Software Quality Concepts

Software Quality Assurance

- How can you tell if software has high quality?
- How can we measure the quality of software?
- How can we make sure software has high quality?

Perspective on quality

- Customer
  - system does not crash
  - system follows documentation
  - system is logical and easy to use
- Developer
  - system is easy to change
  - system is easy to understand
  - system is pleasant to work on

Mistakes about SQA

- Wrong Concepts
  - Quality is conformance to requirements
  - Variation control is the heart of quality control.
- Main Philosophy
  - Feedback and continual improvement is the real heart of quality software.
  - Make sure that the standard is being maintained

Total Quality Management

- Goal is for every item coming off the assembly line to be perfect
- Management, production, engineering, QA
- Everyone is involved in quality
- Develop a reliable, repeatable process
- Continuously improve the process

Failure vs. flaw

- Failure - program didn't work right
- Flaw - mistake in the code of the program
- Failure analysis - what flaw caused this failure?
- Flaw analysis - what is wrong with our process that allowed this flaw to be created and not detected?

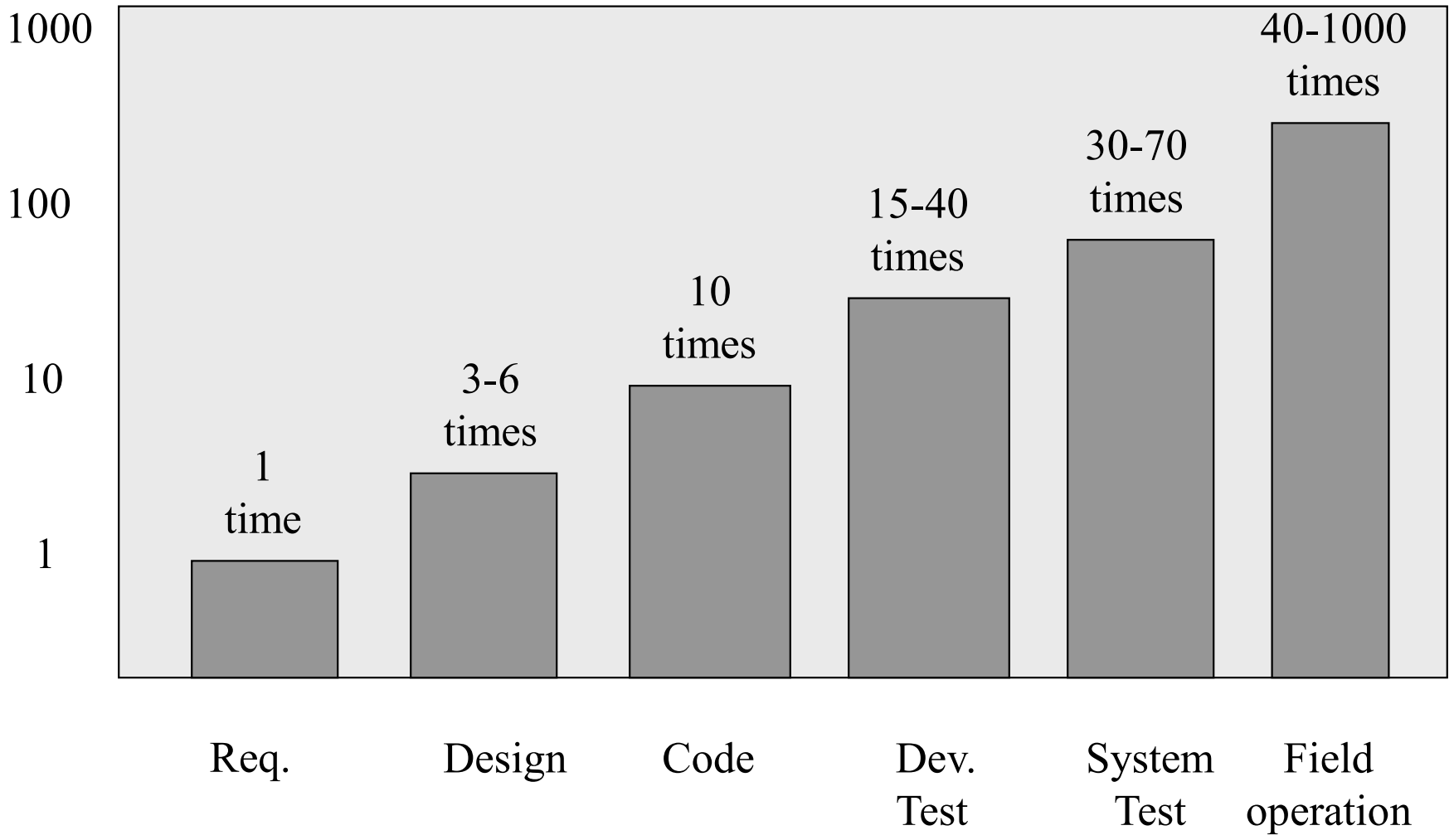
Failure costs

- Internal
  - rework
  - repair
  - failure analysis
- External
  - resolving complaints
  - returning and replacing product
  - help line

Prevention costs

- Prevention
  - planning
  - managing and collecting information
  - reviews
- Appraisal
  - inspection
  - testing

Cost of fixing an error



How to appraise quality

- Requirements
  - reviews by customers
  - prototyping
- Analysis and design models
  - formal reviews, inspections
- Current system
  - bug reports
  - user tests
  - surveys

Technical Reviews

- A way to evaluate the quality of requirements, designs, and software
- A way to improve the quality of requirements, designs, and software
- A way to educate new developers and ensure that developers are consistent
- *Proven to be cost-effective!*

Bug tracking

- Keep track of
  - who reported the bug (the failure)
  - description of the failure
  - severity
  - the flaw that caused this failure
  - who is repairing it
  - the repair

Bug tracking

- Use information about failures to estimate reliability
- Compare
  - critical nature of failure
  - recurrence of similar types of failure
  - module that had the flaw

Bug tracking

- Discover the flaw (defect) that caused each bug
- Categorize flaws
- Look at categories with the most flaws and improve your process to eliminate them.

Use quality information to make decisions

- Quality Information
  - Level of the failures based of severity
  - Level 1 might be users essential requirements
  - Level 2 might be the cosmetic errors
- "Must repair all level 1 failures before shipping"
- "Half of all level 1 and 2 failures in the alpha release were in the Call Processing module; we should rewrite it."
- "Half of all level 1 and 2 defects found in the design reviews were in Call Processing; we should rewrite it."

Ways not to improve quality

- Say “Be more careful!”
- Say “Quality is important.”
- Find out whose fault it is and fire him.

How to improve quality

- Measure and compare
- Determine root cause of problems
- Create ways to eliminate problems