Computer Software continues to be the single most important technology on the world stage.

Fifty years ago no one could have predicted that software would become an indispensable technology for business, science and engineering; that software would enable the creation of new technologies, (e.g. genetic engineering and nano technology), the extension of existing technologies (e.g. telecommunications) and the radical change in older technologies (e.g. printing industry).

Today, software takes on a dual role. It is a product and at the same time, the vehicle for delivering a product.

Software is an information transformer- producing, managing, acquiring, modifying, displaying, or transmitting information.

Why does it take so long to get software finished?

Why are development cost so high?

Why can't we find all errors before we give the software to our customers?

Why do we spend so much time and effort maintaining existing programs?

Whys do we continue to have difficulty in measuring progress as software is being developed and maintained?

Defining Software:

Software is: (1) instructions (computer programs) that when executed provide desired features, functions, and performance, (2) data structures that enable the program to adequately manipulate information, and (3) descriptive information in both hard copy and virtual forms that describes the operation and use of programs.

1. Software is developed or engineered; it is not manufactured in the classic sense.
2. Software doesn't "wear out." But it will deteriorate – due to changes being implemented over time.
3. Although the industry is moving towards component – based construction, most software continues to be custom built.

Software Application Domains:

Today, seven broad categories of computer software present continuing challenges for software engineers:

- System Software
- Application Software
- Engineering / Scientific Software
- Embedded Software
- Product-line software
- Web applications
- Artificial intelligent software

System software – a collection of programs written to service other programs. Some system software (e.g. compilers, editors and file management utilities) process complex, but determinate, information structures.

Other System Applications (e.g. operating system components, device drivers, network software, telecommunications processors) process largely indeterminate data.

In either case, system software area is characterized by heavy interaction with computer hardware; heavy usage by multiple users; concurrent operations that require scheduling, resource sharing, and sophisticated process management, complex data structures; and multiple external interfaces.

Application Software:

Stand-alone programs that solve a specific business need. Applications in this area process business or technical data in a way that facilitates business operations or management / technical decision making. In addition to conventional data processing applications, application software is used to control business functions in real-time (e.g. point-of-sale transaction processing, real-time manufacturing process control).

Engineering / Scientific Software:

Engineering and scientific software has been characterized by “number crunching” algorithms. Applications range from astronomy to volcanology, from automotive stress analysis to space shuttle orbital dynamics, and from molecular biology to automated manufacturing. However, modern applications within the engineering / scientific area are moving away from conventional numerical algorithms. Computer aided design, system simulation, and other interactive applications have begun to take on real-time and even software characteristics.

Embedded Software:

Embedded software resides within a product or system and is used to implement and control features and functions for the end user and for the system itself. Embedded software can perform limited and esoteric functions or provide significant function and control capability.

Product-Line Software:

Product-Line software designed to provide a specific capability for use by many different customers. Product-line software can focus on a limited and esoteric marketplace (e.g. inventory control products) or address mass consumer markets (e.g. word processing, spreadsheets, computer graphics, multimedia, entertainment, database management, and personal and business financial applications).

Web Applications:

Web applications called "WebApps," this network-centric software category spans a wide array of applications. In their simplest form, webapps can be little more than a set of linked hypertext files that present information using text and limited graphics. However, as web 2.0 emerges webapps are evolving into sophisticated computing environments that not only provide stand-alone features, computing functions, and content to the end user, but also are integrated with corporate databases and business applications.

Artificial Intelligence Software:

Artificial intelligence software make use of non-numerical algorithms to solve complex problems that are not amenable to computational or straight forward analysis. Applications within this area include robotics, expert systems, pattern recognition (image and voice), artificial neural networks, theorem proving and game playing.

Millions of software engineers world wide are hard at work on software projects in one or more of these categories.

New Challenges:

Open-world computing – the rapid growth of wireless networking may soon lead to true pervasive, distributed computing. The challenge for software engineers will be to develop systems and application software that will allow mobile devices, personal computers, and enterprise systems to communicate across vast networks.

Net-sourcing – the world wide web is rapidly becoming a computing engine as well as a content provider. The challenge for software engineers is to architect simple and sophisticated applications that provide a benefit to target end-user market world wide.

Open Source – a growing trend that results in distribution of source code for system applications so that many people can contribute to its development.

Legacy Software:

Hundreds of thousands of computer programs fall into one of the seven broad application domains. Some of these are state of the art software – just released to individuals, industry, and government. But other programs are older, in some cases much older.

> Legacy software systems were developed decades ago and have continually modified to meet changes in business requirements and computing platforms. The proliferation of such systems is causing headaches for large organizations who find them costly to maintain and risky to evolve.

Unfortunately, there is sometimes one additional characteristic that is present in legacy software – poor quality.

As time passes, legacy systems often evolve for one or more of the following reasons:

- The software must be adapted to meet the needs of new computing environments or technology.
- The software must be enhanced to implement new business requirements.
- The software must be extended to make it interoperable with other more modern systems or databases.
- The software must be re-architected to make it viable within a network environment.

The Unique Nature of WebApps:

In the early days of the world wide web, websites consisted of little more than a set of linked hyper-text files that presented information using text and limited graphics.

As time passed, the augmentation of HTML by development tools (e.g. XML, Java) enabled web engineers to provide computing capacity at the network system.