engineers anyway. The U.S. Bureau of Labor Statistics uses the term software engineer, too. The term engineering is much older than any regulatory body, so many believe that traditional engineers have few rights to control the term.

The fields of data engineering, knowledge engineering, user interface engineering, and so on have similar concerns about the term *engineering*. Even smaller or newer fields of biological engineering, safety engineering, and corrosion engineering have these concerns.

## Substance versus metaphor

Some believe that the name SE means that practitioners must also be traditional engineers. Others believe that engineering is only a metaphor that SEs should apply appropriately.

### Substance

Those who define software engineering as a branch of traditional engineering often believe that SEs apply concepts from traditional engineering to software development. This means that software engineering student should study physics, chemistry, and calculus; practitioners should earn professional licenses; and so on. They

believe engineering provides a structured, logical approach, and therefore, a stable final product.

### Metaphor

Others are inspired by traditional engineering, but believe that software needs its own solutions. They believe that many traditional engineering concepts cannot apply, because software is fundamentally different from bridges and roads. For example, traditional engineers do not use compilers or linkers to build roads. They believe that students should study computer science and other useful topics, and that practitioners do not necessarily need licenses.

## Meanings of terms

Prior to the mid-1990s, most software practitioners called themselves *programmers* or *developers*, regardless of their actual jobs. Many people prefer to call themselves *software developer* and *programmer*, because most widely agree what these terms mean, while *software engineer* is still being debated.

### Programmer

We widely agree what this means.

### Developer

We widely agree what this means.

### Software engineer

We disagree what this means.

The term *programmer* has often been used as a pejorative term to refer to those who lacked the tools, skills, education, or ethics to write quality software. In response, many practitioners called themselves *software engineers* to escape the stigma attached to the word *programmer*. In many companies, the titles *programmer* and *software developer* were changed to *software engineer*, for many categories of programmers.

These terms cause confusion, because some denied any differences (arguing that everyone does essentially the same thing with software) while others use the terms to create a difference (because the terms mean completely different jobs).

## Fighting over priorities

In the pursuit of better software, the community disagrees on priorities, approaches, and on what an individual should do in specific circumstances. Everyone seems to advocate a different combination of the following issues. Proponents and methodologists advocate conflicting solutions and often heatedly debate their merits. All subfields mix the following priorities to varying degrees.

### Management

Some advocate that software engineering is primarily about the management practices necessary to make reliable budgets and schedules. People at the Software Engineering Institute took this approach and created the CMM.

### Formal methods

Some advocate applying rigorous mathematical analysis to computer programming, especially proofs of correctness. They believe that traditional engineering is carried out with mathematical rigor, while programming is an iterative, trial-and-error process. These advocates strive to make programming more rigorous.

### Process

Some advocate that software engineers must follow step-by-step processes, much like assembly line workers. This inspired CMM, ISO 9000, RUP, SPICE, and other methods and processes.

### Tools

Some advocate that software engineering means tools, especially CASE tools (like Unix tools and IDEs) that emphasize high-level architecture issues. Today's CASE tools emphasize UML.

### Ethics

Some advocate that software engineering is mostly about codes of ethics and social responsibility. They sometimes argue that bugs are due to lapses of ethics.

### Licenses

Some advocate defining software engineering in terms of professional licenses, like some traditional engineers have. The biggest advocates of this position are from Texas and Canada, where the state governments sponsor licenses for SEs.

### Degrees

Some advocate defining SE by college degrees. Most professions have college degrees tailored to the needs of practitioners. Many graduate software engineering degrees are available and undergraduate degrees are becoming available.

### Attributes

Cost, Time, Quality : Different kinds of applications are sensitive to different attributes. Consumer applications are sensitive to cost. Military and medical applications are sensitive to quality. Business web applications are most sensitive to time. Some researchers argue that one attribute or another (usually quality) matters more than the others. But, software engineers work on all kinds of applications.

Retrieved from "[http://en.wikipedia.org/wiki/Debates\\_within\\_software\\_engineering](http://en.wikipedia.org/wiki/Debates_within_software_engineering)"

Categories: Wikipedia cleanup | Software engineering

---

- This page was last modified 23 April 2005 01:16.
- All text is available under the terms of the GNU Free Documentation License (see **Copyrights** for details).