



# CS5127/6027: Requirements Engineering (Fall 2024)

Prof. Nan Niu ([nan.niu@uc.edu](mailto:nan.niu@uc.edu))

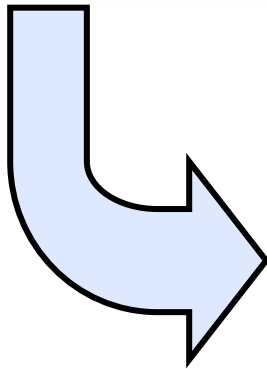
Office Hours: 10am-11am, Mondays, Rhodes 832



# Today's Menu

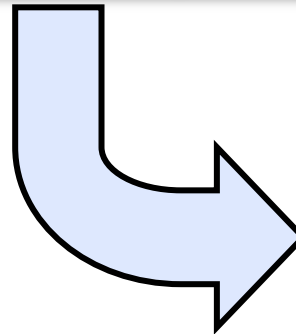
Last Lecture (Friday 9/6):

Meaning of req.s  
Release ASN1



This Lecture (Monday 9/9):

Importance of req.s  
Eliciting req.s



Next Lecture (Friday 9/13):

Elicitation techniques



# Recap the first two weeks

## → Week #1

- ↳ Requirements = stakeholders' needs and desires
- ↳ Stakeholders = those who have a stake in the change being considered & who stand to gain or lose from the change

## → Week #2

- ↳ Meaning of requirements is:

$$\mathcal{E}, S \vdash R$$

- Pairwise, they are disjoint



# Let's classify these phenomena, all of which are about CS 5127/6027 (Fall 2024)

Quiz deadline is in EST.

I want to go to sleep before midnight.

Since the quiz is timed, I need to know how much time is left for me.

I want to earn an "A".

I shall study the required reading first.

Multiple-choice question has no identical options.

I can skip all the lectures.

I'd like to do well on the quizzes.

The quiz questions are within the scope of the required reading.

I don't have to log into the course website on Canvas at all.



Environment  
assertions

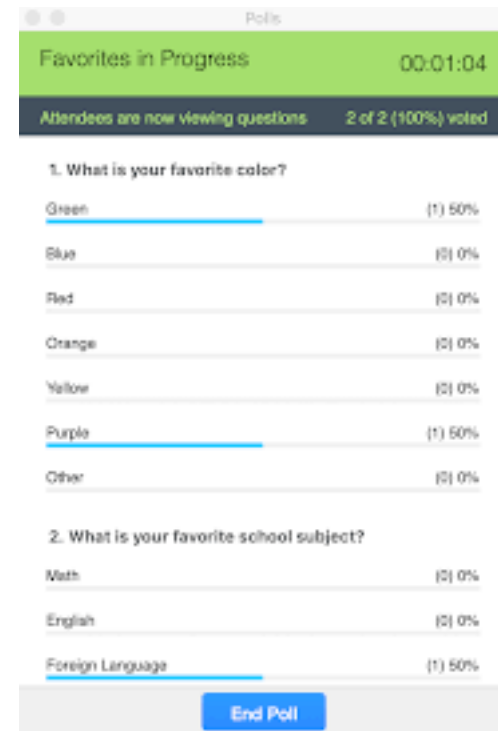
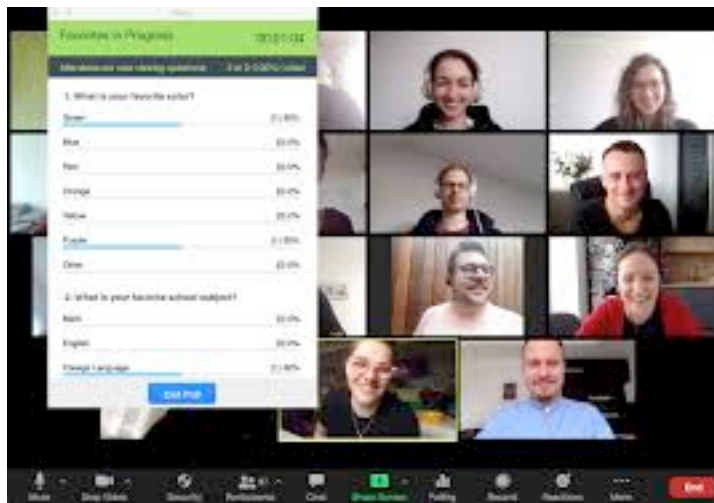


Specifications



Requirements

# Suppose F: "Polling" (an implemented S)



→ You need to provide *at least one* E, F |- R tuple for each given F



## Answer 1

↪ R: "An instructor wants to check if the students understand well the topic just taught"

↪ E:

- "the polling questions are relevant to the topic"
- "the students respond the poll seriously"
- "the students who response the poll are representative of the class"
- ...

↪ Justification:

- (oftentimes, the justification is relatively easy and straightforward, if the right E and R are identified)

## Answer 2

↪ R: "A presenter wants the audience to decide which topic to continue among multiple topic options"

↪ E:

- "the poll choices are consistent with the topic options"
- "only a single choice is permitted"
- "non-participation won't hurt the decision"
- ...

↪ Justification: ...





# Assignment 1: Special Note on "E"

→ We're making the following assumptions as a class (CS 5127/6027, Fall 2024) for Assignment 1, so you do NOT need to document them in your solution, UNLESS you're using them for the justification part of your solution

↳ E: "users (hosts and attendees) have good Internet connections"

↳ E: "users (hosts and attendees) have the latest version of the software"

↳ E: "users (hosts and attendees) have legitimate accounts to login and use the software"

↳ E: "users (hosts and attendees) have all the required, functional hardware, e.g., camera, mic, etc."



## Your conclusion: \_\_\_\_\_

⇒ "48% of the failures observed in a medium-scaled software project were attributed to incorrect or misinterpreted functional specifications or requirements"

V. Basili and B. Perricone, "Software errors and complexity: an empirical investigation", *CACM*'84

⇒ "79.6% of interface faults were due to incomplete or omitted requirements"

D. Perry and C. Stieg, "Software faults in evolving a large, real-time system: a case study", *ESEC*'93

⇒ NASA spacecraft software systems: "the primary cause of safety-related functional faults is errors in recognizing (understanding) the requirements (62% on Voyager; 79% on Galileo)"

R. Lutz, "Analyzing software requirements errors in safety-critical, embedded systems", *RE*'93



Requirements faults  
are common.

## Your conclusion: \_\_\_\_\_

“The hardest single part of building a software system is deciding precisely what to build. No other part of the conceptual work is as difficult as establishing the detailed technical requirements, including the interfaces to people, to machines, and to other software systems. No other part of the work so cripples the resulting system if done wrong. No other part is as difficult to rectify later.”

*Frederick P. Brooks, Jr.*

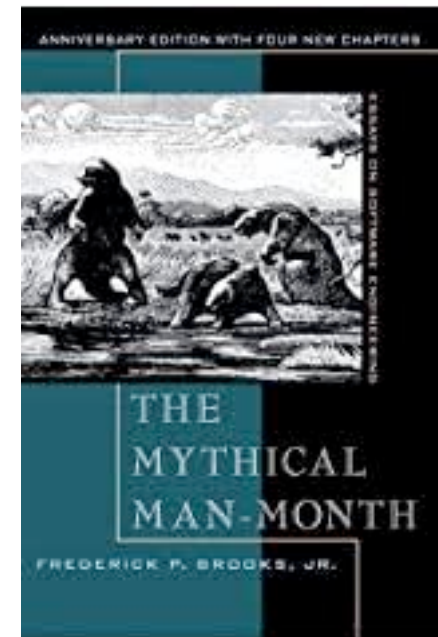
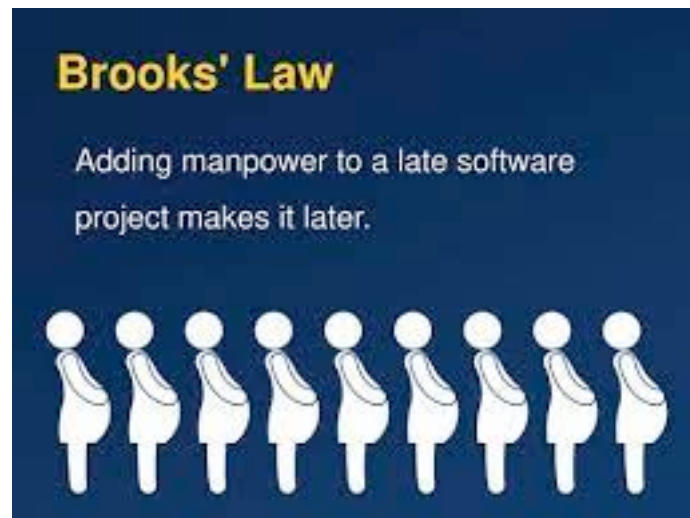
“Clearly, it pays off to invest effort in finding requirements errors early and correcting them, say 1 man-hour rather than waiting to find the error during operations and having to spend 100 man-hours correcting it.”

*Barry Boehm*

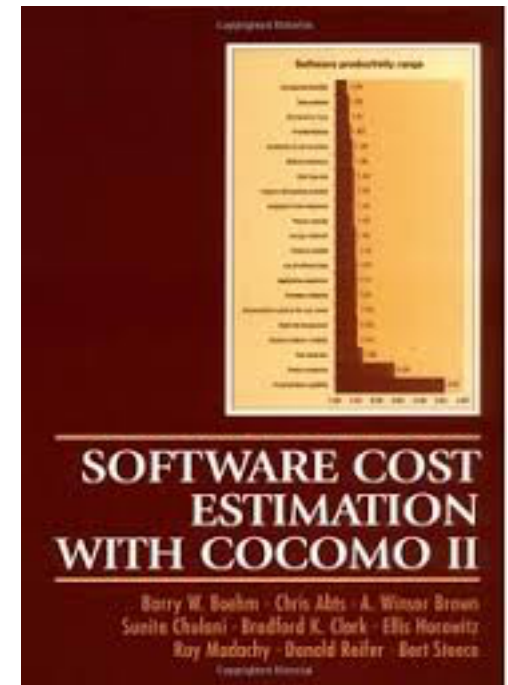
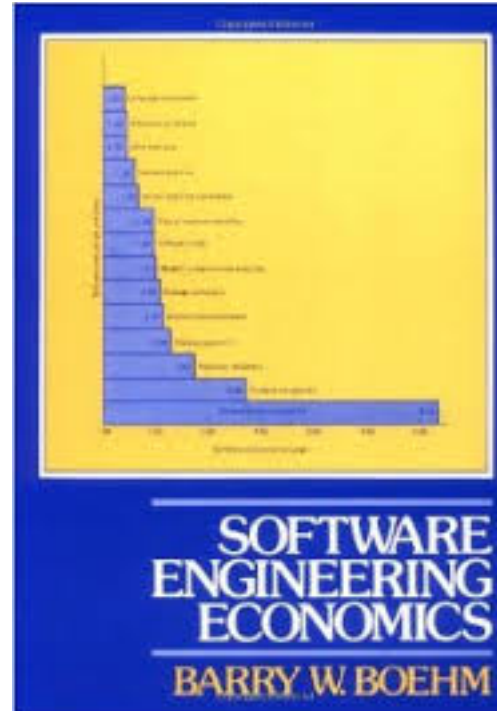


Requirements faults  
are costly.

## Side Note



# Barry Boehm

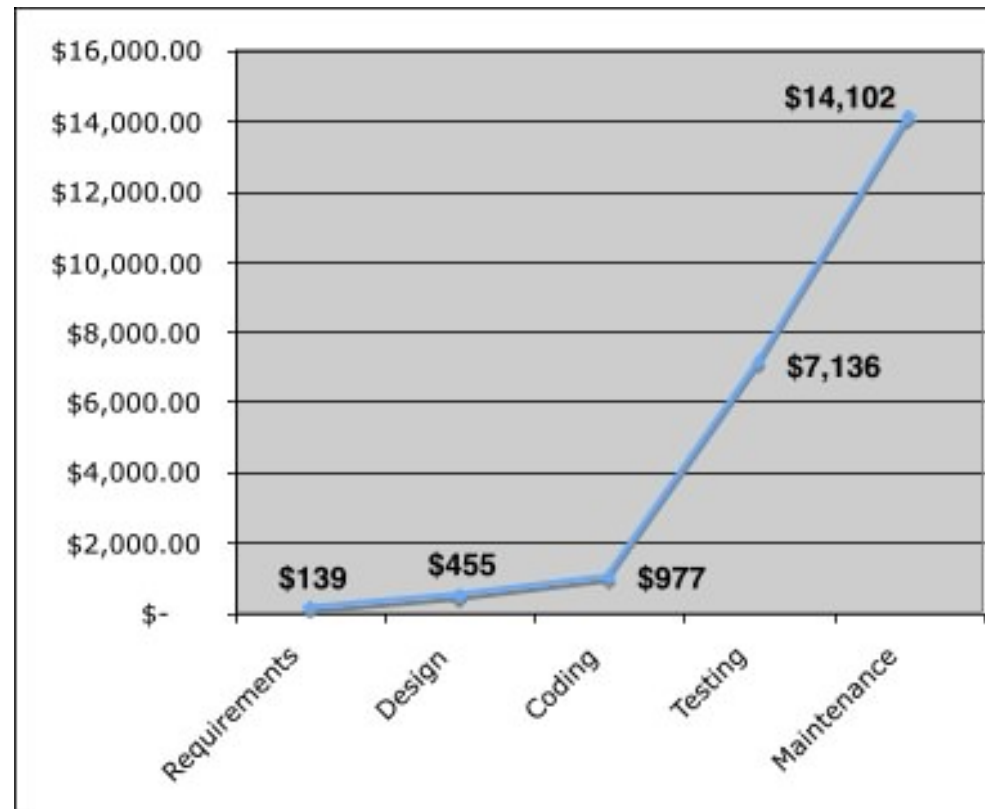


Father of:

- COCOMO (**C**onstructive **C**ost **M**odel)
- spiral (it's about **risk**-mitigation)
- win-win negotiation model
- ...

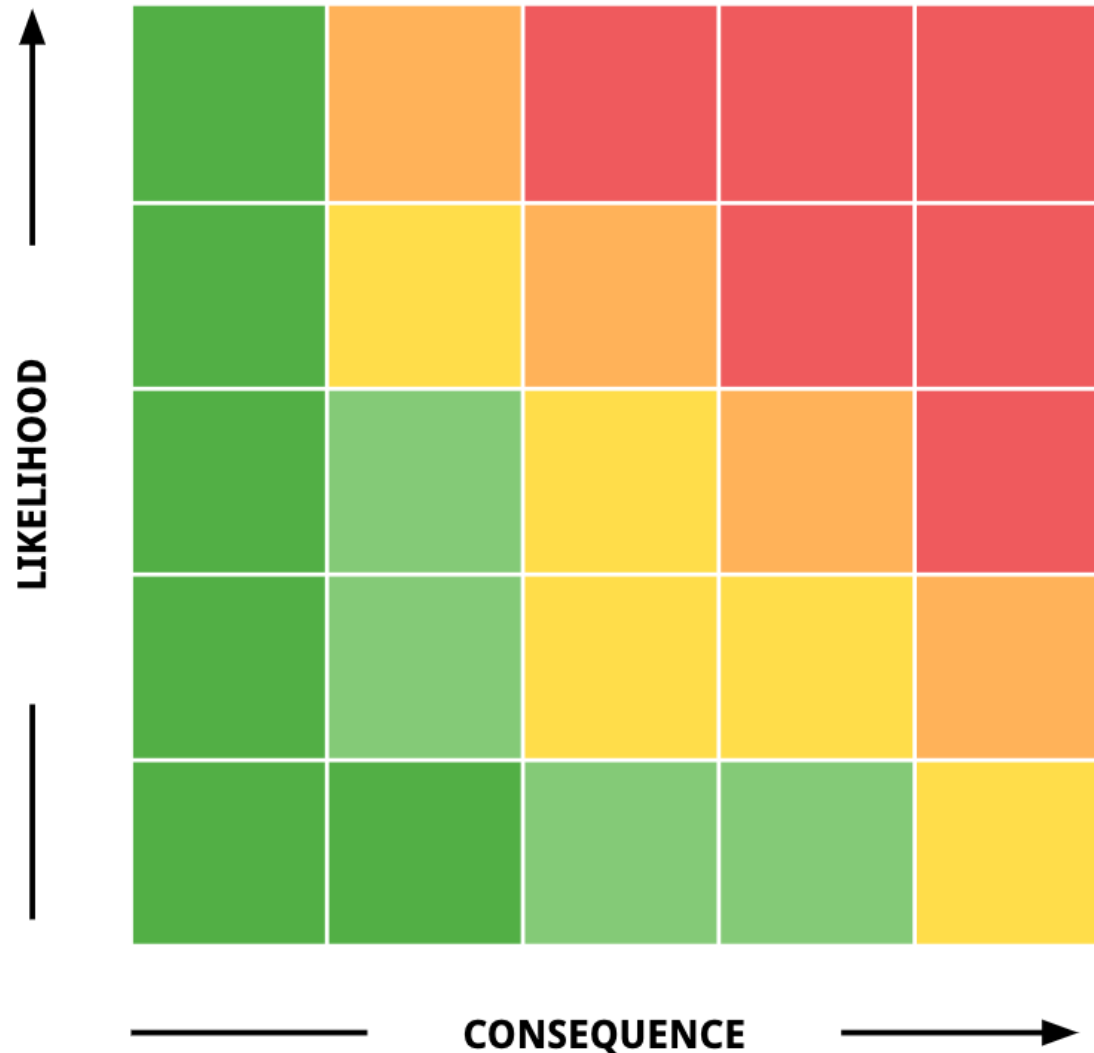
## Boehm's Law

The earlier a problem is discovered in the software development cycle, the lesser the cost to solve it.





# NASA's Risk Matrix





# What's the problem?

A baseball team's GM & scouts



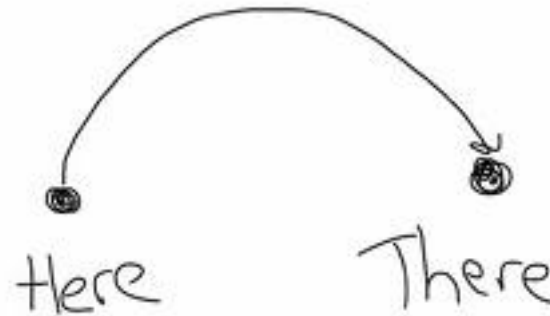
## 2002 MLB Team Salaries

**\$125 million**  
*(highest)*



**\$44 million**  
*(3rd lowest)*

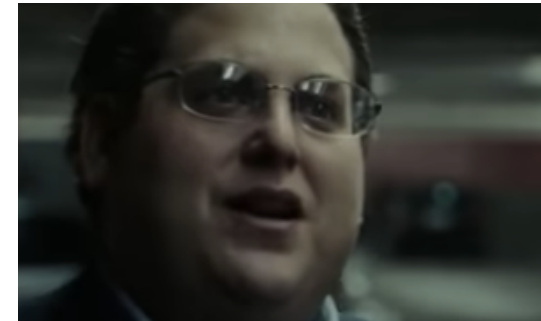
# Problem matters



A **problem** is a difference

between things as desired  
and things as perceived

Scouts: replacing expensive  
players with less expensive  
ones





## Problem matters *a lot*

### → Scouts

↳ Replacing expensive players with less expensive ones

### → Newly recognized needs & desires

↳ Goal should **NOT** be to buy players

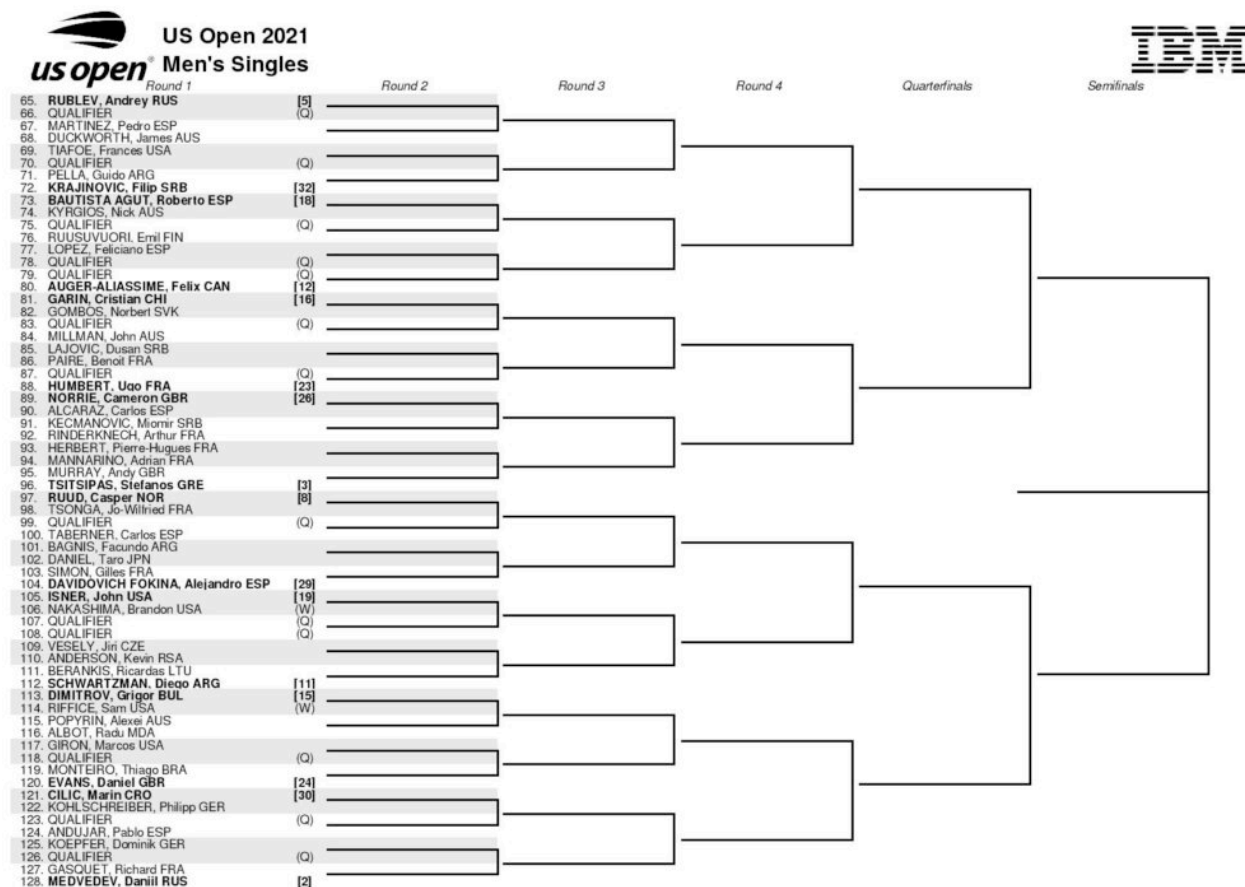
↳ Goal should be to buy wins

➤ In order to buy wins, you need to buy runs → replacing (expensive) players with runs and hopefully wins

→ ... and yes, the problem (**READ: requirements**) changes everything

# Let's try another one

→ How many total games will be played?





## Recap so far

→ Requirements are important *because*

↳ It's hard (*hardest*) to get them right.

↳ It's common to get them wrong.

↳ Getting them wrong is costly.

➤ Some (\$) numbers on the next slide

↳ Hence, *"doing requirements right saves money"*.

# Importance of RE (some numbers)

## → Problems

### ↳ Increased reliance on software

- E.g. cars, dishwashers, cell phones, web services, ...
- Philips estimates that the amount of software in consumer products is doubling every two years

### ↳ Software now the biggest cost element for mission critical systems

- E.g. Boeing 777: approximately 50% of entire development cost for the plane was spent on the avionics software

### ↳ Wastage on failed projects

- E.g. 1997 GAO (US Government Accountability Office) report: \$145 billion over 6 years on software that was never delivered

### ↳ High consequences of failure

- E.g. Ariane 5: \$500 million payload
- E.g. Intel Pentium bug: \$475 million

## → Key factors:

### ↳ Certification costs

- E.g. Boeing 777: >40% of software budget spent on testing to FAA standards compliance

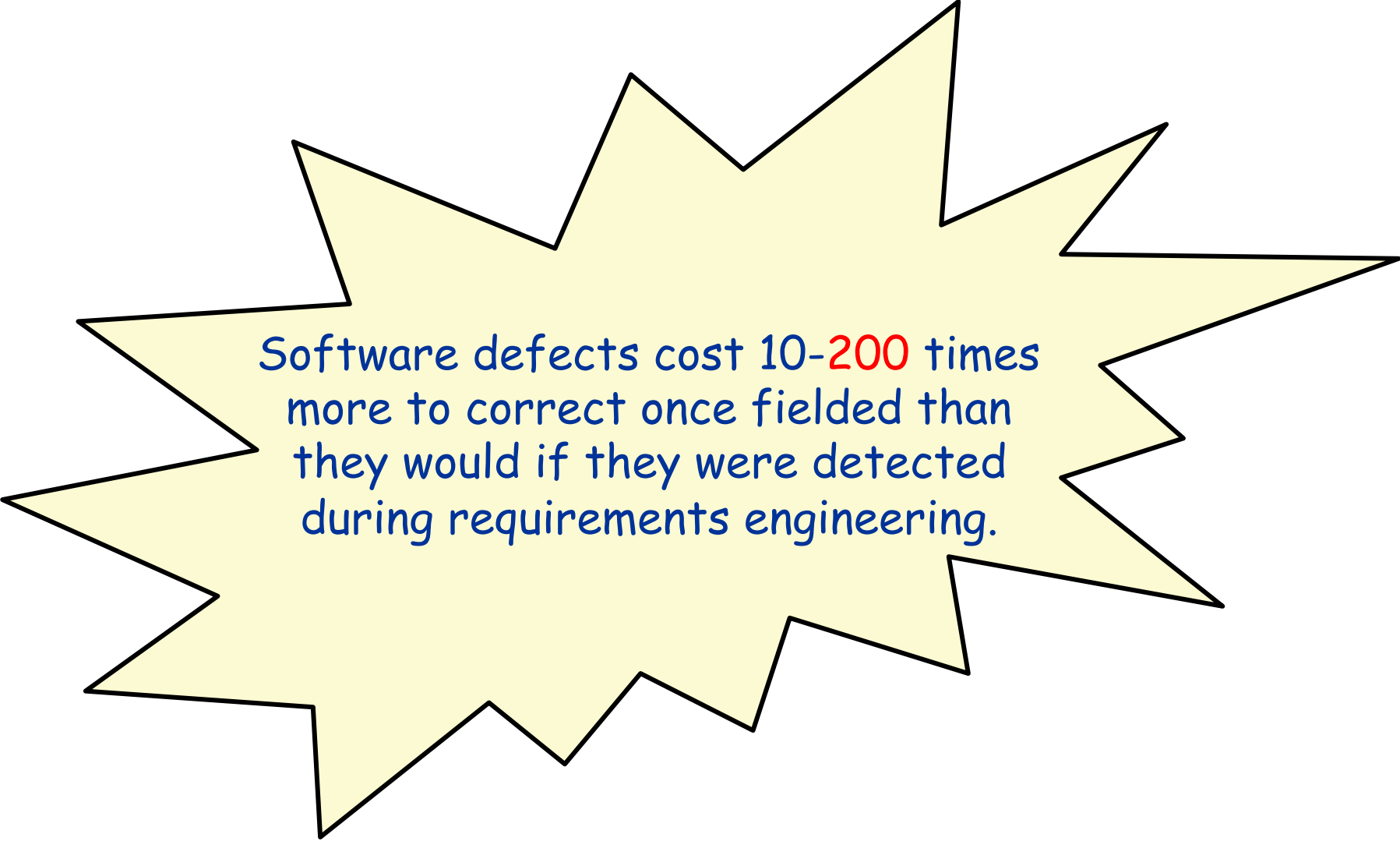
### ↳ Re-work from defect removal

- E.g. Motorola: 60-80% of software budget (was) spent on re-work

### ↳ Changing Requirements

- E.g. California DMV (Dept of Motor Vehicles) system was cancelled after 6 years, after spending \$44 million
- E.g. London ambulance dispatch system; Denver airport luggage management system

## Importance of RE (most memorable #)



Software defects cost 10-200 times more to correct once fielded than they would if they were detected during requirements engineering.





## Today's Take-Aways

→ Req.s are important because

↳ doing RE right saves money

→ To-do

↳ Review today's slides

↳ Continue working on ASN1 (due: Wed., 9/18)

↳ Read the "Experts Elicit Req.s" paper

↳ Complete Quiz3 before 11:59pm, Wed. (9/11)

↳ Attend the lecture on "Elicitation techniques" on Friday (9/13)