Chapter 1
Software and Software Engineering

Software Engineering: A Practitioner's Approach, 6th edition
by Roger S. Pressman

Software's Dual Role

- Software is a product
  - *Transforms* information - produces, manages, acquires, modifies, displays, or transmits information
  - Delivers computing potential of hardware and networks
- Software is a vehicle for delivering a product
  - Controls other programs (operating system)
  - Effects communications (networking software)
  - Helps build other software (software tools & environments)

Software Applications

- system software
- application software
- engineering/scientific software
- embedded software
- product-line software
- web applications
- AI software

Hardware vs. Software

Hardware

- Manufactured
- Wears out
- Built using components
- Relatively simple

Software

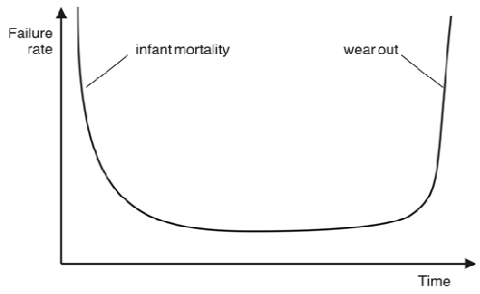
- Developed/engineered
- Deteriorates
- Custom built
- Complex

Manufacturing vs. Development

- Once a hardware product has been manufactured, it is difficult or impossible to modify. In contrast, software products are routinely modified and upgraded.
- In hardware, hiring more people allows you to accomplish more work, but the same does not necessarily hold true in software engineering.
- Unlike hardware, software costs are concentrated in design rather than production.

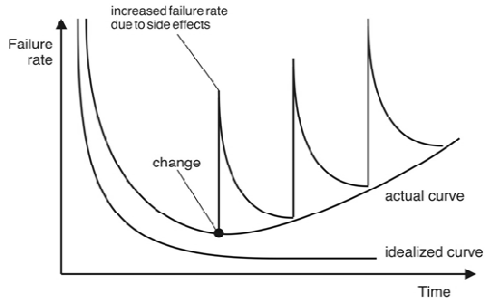
Wear vs. Deterioration

Hardware wears out over time



Wear vs. Deterioration

Software deteriorates over time



Component Based vs. Custom Built

- Hardware products typically employ many standardized design components.
- Most software continues to be custom built.
- The software industry does seem to be moving (slowly) toward component-based construction.

Software Complexity

I believe the 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.

If this is true, building software will always be hard. There is inherently no silver bullet.

- Fred Brooks, "No Silver Bullet"

http://www.computer.org/computer/homepage/misc/Brooks/

Legacy Software

Why must it change?

It must be fixed to eliminate errors.
It must be enhanced to implement new functional and non-functional requirements
Software must be adapted to meet the needs of new computing environments or technology.
Software must be enhanced to implement new business requirements.
Software must be extended to make it interoperable with other more modern systems or databases.
Software must be re-architected to make it viable within a network environment.

Software Myths

- Affect managers, customers (and other non-technical stakeholders) and practitioners
- Are believable because they often have elements of truth,

but …

- Invariably lead to bad decisions,

therefore …

- Insist on reality as you navigate your way through software engineering

Software Myths

- If we get behind schedule, we can add more programmers and catch up.
- A general statement about objectives is sufficient to begin building programs.
- Change in project requirements can be easily accommodated because software is flexible.

Software Myths

- Once we write a working program, we're done.
- Until I get the program running, I have no way of assessing its quality.
- The only deliverable work product for a successful project is the working program.
- Software engineering will make us create too much documentation and will slow us down.

Management Myths

- "We already have a book of standards and procedures for building software. It does provide my people with everything they need to know …"
- "If my project is behind the schedule, I always can add more programmers to it and catch up …"
- "If I decide to outsource the software project to a third party, I can just relax: Let them build it, and I will just pocket my profits …"

Customer Myths

- "A general statement of objectives is sufficient to begin writing programs - we can fill in the details later …"
- "Project requirements continually change but this change can easily be accommodated because software is flexible …"

Practitioner's Myths

- "Let's start coding ASAP, because once we write the program and get it to work, our job is done …"
- "Until I get the program running, I have no way of assessing its quality …"
- "The only deliverable work product for a successful project is the working program …"
- "Software engineering is baloney. It makes us create tons of paperwork, only to slow us down …"