Software Project Management

Lecture 1

- **1968:** The notion of 'software engineering' was first proposed at a conference held to discuss what was called the "software crisis".

Software Crisis

YEAR	COMPANY	OUTCOME (COSTS IN US $)
2005	Hudson Bay Co. [Canada]	Problems with inventory system contribute to $33.3 million* loss.
2004–05	UK Inland Revenue	Software errors contribute to $3.45 billion* tax-credit overpayment.
2004	Avis Europe PLC [UK]	Enterprise resource planning (ERP) system canceled after $54.5 million† is spent.
2004	Ford Motor Co.	Purchasing system abandoned after deployment costing approximately $400 million.
2004	J Sainsbury PLC [UK]	Supply-chain management system abandoned after deployment costing $527 million.†
2004	Hewlett-Packard Co.	Problems with ERP system contribute to $160 million loss.
2003–04	AT&T Wireless	Customer relations management (CRM) upgrade problems lead to revenue loss of $100 million.
2002	McDonald's Corp.	The Innovate information-purchasing system canceled after $170 million is spent.
2002	Sydney Water Corp. [Australia]	Billing system canceled after $33.2 million† is spent.
2002	CIGNA Corp.	Problems with CRM system contribute to $445 million loss.
2001	Nike Inc.	Problems with supply-chain management system contribute to $100 million loss.
2001	Kmart Corp.	Supply-chain management system canceled after $130 million is spent.
2000	Washington, D.C.	City payroll system abandoned after deployment costing $25 million.
1999	United Way	Administrative processing system canceled after $12 million is spent.
1999	State of Mississippi	Tax system canceled after $11.2 million is spent; state receives $185 million damages.
1999	Hershey Foods Corp.	Problems with ERP system contribute to $151 million loss.
1998	Snap-on Inc.	Problems with order-entry system contribute to revenue loss of $50 million.

Year	Organization	Description
1997	U.S. Internal Revenue Service	Tax modernization effort canceled after $4 billion is spent.
1997	State of Washington	Department of Motor Vehicle (DMV) system canceled after $40 million is spent.
1997	Oxford Health Plans Inc.	Billing and claims system problems contribute to quarterly loss; stock plummets, leading to $3.4 billion loss in corporate value.
1996	Arianespace [France]	Software specification and design errors cause $350 million Ariane 5 rocket to explode.
1996	FoxMeyer Drug Co.	$40 million ERP system abandoned after deployment, forcing company into bankruptcy.
1995	Toronto Stock Exchange [Canada]	Electronic trading system canceled after $25.5 million** is spent.
1994	U.S. Federal Aviation Administration	Advanced Automation System canceled after $2.6 billion is spent.
1994	State of California	DMV system canceled after $44 million is spent.
1994	Chemical Bank	Software error causes a total of $15 million to be deducted from 100 000 customer accounts.
1993	London Stock Exchange [UK]	Taurus stock settlement system canceled after $600 million** is spent.
1993	Allstate Insurance Co.	Office automation system abandoned after deployment, costing $130 million.
1993	London Ambulance Service [UK]	Dispatch system canceled in 1990 at $11.25 million**; second attempt abandoned after deployment, costing $15 million.**
1993	Greyhound Lines Inc.	Bus reservation system crashes repeatedly upon introduction, contributing to revenue loss of $61 million.
1992	Budget Rent-A-Car, Hilton Hotels, Marriott International, and AMR [American Airlines]	Travel reservation system canceled after $165 million is spent.

Software & Economies

- The worldwide software industry in 2000 was $300 billion
- The worldwide packaged software industry for all platforms was $220 billion in 2005
- United States
  - The United States is estimated to hold just under a 50% share of the world market.
  - The packaged software market in the United States is estimated at nearly $90 billion - of this, $82 billion is business software, $5.5 billion is home use products and $600 million worth of software is designed for schools.
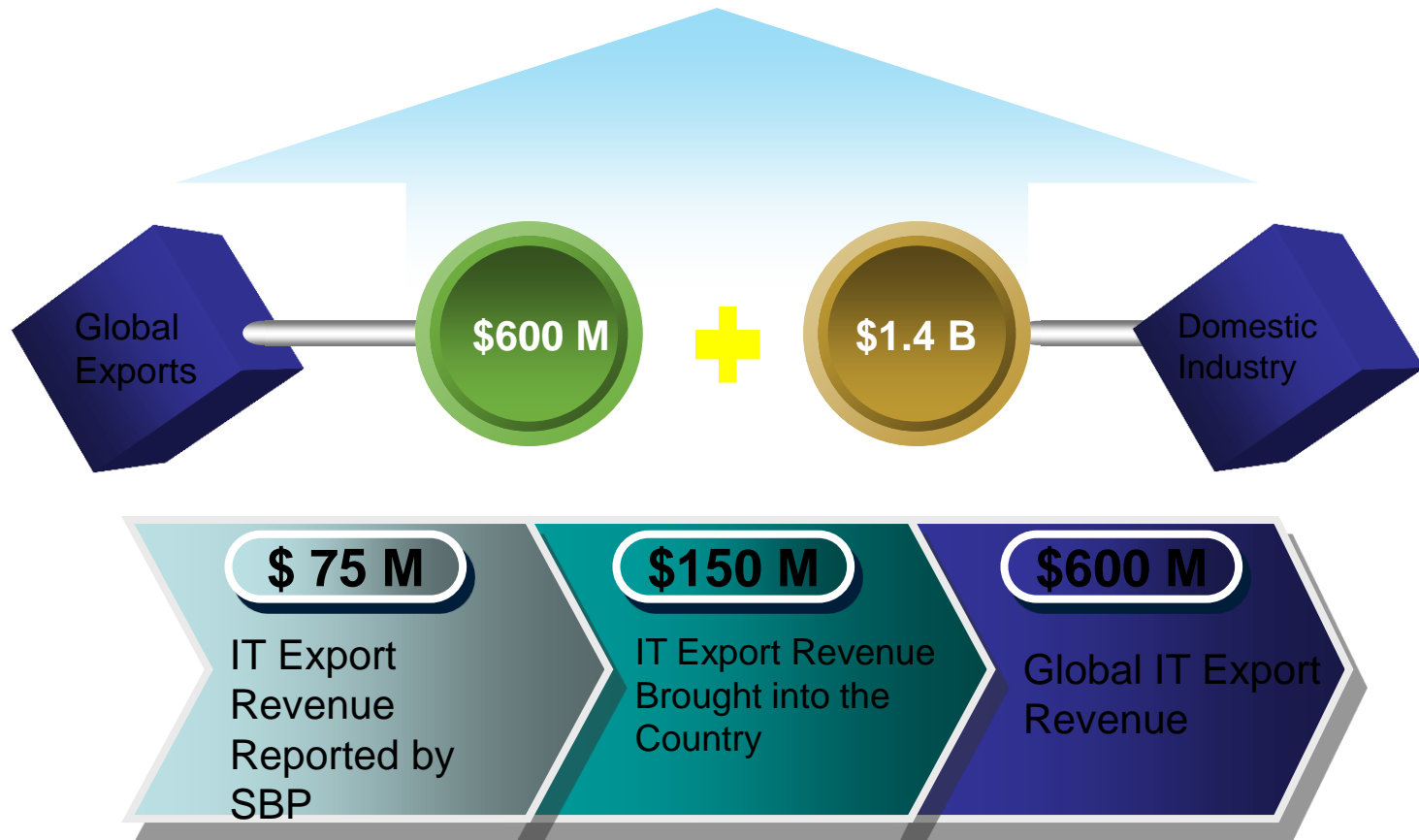
Software & Economies

- INDIA
  - 1999-2000: $5.7 billion (34% growth, 10% of Total Exports)*
  - 2006-2007: $31.4 billion in the software and services exports
  - 2007-2008: $40.4 billion (a growth of 29% from 2006-2007)

Pakistan : Software & Economies

2000: Pakistan's share in the global software market is $60 million
(Business Recorder: Dec 13, 2000)

Total Industry Size: $ 2 Billion

Global Exports

$600 M + **$1.4 B**

Domestic Industry

$ 75 M
IT Export Revenue Reported by SBP

$150 M
IT Export Revenue Brought into the Country

$600 M
Global IT Export Revenue

Why Software Project Fail?

- Management problems were more frequently dominant cause than technical problems
- Schedule overruns were more common (89%) than cost overruns (62%)

  KPGM's Survey in UK

- Primary causes of software runaway
  - Project Objectives not fully specified
  - Bad planning and estimating
  - Technology new to the organization
  - Inadequate/No project management methodology
  - Insufficient senior staff on the team
  - Poor performance by Supplier of hardware/software

What makes a software project success?

- User involvement – 20 points
- Executive Support – 15 points
- Clear Business Objectives – 15 points
- Experienced Project Manager – 15 points
- Small milestones – 10 points
- Firm basic requirements – 5 points
- Competent staff – 5 points
- Proper planning – 5 points
- Ownership – 5 points
- Others – 5 points

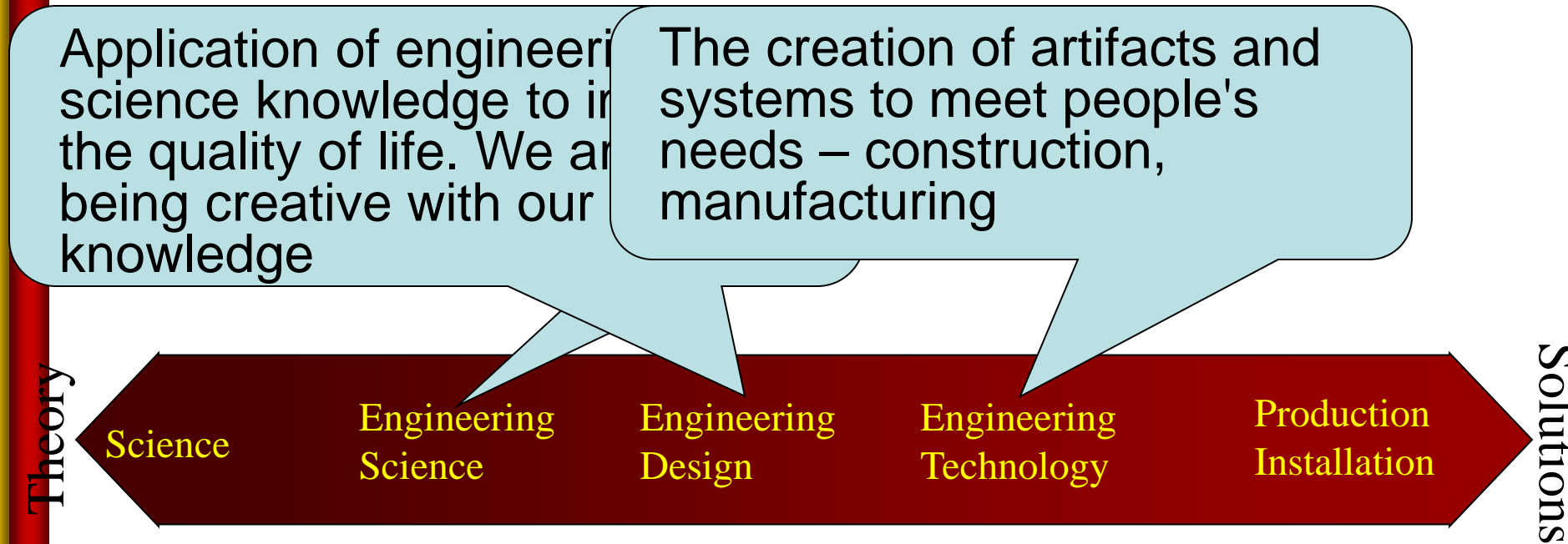Chaos ten – Standish Group Report

Software Project Management **FIRST**

- "Organizations that attempt to put software engineering discipline in place before putting project management discipline in place are domed to fail"
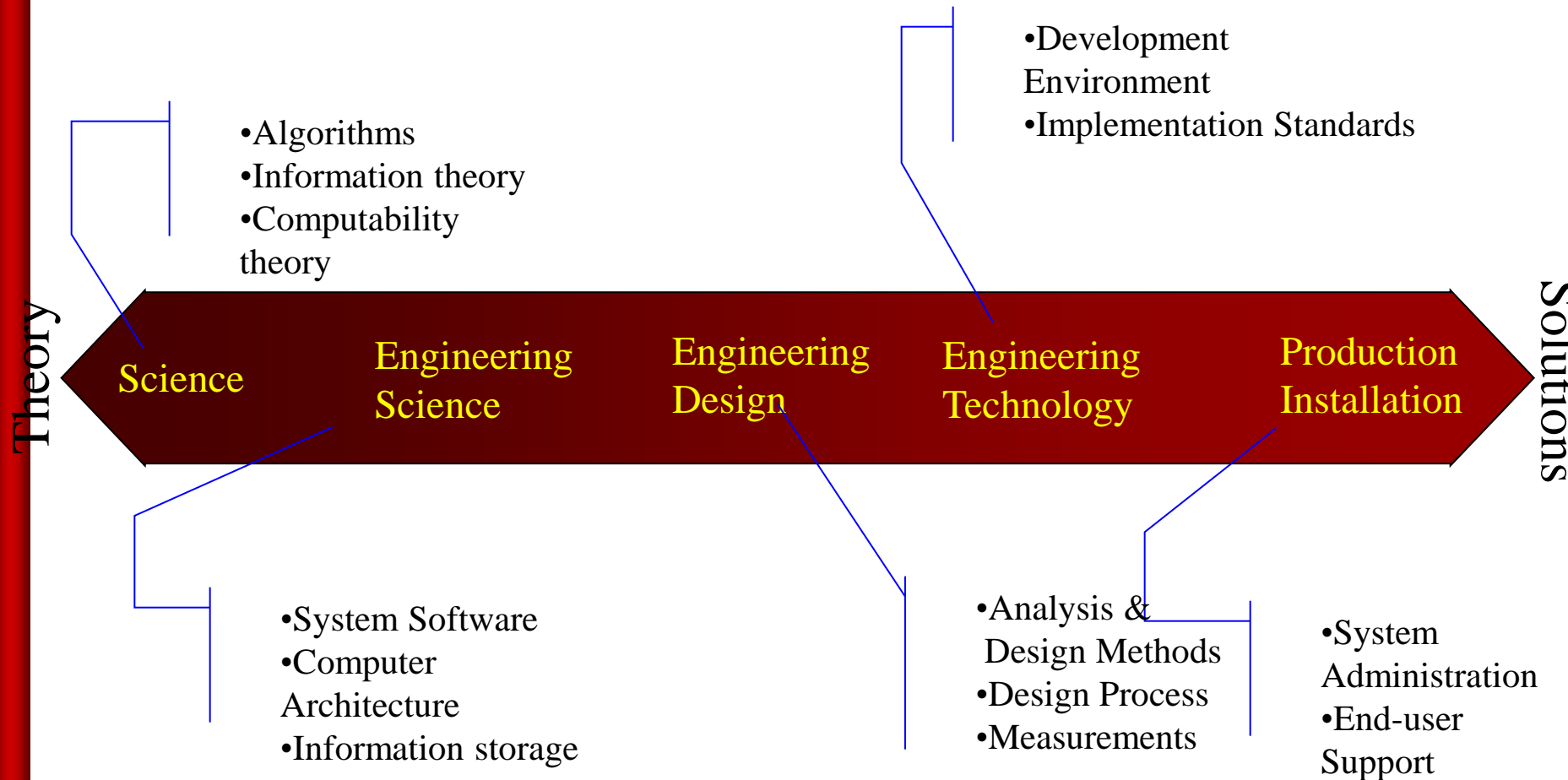
  Software Engineering Institute

What is Software Engineering?

- Software Engineering is an engineering discipline
- Concerns with the development of software by applying Engineering Principles
- Goal is the cost-effective development of software systems
- Software Engineering [IEEE-93]*:
  - The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software.
  - The study of approaches as in 1

- *IEEE-93: IEEE Standard Glossary & Software Engineering Terminology

Knowledge Spectrum

Application of engineeri[ng] science knowledge to i[mprove] the quality of life. We ar[e] being creative with our knowledge

The creation of artifacts and systems to meet people's needs – construction, manufacturing

Theory

Science

Engineering Science

Engineering Design

Engineering Technology

Production Installation

Solutions

A Computing Spectrum

•Algorithms
•Information theory
•Computability theory

•Development Environment
•Implementation Standards

Theory

Science | Engineering Science | Engineering Design | Engineering Technology | Production Installation

Solutions

•System Software
•Computer Architecture
•Information storage

•Analysis & Design Methods
•Design Process
•Measurements

•System Administration
•End-user Support

Computing Professions

•Creates new applications applying engineering principles and CS best practices

Software Engineer

Theory

Science
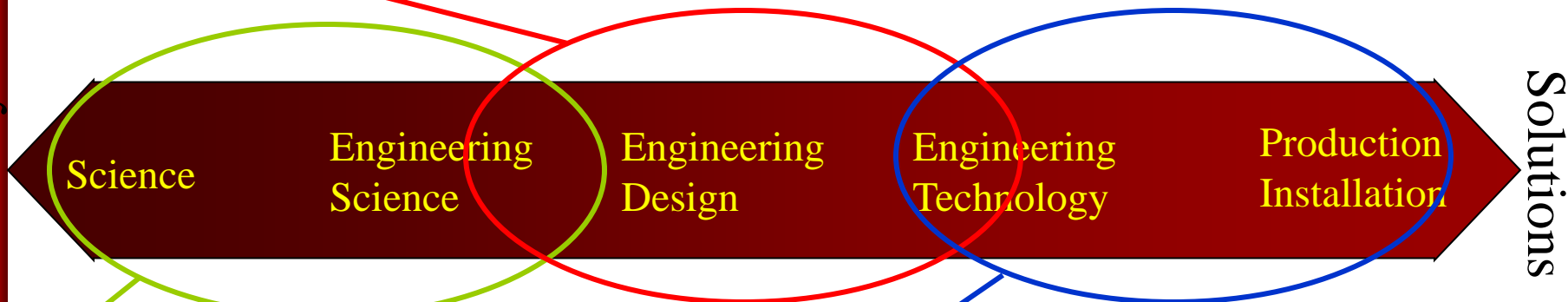
Engineering Science

Engineering Design

Engineering Technology

Production Installation

Solutions

Computer Scientist

Information Technologist

•Creates New Infrastructure

•Provides end-user solutions

"Software Engineering"
Barry Boehm ICSE 2006 Keynote Address May 25, 2006

- SE definition (Based on Webster definition of "engineering")
  The application of science and mathematics by which the properties of software are <u>made useful to people</u>

- Includes computer science and the sciences of making things useful to people
  – Behavioral sciences, economics, management sciences

What is Software Engineering?

- What is Software?
- What is Engineering?

The Nature of Software

F.P. Brooks, "No Silver bullet: Essence and Accident of Software Engineering", IEEE Computer Magazine, April 1987

No Silver bullet by F. P. Brook

- Motivation of Paper
  - Missed schedule, blown budgets and flawed products
  - Something to make software cost drop as rapidly as computer hardware cost do
  - "We look to the horizon of a decade hence we see **no silver bullet.** There is no single development, in either technology or in management technique that by itself promises even one order-of-magnitude improvement in productivity, in reliability, in simplicity"

Nature of software

- Not only are there no silver bullets now in view, the very nature of software makes it unlikely that there will be any

- **Essence**
  - The difficulties inherent in the nature of software

- **Accidents**
  - Those difficulties that today attend its production but are not inherent

- The essence of a software entity is a construct of interlocking concepts (conceptual construct)
- I believe that hard part of building software to be the specification, design, and testing of this conceptual construct, not the labor of representing it and testing the fidelity of the representation. We still make syntax errors, to be sure; but they are fuzz compared with the conceptual errors in most systems

Essence and Accident of SE

- **Irreducible essence of modern software**
  - Complexity
  - Conformity
  - Changeability, and
  - Invisibility.

Complexity

- No two parts are alike
- Software systems have many times more states even than computer hardware.
- Scaling-up of software entity is not merely a repetition of the same elements in larger sizes. It is necessarily an increase in the number of different elements. In most cases, the elements interact with each other in some non-linear fashion, and the complexity of the whole increases much more than linearly.

Complexity

- The complexity of software in an essential property, not an accidental one.

- Hence, descriptions of a software entity that abstract away its complexity often abstract away its essence.

- Models are abstractions – simplification of reality

Complexity

- From complexity comes the difficulty of
  - Communication among team members which leads to product flaws, cost overrun, schedule delays.
  - Enumerating
  - Less understanding of all the possible states of the program

Conformity

- Natural Science – God VS Software – People
- Much of the complexity that he (developer) must master is arbitrary complexity, forced without rhyme or reason by many human institutions and systems to which his interfaces must conform.
- These (standards) differ from interface to interface, and from time to time, not because of necessity but only because they were designed by different people, rather than by God.

Changeability

- The software entity is constantly subject to pressures for change. Of course, so are buildings, cars, and computers. But manufactured things are infrequently changed.

- Software of a system embodies its function, and the function is the part that most feels the pressure of change

- In part it is because software can be changed more easily – it is pure thought stuff, infinitely malleable

- Building do in fact get change but the high cost of change understood by all serve to dampen the whims of the changes
- The software product is embodied in a cultural matrix of applications, users, laws, and machine vehicles. These change continually, and their changes inevitably force change upon the software product.

Invisibility

- Software is invisible and un-visualizable.
- A geometric reality is captured in a geometric abstraction
- The reality of software is not inherently embedded in space
- As soon as we attempt to diagram software structure, we find it to constitute not one, but several, general directed graphs superimposed one upon another.
- This lack not only obstruct the process of design within one mind, it severely hinders communication among minds.

Engineering

- Derived from old French engin = skill, which stems from Latin ingenium = ability to invent, brilliance, genius
- The word was created in the 16th century and originally described a profession that we would probably call an artistic inventor *("Encyclopedia" by The Software Toolworks, 1991)*
- Engineering is applied science
- Engineering is the application of science for practical purposes
- Engineers put theory into practice.

Engineering

- ABET, USA
  - 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 material and forces of nature for the benefit of mankind

- The profession of or work performed by an engineer. Engineering involves the knowledge of the mathematical and natural sciences (biological and physical) gained by study, experience, and practice that are applied with judgment and creativity to develop ways to utilize the materials and forces of nature for the benefit of mankind (International Technology Education Association).

Engineering Software

- Engineering methods put a lot of emphasis on planning before you build
- Design, Plan and then Construct
- Construction is much bigger in both cost and time than design and planning
  - 80-90%
- What is the proportion in Software Engineering?
  - 10-30%
  - Software construction is so cheap as to be free
  - All effort is of design

Engineering Software

- Design requires creative and talented people
- Creative processes are not easily planned, and so predictability may well be an impossible target
  - Everything else in software development depends on the requirements. If you cannot get stable requirements you cannot get a predictable plan
- This raises an important question about the nature of design in software compared to its role in other branches of engineering
- "We should be very wary of the traditional engineering metaphor for building software. It's a different kind of activity and requires a different process" (Martin Fowler, The New methodology)

Reading Material

- "Software's Chronic Crisis", W. Wayt Gibbs, Scientific American, Vol. 271, No. 3, September 1994

  OR

- "Who killed the Virtual Case File", Harry Goldstien, IEEE Spectrum, September 2005

Q&A