

## TOPICS

The Nature and History of Software  
Development

**Problems with Software Development**

Software Engineering Paradigms and  
Technology

## **PROBLEMS WITH SOFTWARE DEVELOPMENT**

- **Problems**
- **Causes**

## Problems

1. We have little data on the software development process.
2. Customers are often dissatisfied with the software they get.
3. Software quality is hard to define and measure.
4. Existing software is often very difficult to maintain.

Can these problems be overcome?

1B - 3

- Historically, measurements of software development were not done, so we did not gather any data from past experiences to use in predicting the schedules and costs of future projects.
- Customer needs were usually understood only vaguely. Consequently, programs often fell short of the customers' desires.
- A solid quantification of the metrics associated with the software does not exist, so it becomes difficult to predict software quality.
- Maintenance has become the most expensive, difficult, and poorly planned task of the entire software life cycle.

## **Causes**

- **No spare parts to replace, so an error in the original software is also in every copy.**
- **Software quality is a human problem.**
- **Project managers often have no software development experience.**
- **Software developers often have little or no formal training in engineering the development of the software product.**
- **Resistance to change from programming as an art to programming as an engineering task can be significant.**

## **SOFTWARE MYTHS**

- **Customer Myths**
- **Developer Myths**
- **Management Myths**

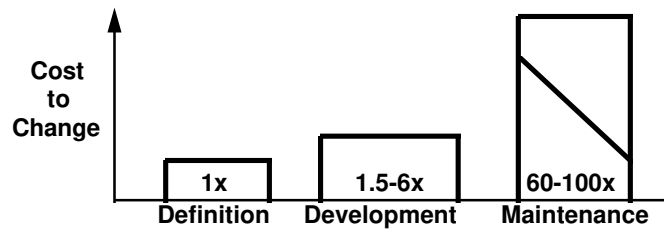
## Customer Myths

### Myth

- A general statement of objectives is enough to get going. Fill in the details later.
- Project requirements continually change, but change can be easily accommodated because software is flexible.

### Reality

- Poor up-front definition of the requirements is *THE* major cause of poor and late software.
- Cost of the change to software in order to fix an error increases dramatically in later phases of the life of the software.



## Developer Myths

### *Myth*

- Once a program is written and works, the developer's job is done.
- Until a program is running, there is no way to assess its quality.
- The only deliverable for a successful project is a working program.

### *Reality*

- 50%-70% of the effort expended on a program occurs after it is delivered to the customer.
- Software reviews can be more effective in finding errors than testing for certain classes of errors.
- A software configuration includes documentation, regeneration files, test input data, and test results data.

## Management Myths

### *Myth*

- Books of standards exist in-house so software will be developed satisfactorily.
- Computers and software tools that are available in-house are sufficient.
- We can always add more programmers if the project gets behind.

### *Reality*

- Books may exist, but they are usually not up to date and not used.
- CASE tools are needed but are not usually obtained or used.
- "Adding people to a late software project makes it later." -- *Brooks*