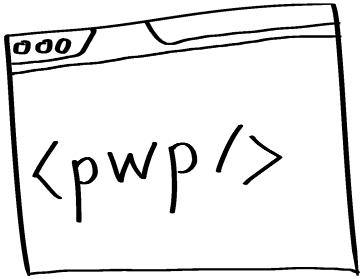
Practical Course: Web Development

Requirements Engineering

Winter Semester 2016/17

Juliane Franze

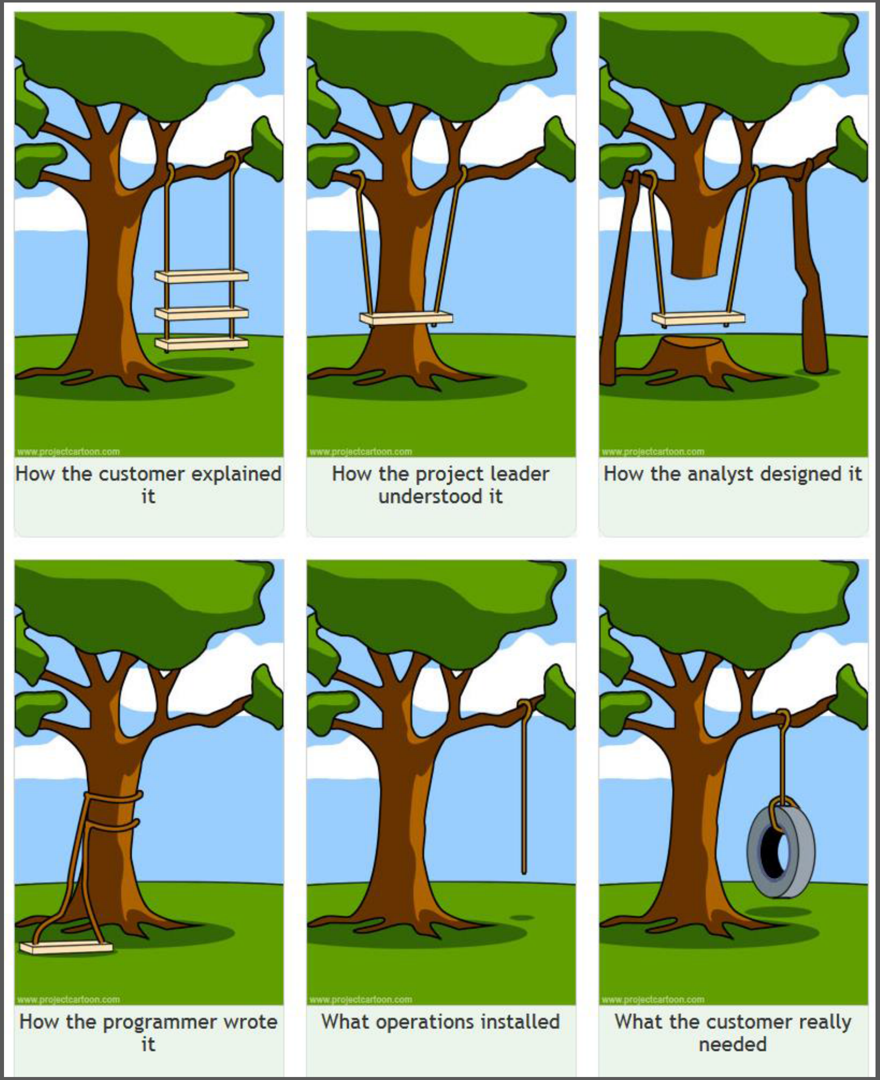


Today’s Agenda

- What is RE?
  - Importance of Requirements
  - Categories of Requirements
  - Writing Requirements
- Homework!

Requirements Engineering (RE)

Why do I need requirements?


How the customer explained it
How the project leader understood it
How the analyst designed it
How the programmer wrote it
What operations installed
What the customer really needed
www.projectcartoon.com

http://www.projectcartoon.com/

Why do I need requirements?

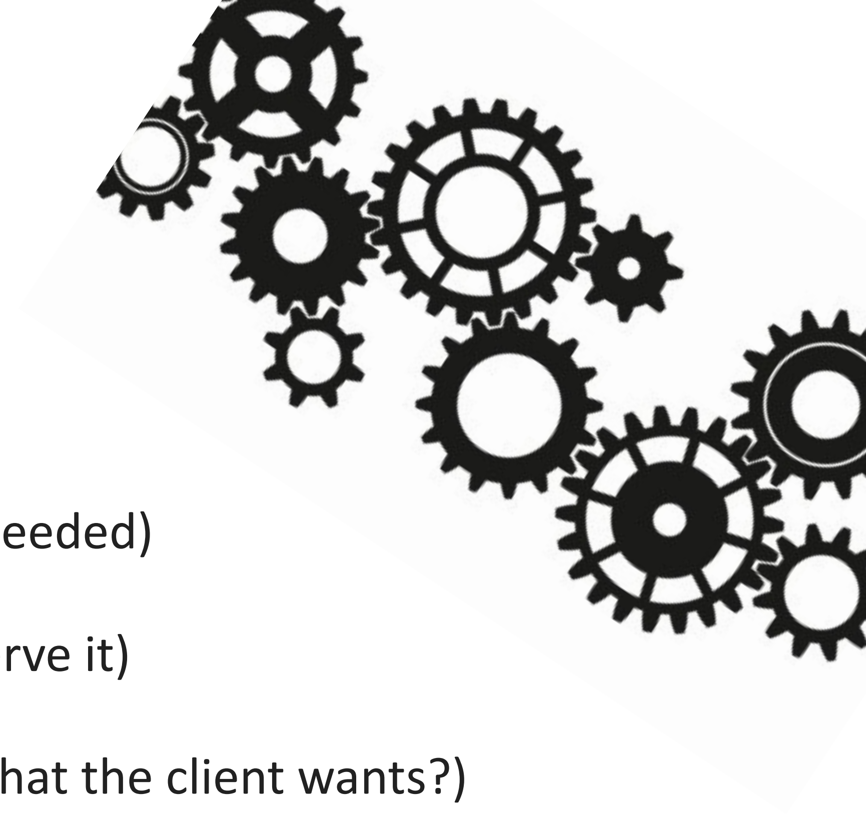
- Common understanding
  - Between clients, project managers, developers and other stakeholders
  - For the needs of the client

- Improve confidence
  - Client knows what to expect = confidence in the delivered product

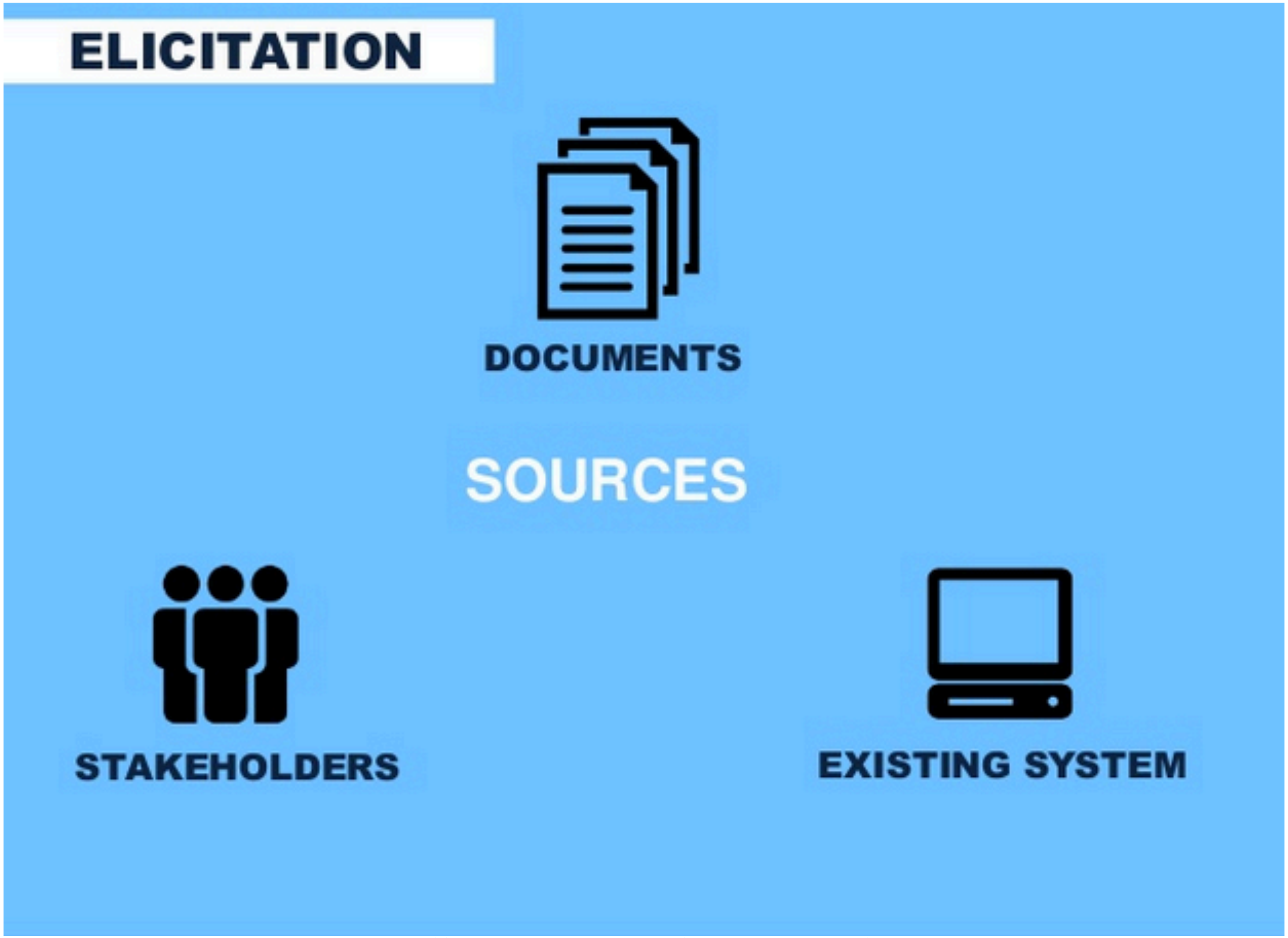
- Specify
  - The features of the system & the constraints
  - A roadmap to development
  - A time table (with milestones and important dates / deadlines)

- Negotiate
  - Reasonable solutions

Requirement Activities



- **Elicitation** (what is wanted and needed)
- **Documentation** (how to preserve it)
- **Validation** (what we defined is what the client wants?)
- **Management** (deal with changes)



ELICITATION

DOCUMENTS

SOURCES

STAKEHOLDERS

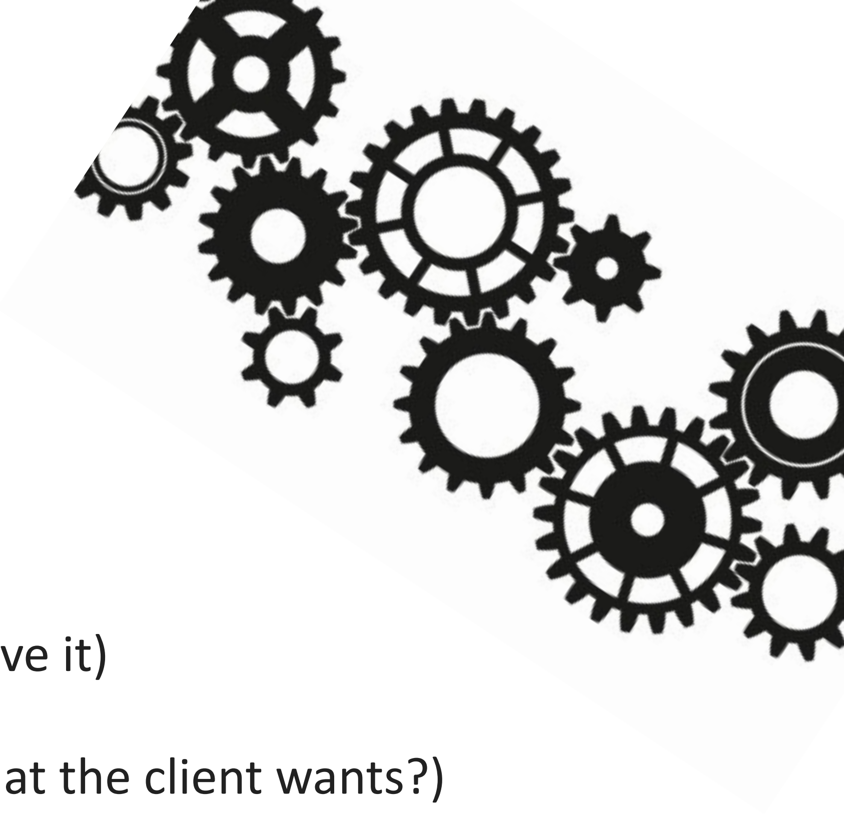
EXISTING SYSTEM

http://www.slideshare.net/ankitabhishek9/requirements-engineering-overview

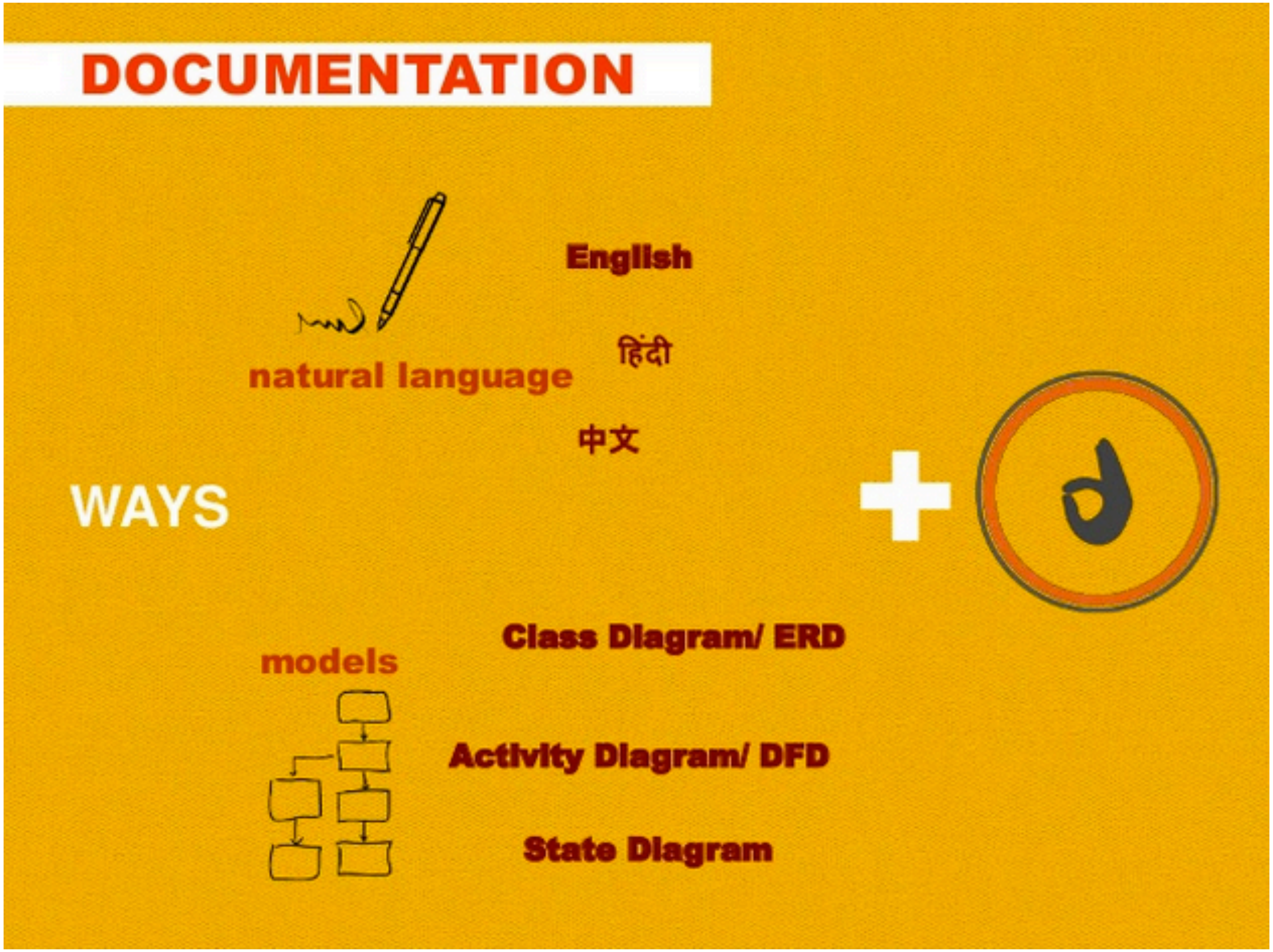
Questions to ask

- … the customer, client or end-user
  - What is the problem?
  - What is the need to solve it?
  - What could be the solution to the problem?
  - What sort of complexities might arise while solving the task?
  - What kind of input or output will be for the system?

Requirement Activities



- **Elicitation** ✓
- **Documentation** (how to preserve it)
- **Validation** (what we defined is what the client wants?)
- **Management** (deal with changes)



DOCUMENTATION

WAYS

natural language

English

हिंदी

中文

models

Class Diagram/ ERD

Activity Diagram/ DFD

State Diagram

Kinds of Requirements

User

- High-level abstraction
- Written in natural language
- Use case diagrams

System

- Structured
- Detailed description of the system
- Exact definition what is implemented

.. May be part of contract between client and developer

Things to write down

- Should include
  - System requirements
  - What will be delivered to the client
  - When it will be delivered to the client
  - → What the system should do BUT NOT in which way particularly

Classes of Requirements

Functional

= WHAT system should do

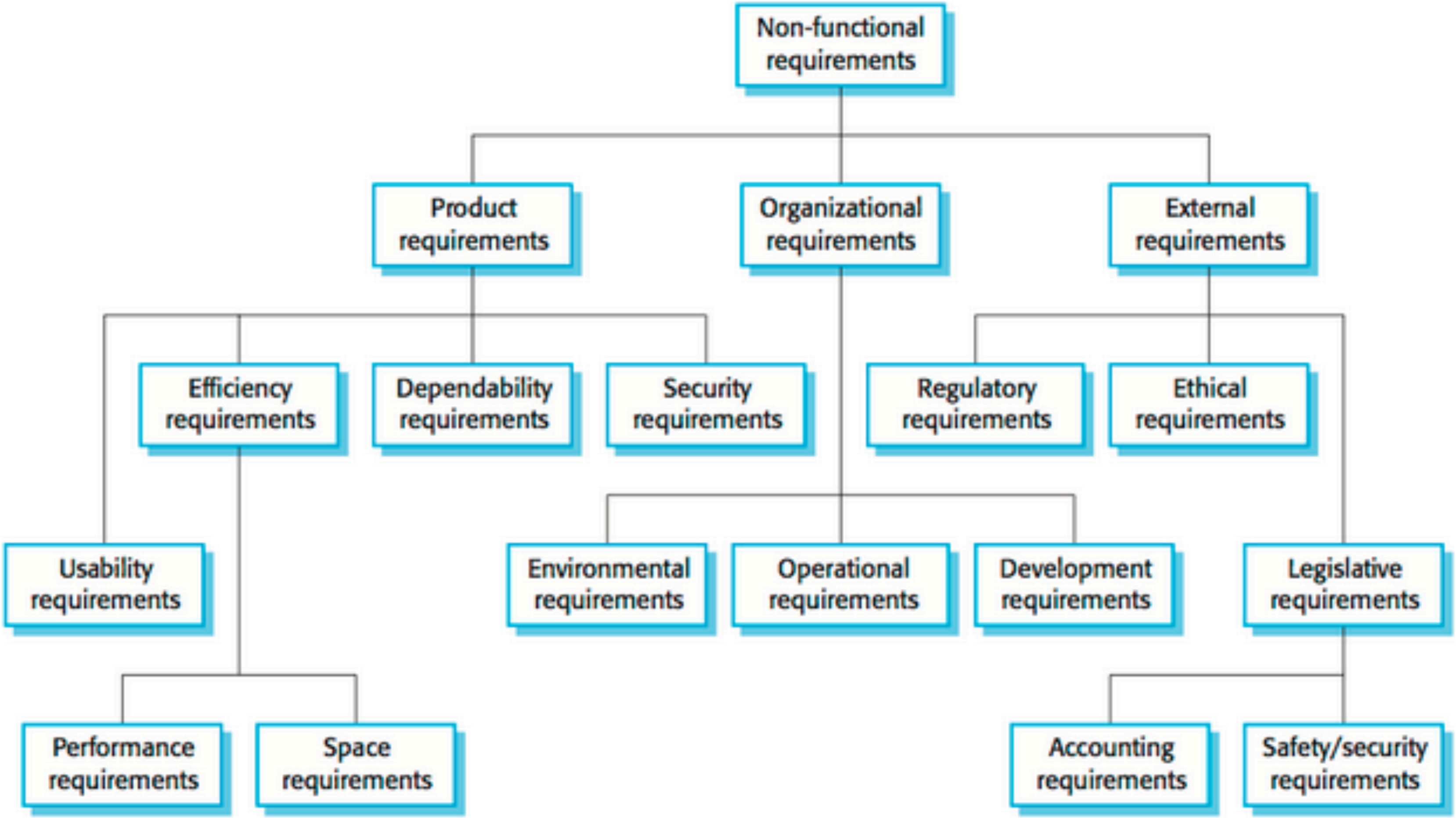
- Individual features
  - Precise
  - Complete
  - Consistent
  - Not open to interpretation
  - Not contradictory

Non-functional

= HOW system should behave

- Constraints on functionality of the entire system
  - Performance
  - Security
  - Reliability

Overview: Non-Functional R.



http://www.cs.ccsu.edu/~stan/classes/CS530/notes14/04-Requirements.html

Requirement Activities

- **Elicitation** ✓
- **Documentation** ✓
- **Validation** (what we defined is what the client wants?)
- **Management** (deal with changes)

Verify vs. Validate

Verification

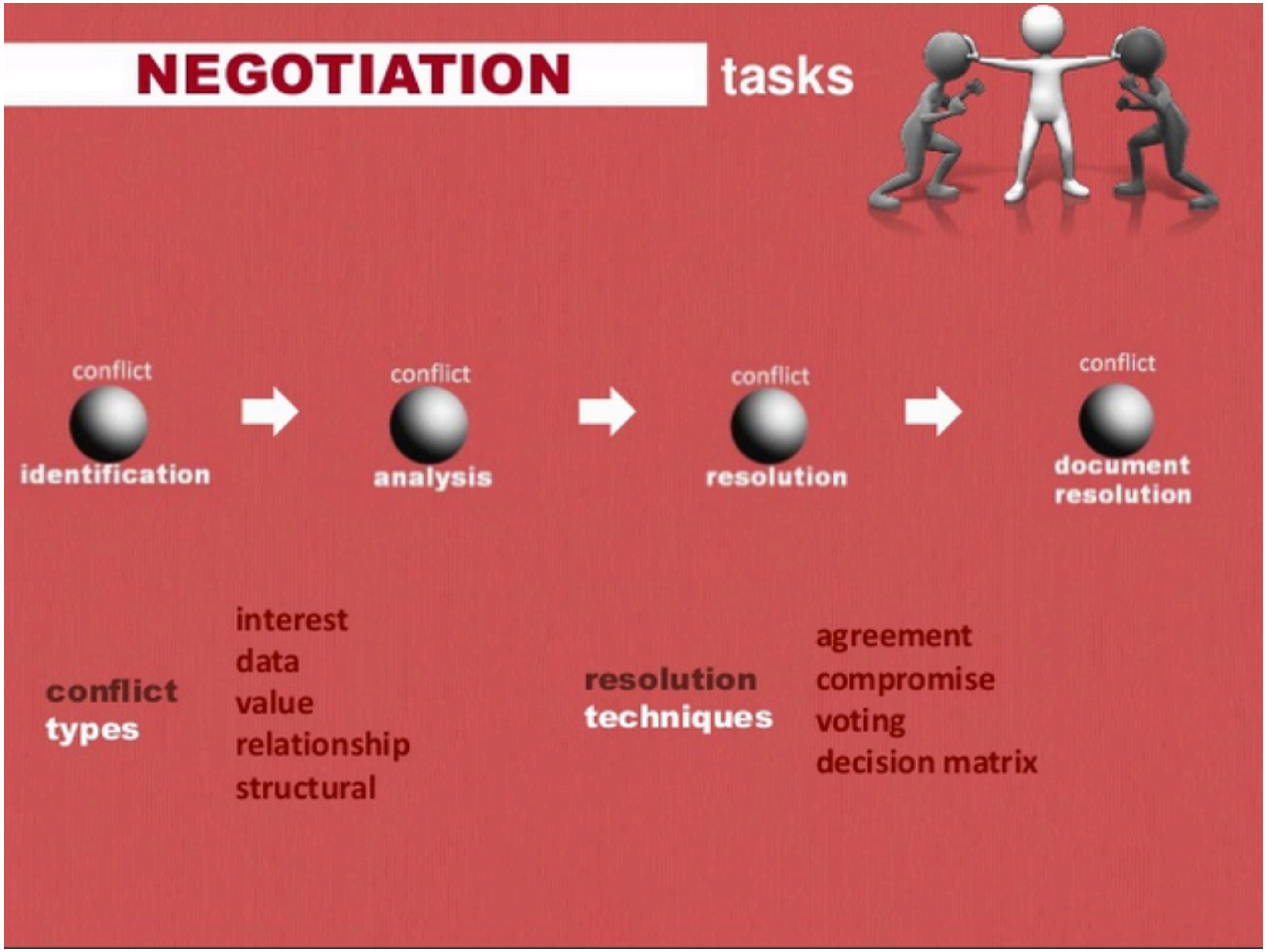
Am I building the product right?

- Internal discussion
- Error detection / Error correction

Validation

Am I building the right product?

- Discussion with client



NEGOTIATION tasks

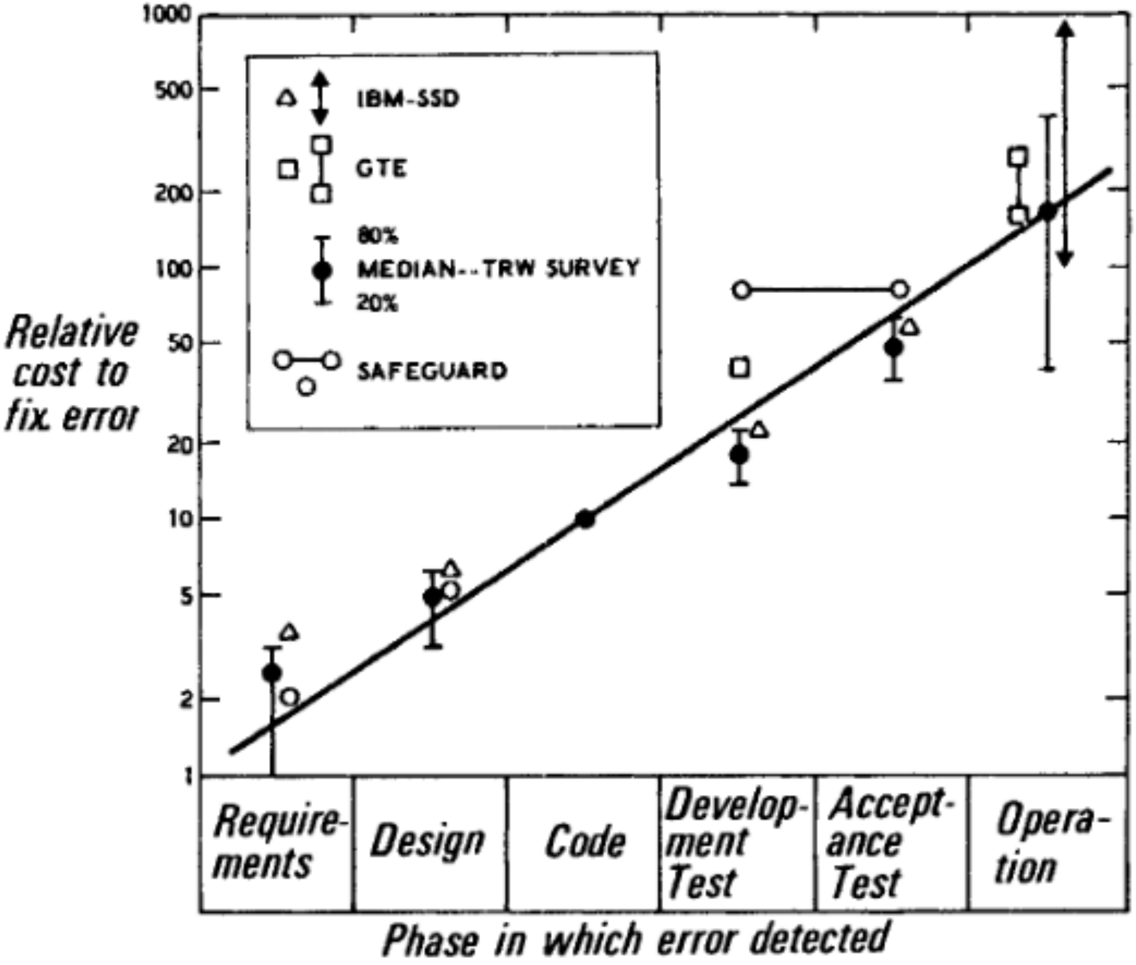
conflict identification → conflict analysis → conflict resolution → conflict document resolution

conflict types: interest, data, value, relationship, structural

resolution techniques: agreement, compromise, voting, decision matrix

http://www.slideshare.net/ankitabhishek9/requirements-engineering-overview

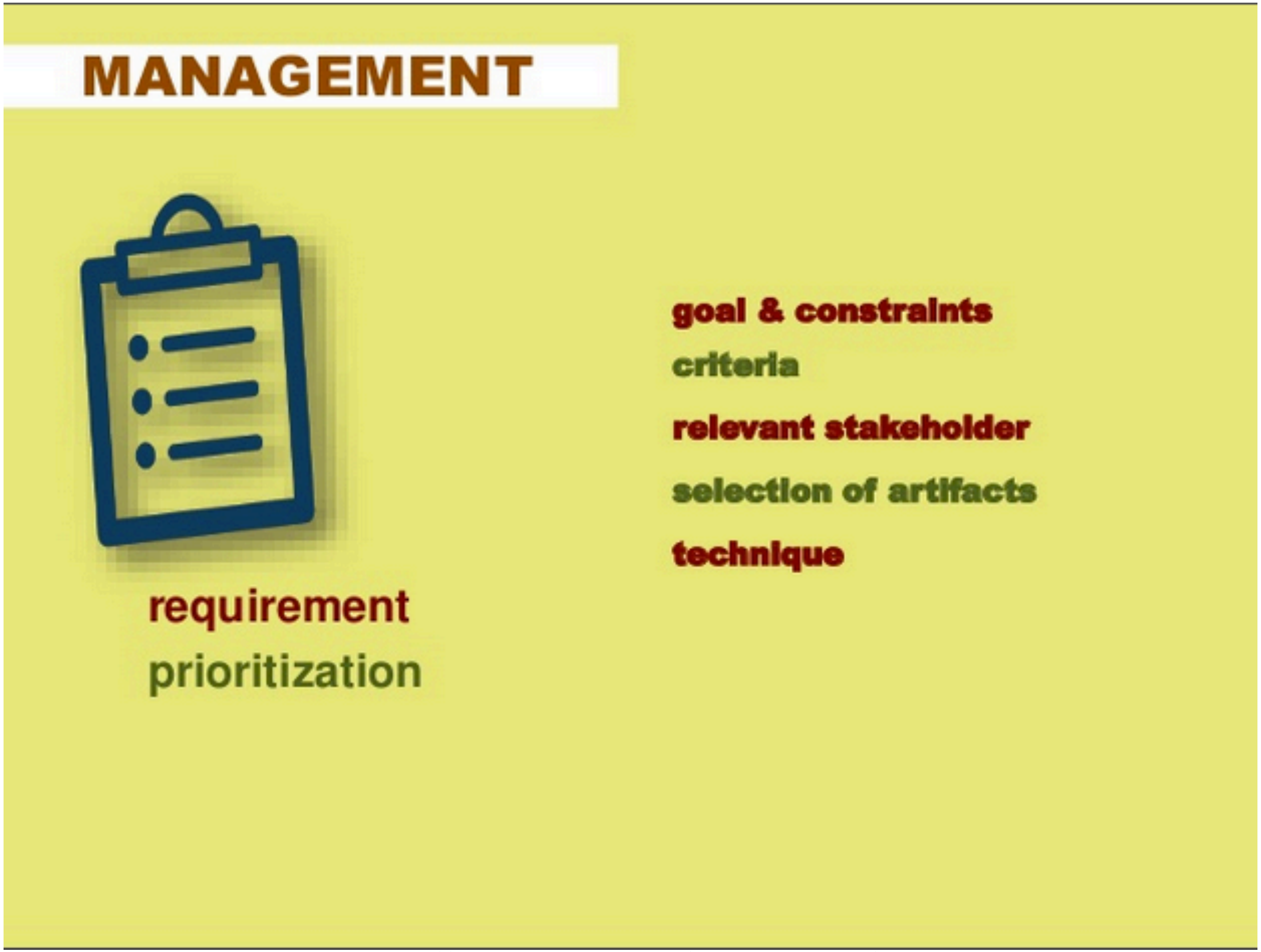
REQUIREMENTS ERRORS MUST BE CAUGHT EARLY



http://csse.usc.edu/TECHRPTS/1979/usccse79-501/usccse79-501.pdf

Requirement Activities

- **Elicitation** ✓
- **Documentation** ✓
- **Validation** ✓
- **Management** (deal with changes)



MANAGEMENT

requirement prioritization

- goal & constraints
- criteria
- relevant stakeholder
- selection of artifacts
- technique

http://www.slideshare.net/ankitabhishek9/requirements-engineering-overview

Requirement Activities

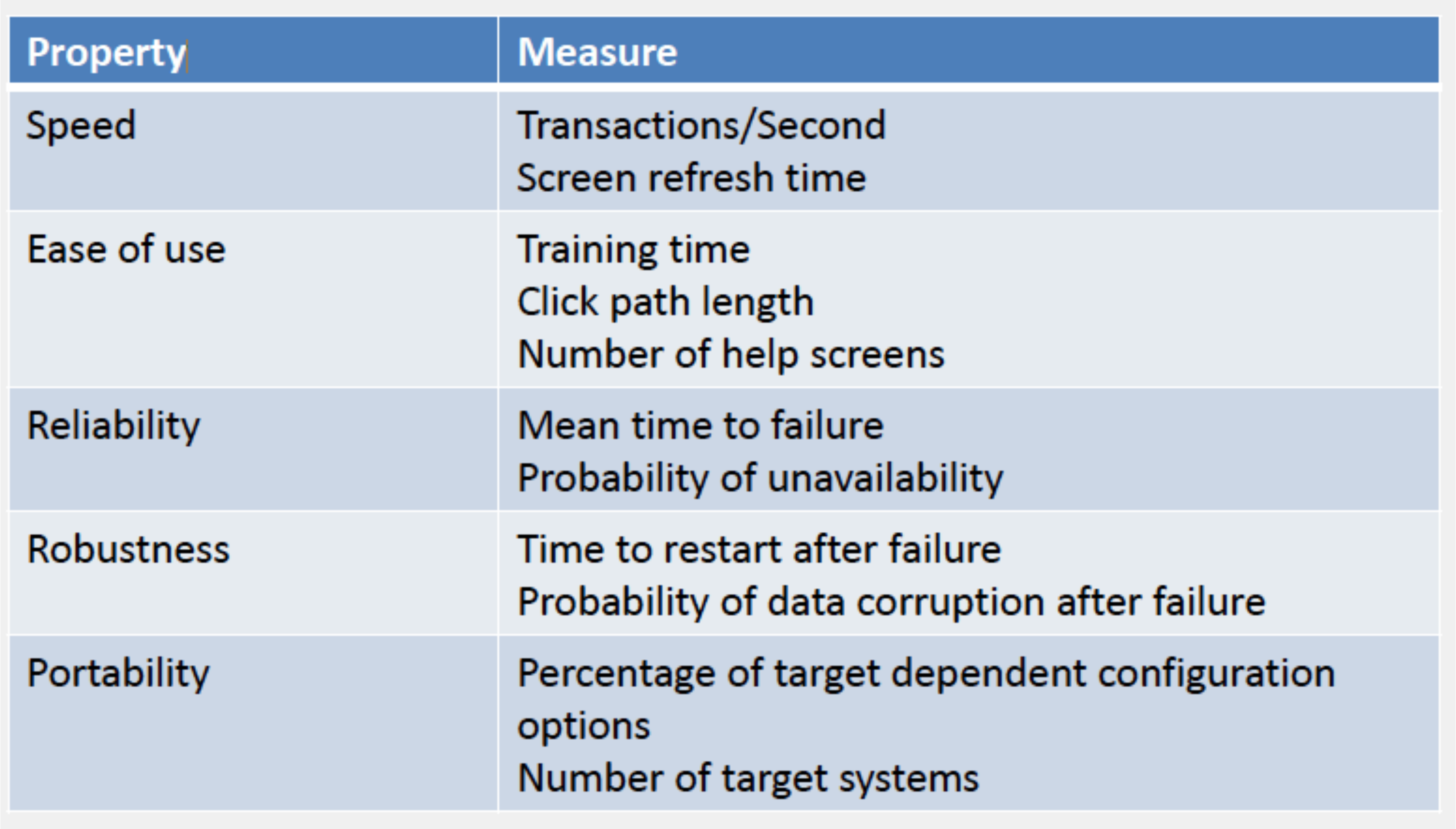
- Elicitation ✓
- Documentation ✓
- Validation ✓
- Management ✓

How to write good and reliable Requirement Documents?

Functional Requirements

- Precise
- Complete
- Consistent
- Not open to interpretation
- Not contradictory

Metrics for Non-functional requirements



Property	Measure
Speed	Transactions/Second Screen refresh time
Ease of use	Training time Click path length Number of help screens
Reliability	Mean time to failure Probability of unavailability
Robustness	Time to restart after failure Probability of data corruption after failure
Portability	Percentage of target dependent configuration options Number of target systems

Alright → Better Non-Functional R.

- „The application should be user friendly and easy to use.“
  - Better:
    Students should be able to use the application after going through the 1min tutorial. Questions about usage should not exceed 1 per day.

- „The system should respond fast.“
  - Better:
    The system should be able to process up to 1000 requests per second in peak load.

Task & Homework

- Write Requirements for your project!

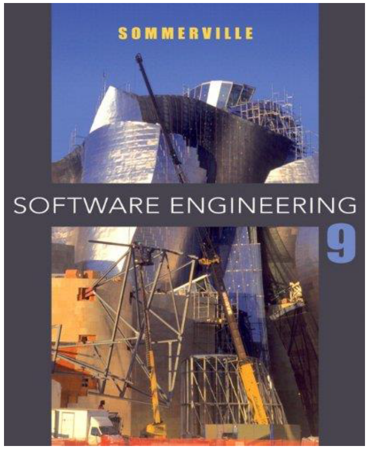
- Distinguish between Functional R. and Non-functional R.

- Try to think about:
  - Precision
  - Completeness
  - Robustness
  - Measurability

- Document and Validate!

Good Stuff

- Ian Sommerville: Software Engineering, 9th revised Edition. Pearson Education, 2011.



References

- [1] "Systems and software engineering - Vocabulary," ISO/IEC/IEEE std 24765:2010(E), 2010.
- [2] Sommerville, Software Engineering, 9 ed., Chapter 4. http://www.cs.ccsu.edu/~stan/classes/CS530/notes14/04-Requirements.html