Scope of Software Engineering (Schach Ch1)

- Software Engineering's aim:
  - produce fault-free SW meets user's needs
  - delivered on time/in budget
  - Easy to modify when user's needs change

- Historical Aspects: 1968 NATO Conference
  - Goal: to solve the “Software Crisis”
  - Overlooked: bridges not same as software

- **Question**: Is "Software Engineering" the same as “Engineering?” How are bridge building and software development similar, how are they different?

ICE: Building Bridges vs. Building Software

Issue	Bridge	Software
Complexity *(Maturity of Field)*	Bridges around since a tree fell across a stream. Fundamentals of bridge design don't change rapidly	50 years (only) of software development. Rapidly evolving principals
(Im)perfect engineering *(Expected Conditions)*	Built to withstand all *expected* conditions	
Maintenance *(Scale)*	Remove rust and paint, wouldn't even consider rotating 90 degrees	?
Collapse *(Fix or replace)*	Rebuild rather than repair	?

ICE: Building Bridges vs. Building Software

Issue	Bridge	Software
Complexity *(Maturity of Field)*	Bridges around since a tree fell across a stream. Fundamentals of bridge design don't change rapidly	50 years (only) of software development. Rapidly evolving principals
(Im)perfect engineering *(Expected Conditions)*	Built to withstand all *expected* conditions	Attitude: cannot anticipate all *unexpected* conditions
Maintenance *(Scale)*	Remove rust and paint, wouldn't even consider rotating 90 degrees	?
Collapse *(Fix or replace)*	Rebuild rather than repair	?

"Classical" Software Life Cycle

Series of Development Steps, from Concept Exploration through Final Retirement, Broken into 6 Phases:

- **Requirements phase** (concept explored, includes rapid prototyping)
- **Specification/Analysis phase** (contract)
- **Design phase**
  - high-level (architectural design => modules)
  - detailed (design of each module)
- **Implementation phase** (coding/testing)
  - Unit testing
  - Integration of sub-systems
- **Maintenance phase** (any changes after acceptance)
- **Retirement**

Spending on software development

- ICE: How much would you plan to spend assuming you have a $100k budget to develop/deliver a product? i.e. ignore post-delivery maintenance & retirement
  - Phases/Groups:
    - Requirements & Analysis
    - Design
    - Implementation: Coding & Unit Testing
    - Implementation: Integration

  - How does your plan compare with industry averages?

Spending on software development (con't)

- ICE Continued: Assuming that you spent the $100k to develop/deliver a product…
  How much additional money would you need to budget for post-delivery maintenance?

  - How does your plan compare with industry averages?

Types of Software Maintenance

- ***Corrective*** maintenance (fixing bugs)
- ***Perfective*** maintenance (improved functionality)
- ***Adaptive*** maintenance (changing environment)

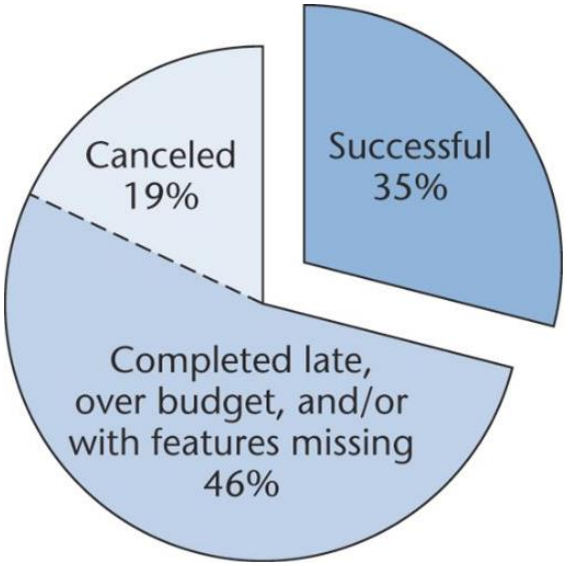
Different reactions to finding a software bug

Question: Which type(s) of maintenance can be improved through better requirements, specification, design, implementation, testing?

Recent Studies on State of Sys Analysis & Design

- Standish Study (2000) reviewed 28,000 projects =>

- Standish Study (2006) reviewed 9,000 projects =>



Recent Studies on State of Sys Analysis & Design

- Cutter Study (2002): 78% of projects have been involved in disputes ending in litigation. Of those cases:
  - In 67%, the functionality of the information system as delivered did not meet up to the claims of the developers
  - In 56%, the promised delivery date slipped several times
  - In 45%, the defects were so severe that the system was unusable

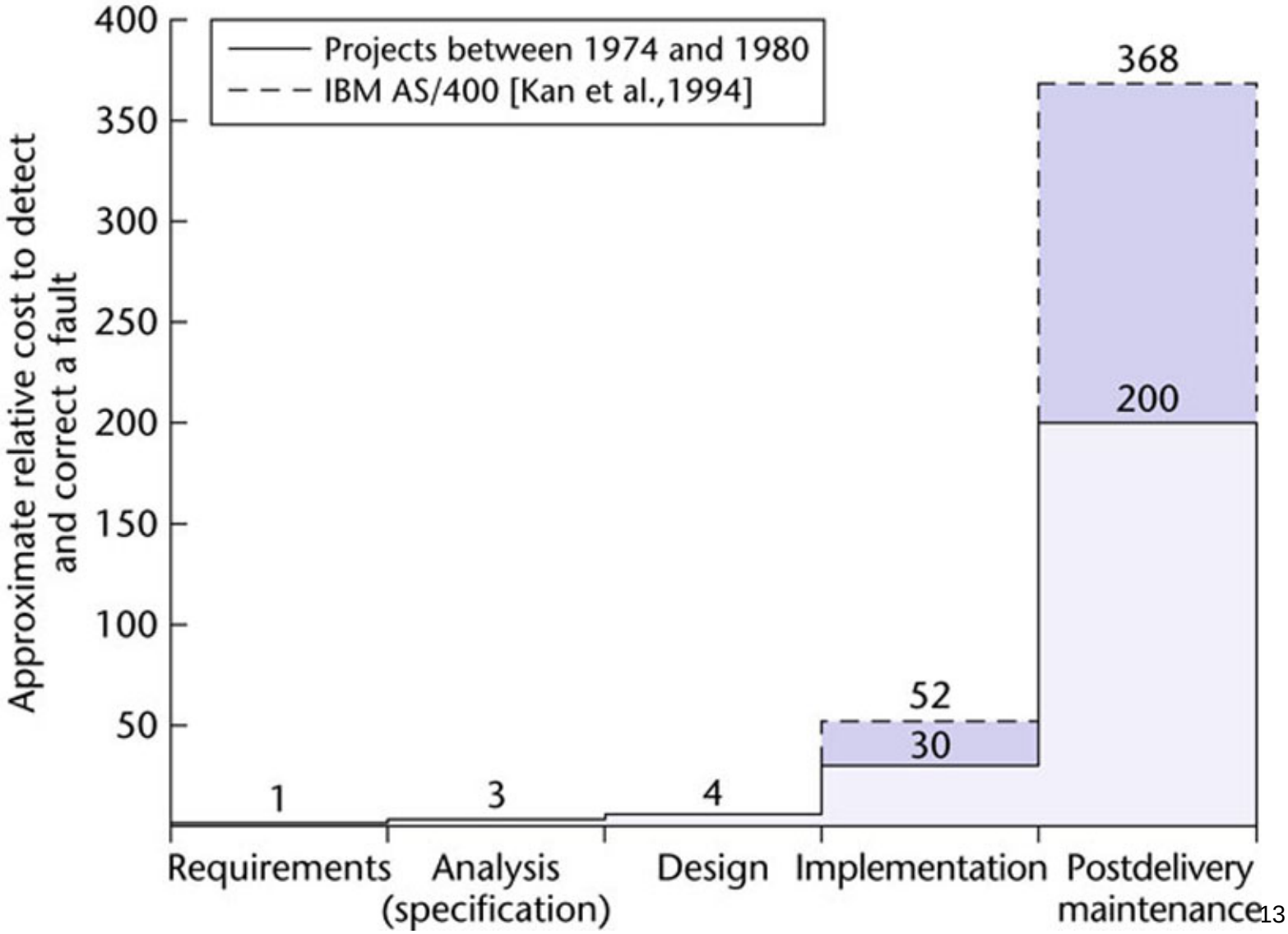
Where to focus efforts + reduce software costs?

- 60-70% of Faults: Specification/Design Faults

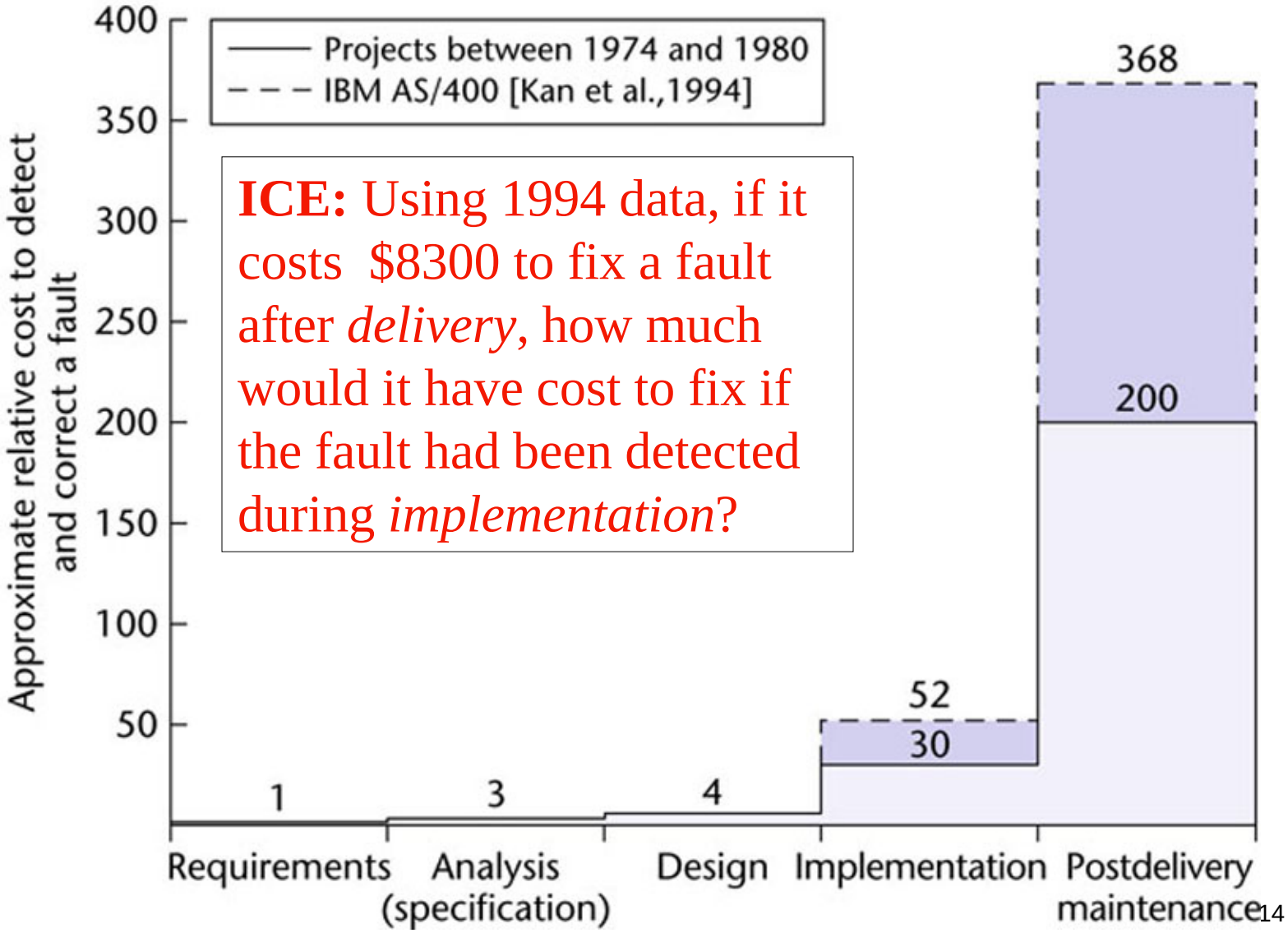
- Kelly, Sherif, and Hops [1992]
  - 1.9 faults per page of specification
  - 0.9 faults per page of design
  - 0.3 faults per page of code

- Bhandari[1994]: Faults at end of design phase of **new** version of product
  - 13% of faults from previous version of product
  - 16% of faults in new specifications
  - 71% of faults in new design

*Cost to Detect and Correct a Fault



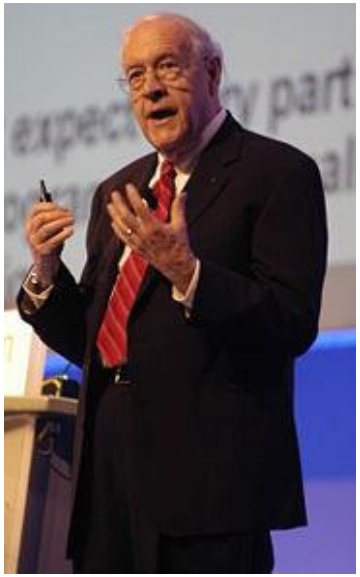
*Cost to Detect and Correct a Fault



ICE: Using 1994 data, if it costs $8300 to fix a fault after *delivery*, how much would it have cost to fix if the fault had been detected during *implementation*?

*Aspects of Team Programming

- Hardware (relatively) inexpensive — lead to increased demand for SW too large for one person to write in available time.
- Brooks paper [1975] on the *Mythical Man-Month:*
  - Single Programmer => delivery in 1 year
  - Team of 6 Programmers => delivery in ?
  - Quality of work ?



- What's the Difficulty with Teams?

Programmer vs. Software Engineer

- **Responsibilities:** Recent Ad for a Software Engineer (NASA Goddard, Greenbelt, MD) *not that Ken Jennings is looking*:

  - **Determine** embedded system **requirements**, prepare **specification**.
  - **Design** and develop software using object-oriented methods.
  - Perform **unit testing**. Maintain documentation. Assist with integration and testing.
  - **Assess risk** and propose design changes. Perform regression testing.
- **Required Skills:**
  C, C++, Java, Real Time embedded systems
  Education: B.S. – CS/IT/SE
- Clearance: Secret, existing clearance preferred