MICHIGAN ENGINEERING
UNIVERSITY OF MICHIGAN

Requirements and Specifications

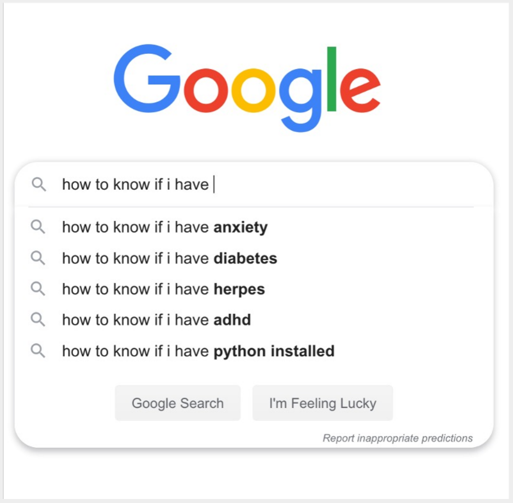
I'LL NEED TO KNOW YOUR REQUIREMENTS BEFORE I START TO DESIGN THE SOFTWARE.
FIRST OF ALL, WHAT ARE YOU TRYING TO ACCOMPLISH?
I'M TRYING TO MAKE YOU DESIGN MY SOFTWARE.
I MEAN WHAT ARE YOU TRYING TO ACCOMPLISH WITH THE SOFTWARE?
I WON'T KNOW WHAT I CAN ACCOMPLISH UNTIL YOU TELL ME WHAT THE SOFTWARE CAN DO.
TRY TO GET THIS CONCEPT THROUGH YOUR THICK SKULL: THE SOFTWARE CAN DO WHATEVER I DESIGN IT TO DO!
CAN YOU DESIGN IT TO TELL YOU MY REQUIREMENTS?
E-mail: SCOTTADAMS@AOL.COM
© 2006 Scott Adams, Inc./Dist. by UFS, Inc.
www.dilbert.com

EECS 481 (W24)

MICHIGAN ENGINEERING
UNIVERSITY OF MICHIGAN

The Story so far…

- **Quality assurance** is critical to software engineering
- Okay, so we want to build a quality product.
- What are we supposed to be building again?
- Design of a system is a process of having a realization from a specification or a requirement.
- Implementation is a type of realization.


how to know if i have
how to know if i have **anxiety**
how to know if i have **diabetes**
how to know if i have **herpes**
how to know if i have **adhd**
how to know if i have **python installed**

One-Slide Summary

- **Requirements** articulate the relationship and interface between a desired system and its environment. This includes both what is (or is expected) and what should be.
- We distinguish between **functional** and **quality** (or non-functional) requirements. Both should be stated in measurable ways.
- Requirements can describe variables, inputs, and outputs, and assumptions between them.
- We distinguish between informal statements and formal (verifiable) requirements.

Outline (the emotional journey)

- Background
  - Basic concepts and definitions
  - Motivating case study
  - Stories and examples
- Functional and Quality Requirements
- Requirements Engineering

MICHIGAN ENGINEERING
UNIVERSITY OF MICHIGAN

Learning Objectives: by the end of today's lecture you should be able to…

1. (*knowledge*) explain the difference between system and software requirements.

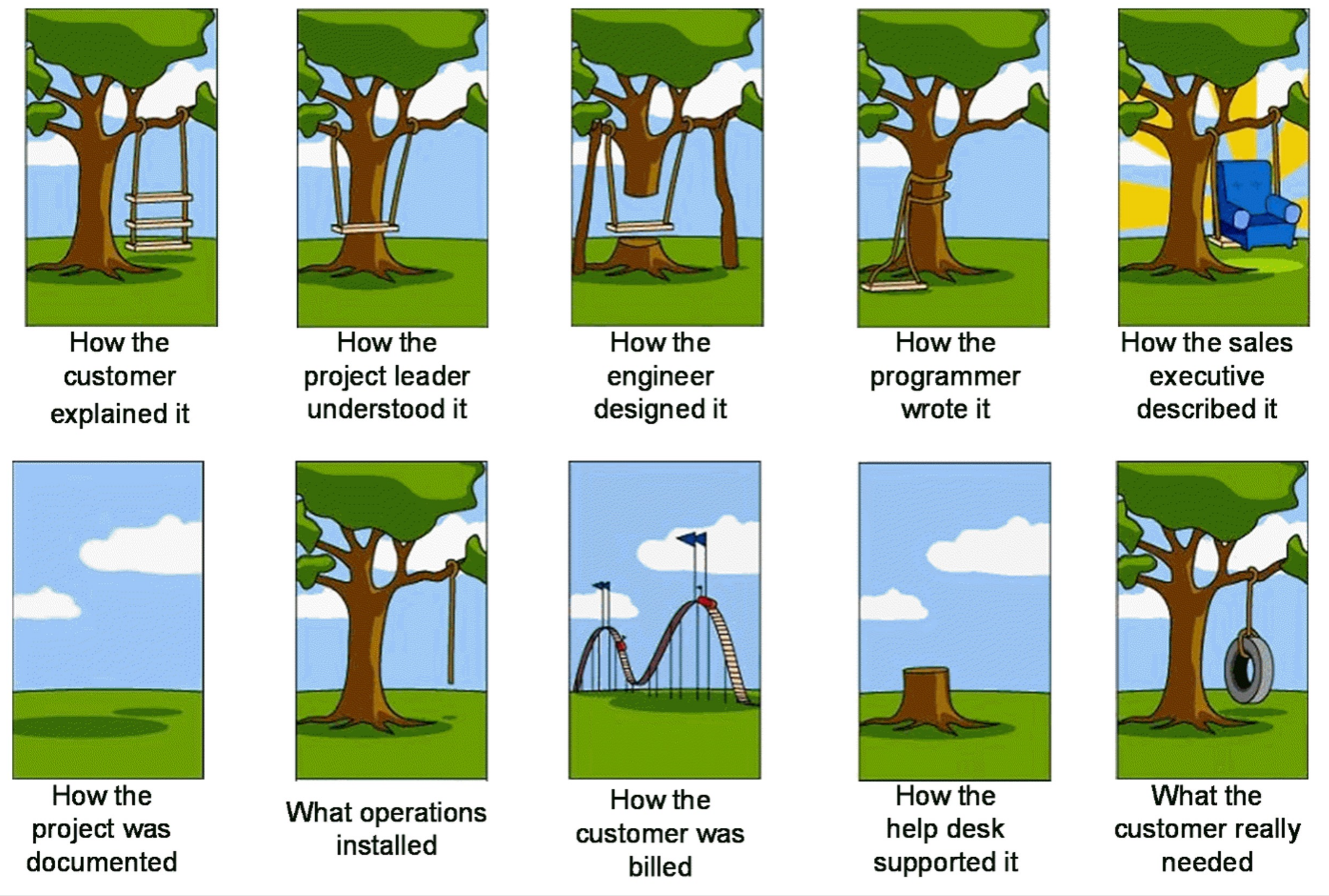
2. (*knowledge*) identify or give examples of functional and quality requirements.

3. (*knowledge*) describe the steps of requirements elicitation
   - (*stretch skill*) feel comfortable in a mock interview for requirements elicitation.

4. (*value*) sincerely believe requirements are important and difficult.

MICHIGAN ENGINEERING
UNIVERSITY OF MICHIGAN

Background


How the customer explained it
How the project leader understood it
How the engineer designed it
How the programmer wrote it
How the sales executive described it
How the project was documented
What operations installed
How the customer was billed
How the help desk supported it
What the customer really needed

Fred Brooks

- **Frederick Phillips Brooks Jr.** (April 19, 1931 – November 17, 2022) was an American computer architect, software engineer, and computer scientist, best known for managing the development of IBM's System/360 family of computers and the OS/360 software support package, then later writing candidly about those experiences in his seminal book *The Mythical Man-Month*.
- In 1976, Brooks was elected to the National Academy of Engineering for "contributions to computer system design and the development of academic programs in computer sciences".
- Brooks received many awards, including the National Medal of Technology in 1985 and the Turing Award in 1999.

Requirements

- **Requirements** say what the system will do, not how it will do it
- "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 … No other part of the work so cripples the resulting system if done wrong. No other part is as difficult to rectify later."
  -- Fred Brooks

Requirement Elicitation

- Requirement elicitation is the process of gathering and defining the requirements for a software system.
- The goal of requirement elicitation is to ensure that the software development process is based on a clear and comprehensive understanding of the customer's needs and expectations.
- Requirement elicitation involves the identification, collection, analysis, and refinement of the requirements for a software system.
- It is a critical part of the software development life cycle and is typically performed at the beginning of the project.

https://en.wikipedia.org/wiki/Requirements_elicitation

Requirement Elicitation (Cont'd)

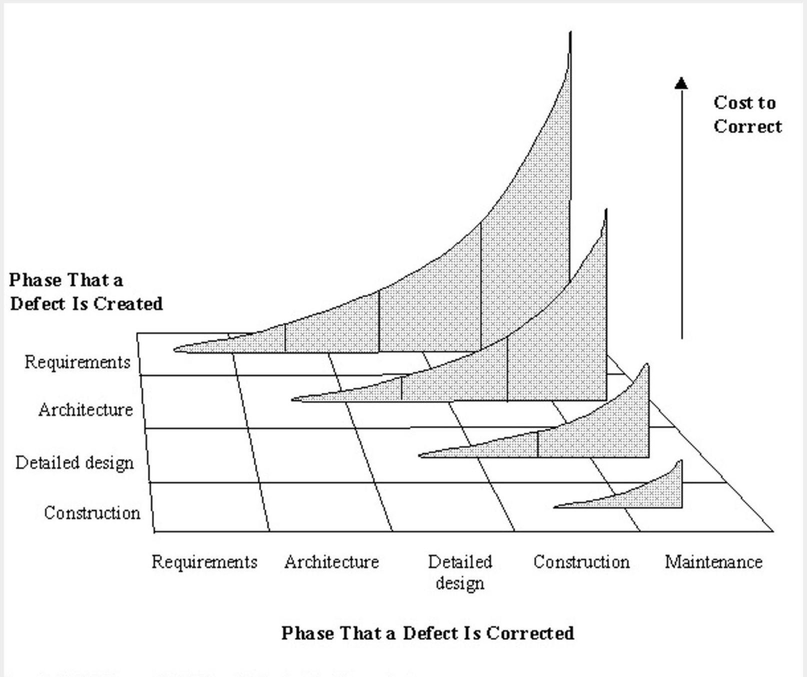
- Requirement elicitation involves stakeholders from different areas of the organization, such as business owners, end-users, and technical experts.
- The output of the requirement elicitation process is a set of clear, concise, and well-defined requirements that serve as the basis for the design and development of the software system.
- Requirement elicitation can be challenging and error-prone, as it may involve problems of scope, understanding, volatility, and communication among the stakeholders.

Requirement Elicitation Methods

Requirement elicitation can be done using various methods and techniques, such as interviews, questionnaires, workshops, brainstorming, use cases, prototyping, and model checking.

Each method has its own advantages and disadvantages, and the choice of the method depends on factors such as the type, size, and complexity of the project, the availability and maturity of the stakeholders, and the resources and time constraints.

“Difficult to Rectify Later”



Copyright 1998 Steven C. McConnell. Reprinted with permission from *Software Project Survival Guide* (Microsoft Press, 1998).

MICHIGAN ENGINEERING
UNIVERSITY OF MICHIGAN

Healthcare.gov

MARCH 10, 2014
Ukraine's Future / Minimum-Wage Wars / Wes Anderson
TIME
CODE RED_
Inside the nightmare launch of HealthCare.gov and the team that figured out how to fix it
BY STEVEN BRILL
time.com

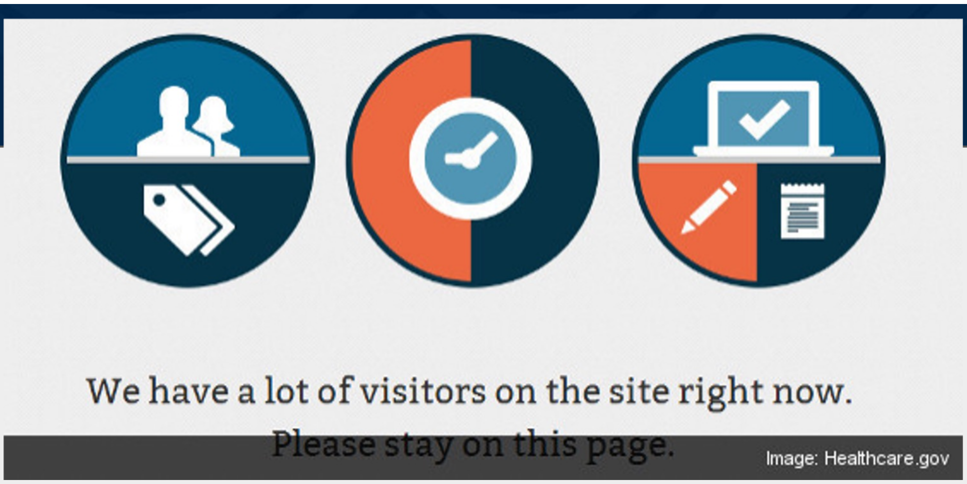
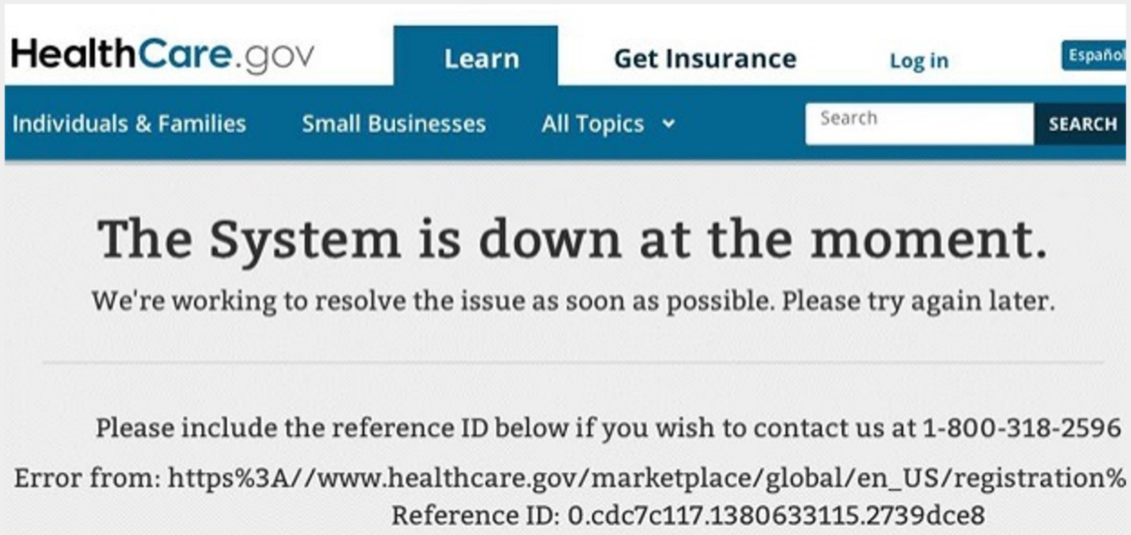

We have a lot of visitors on the site right now.
Please stay on this page.

Image: Healthcare.gov



The System is down at the moment.

We're working to resolve the issue as soon as possible. Please try again later.

Please include the reference ID below if you wish to contact us at 1-800-318-2596

Error from: https%3A//www.healthcare.gov/marketplace/global/en_US/registration%
Reference ID: 0.cdc7c117.1380633115.2739dce8

Healthcare.gov (Cont'd)

- HealthCare.gov was launched on October 1, 2013, as planned, despite the concurrent partial government shutdown.
- However, the launch was marred by serious technological problems, making it difficult for the public to sign up for health insurance.
- Some of the problems included slow loading times, error messages, incorrect information, security breaches, and system crashes.
- Many users were unable to create accounts, verify their identities, or complete their applications.
- The website also had issues with transferring data to insurance companies and state agencies.

Healthcare.gov (Cont’d)

- One of the reasons for the website's failure was that it was too popular.
- The website received over 4.7 million visits on the first day, far exceeding the expectations of the developers and administrators.
- The website was not designed or tested to handle such a high volume of traffic, and the servers were overwhelmed by the demand.
- The website also had to interact with multiple federal and state databases, which added to the complexity and vulnerability of the system.

Healthcare.gov (Cont'd)

- The Obama administration faced intense criticism and scrutiny from the media, the public, and Congress over the website's problems.
- The administration apologized for the inconvenience and frustration caused by the website and promised to fix it as soon as possible.
- The administration also hired additional contractors and experts to work on improving the website's performance and functionality.
- By December 2013, most of the major issues with the website were resolved, and more people were able to enroll in health insurance plans through HealthCare.gov.

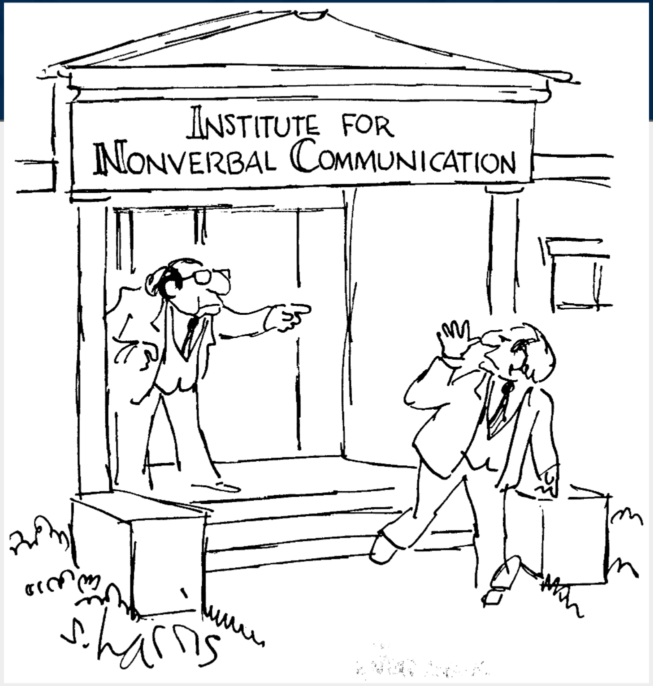
What is Past is Prologue

- A 1994 survey of 8000 projects at 350 companies found: 31% of projects canceled before completed; 9% of projects delivered on time, within budget in large companies, 16% in small companies. (Similar results reported since.)
- Largest causes:
  - Incomplete requirements (13.1%)
  - Lack of user involvement (12.4%)
  - Lack of resources (10.6%)
  - Unrealistic expectations (9.9%)
  - Lack of executive support (9.3%)

No “programmers were too inept” or “we forgot how AVL trees work”

Communication Problem

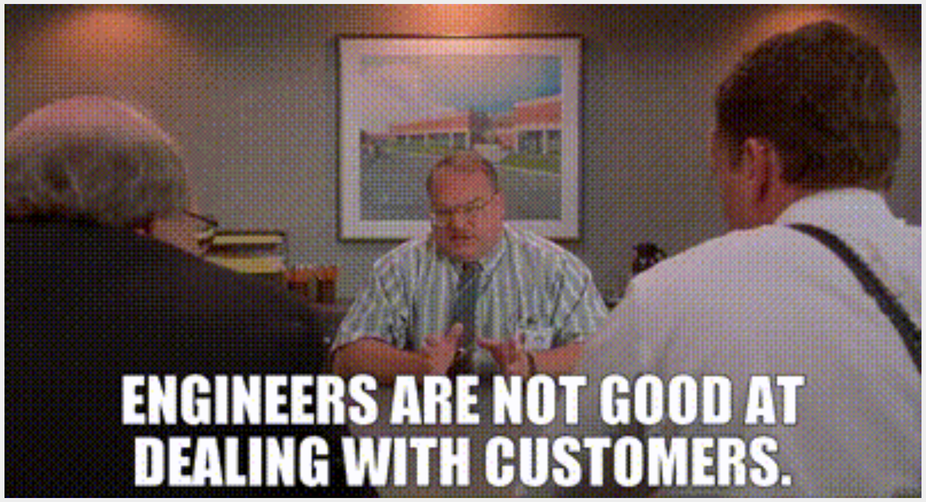
- Goal: figure out what should be built
- Express those ideas that the correct thing is built

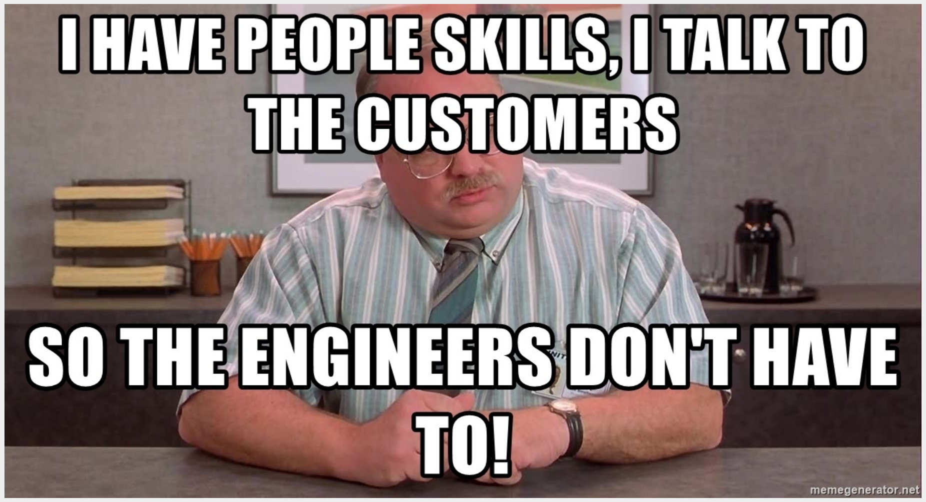


“I’m Already Good At This”: Denial

- Denial by prior knowledge – we have done this before, so we know **what** is required
- Denial by hacking – our fascination with machines dominates our focus on the **how**
- Denial by abstraction – we pursue elegant models which obscure, remove or downplay the real world
- Denial by vagueness – imply (vaguely) that machine descriptions are actually those of the world (cf. “threat to validity”)

https://www.youtube.com/watch?v=hNuu9CpdjIo




I HAVE PEOPLE SKILLS, I TALK TO THE CUSTOMERS
SO THE ENGINEERS DON'T HAVE TO!
memegenerator.net

MICHIGAN ENGINEERING
UNIVERSITY OF MICHIGAN

Requirements Brainstorming Example

- You are paying someone to write software for “selling videos on the web”
- Your thoughts on …
  - How fast should we deliver content, at what quality, for what price?
  - “Nice to have” functionality
  - Required functionality
  - Expected qualities
  - Involved subproblems

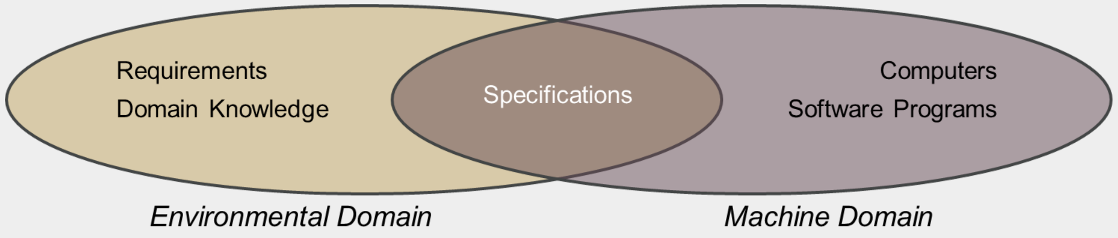
Use clothing to keep warm during winter

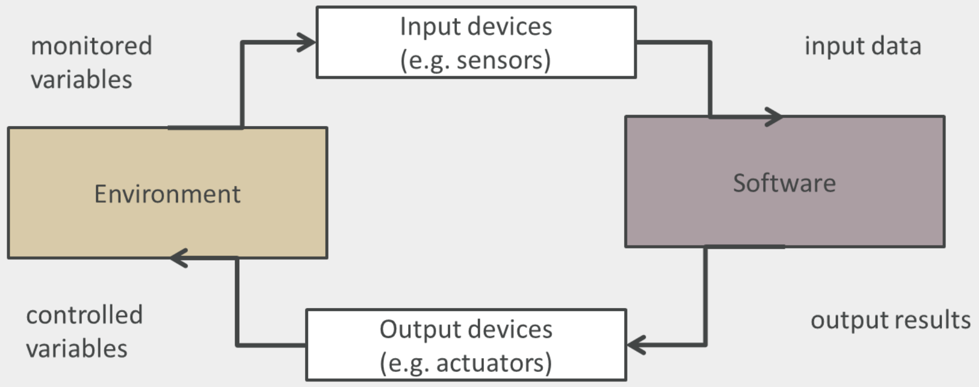
Winter has arrived, but you have decided to cut down on your bills this year by lowering the thermostat. Although your feet will soon feel chilly, there are

PROBABLY BAD NEWS.COM

MICHIGAN ENGINEERING
UNIVERSITY OF MICHIGAN

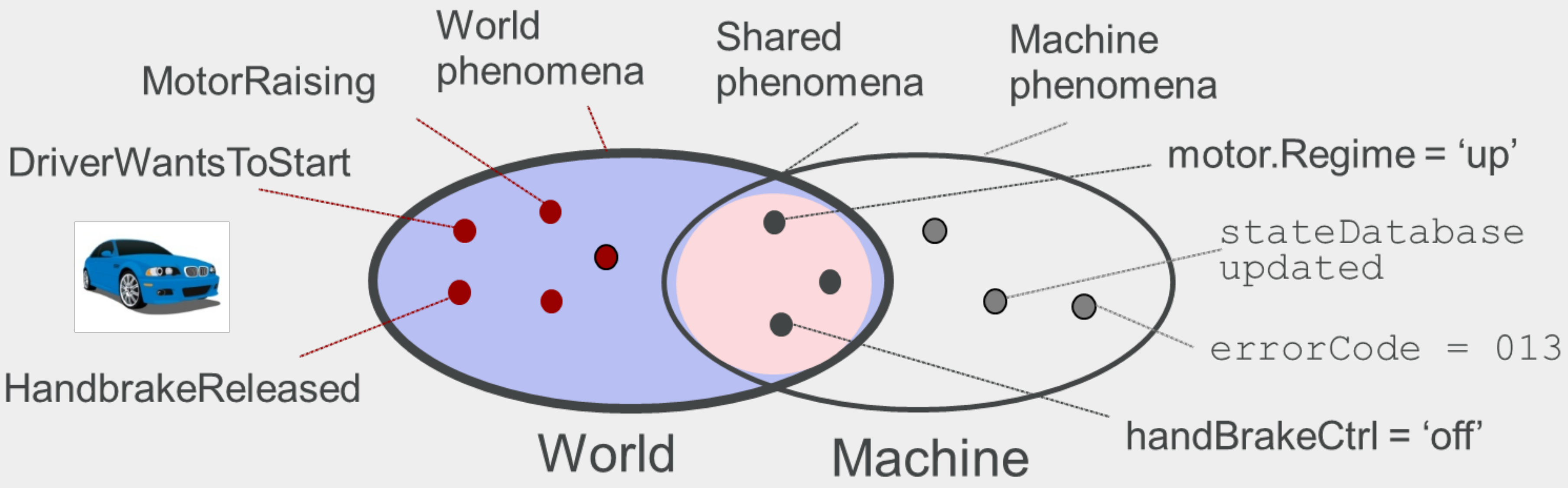
Environment vs. Machine





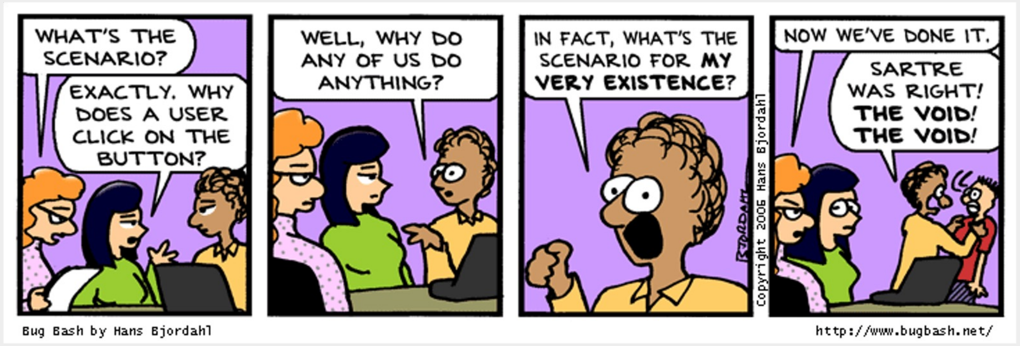
Pamela Zave & Michael Jackson, "Four Dark Corners of Requirements Engineering," *ACM Transactions on Software Engineering and Methodology,* 6(1): 1-30, 1997. https://www.cse.msu.edu/~chengb/RE-491/Papers/dark-corners-re-zave-jackson.pdf

Environment vs. Machine Example: Automobile



Delving into Requirements: System, Software, Assumptions

- **System requirements**: relationships between monitored and controlled variables
- **Software requirements**: relationship between inputs and outputs
- Domain properties and **assumptions** state relationships between those


WHAT'S THE SCENARIO?
EXACTLY. WHY DOES A USER CLICK ON THE BUTTON?
WELL, WHY DO ANY OF US DO ANYTHING?
IN FACT, WHAT'S THE SCENARIO FOR MY VERY EXISTENCE?
NOW WE'VE DONE IT.
SARTRE WAS RIGHT! THE VOID! THE VOID!
Copyright 2006 Hans Bjordahl
Bug Bash by Hans Bjordahl
http://www.bugbash.net/

Environment vs. Machine

Example: Airbus Braking System

- The Airbus A320-200 airplane has a software based braking system that consists of:
  - Ground spoilers (wing plates extend to reduce lift)
  - Reverse thrusters
  - Wheel brakes on the main landing gear
- “To engage the braking system, the wheels of the plane must be on the ground”
  - Is this a shared or an unshared action/condition?

Lufthansa Flight 2904 crashed on September 14, 1993

Lufthansa Flight 2904 (Cont'd)

- There are two "on ground" conditions:
  1. Either the shock absorber bears a load of 6300 kgs
  2. Both wheels turn at 72 knots (83 mph) or faster

- Ground spoilers activate for conditions 1 or 2
- Reverse thrust activates for condition 1 on both main landing gears
- Wheel brake activation depends upon the rotation gain and condition 2

Why didn’t it Stop? (2 died, 56 injured)

- There is no way for the pilots to override the software decision manually
- The thrust reversers are only activated if the first condition is true
- The braking system was not activated
  - Point one was not fulfilled because the plane landed inclined (to counteract crosswind). Thus, the required pressure on both landing gears was not reached.
  - Point two was not fulfilled due to a hydroplaning effect on the wet runway.

ATM Example

- Actions of an ATM customer:
  - withdrawal-request(a, m)
- Properties of the environment:
  - balance(b, p)

- Actions of an ATM machine:
  - withdrawal-payout(a, m)
- Properties of the machine:
  - expected-balance(b,p)

What other **models of the world** do machines maintain?

Implementation Bias

- Requirements say what the system will do (and not how it will do it)
  - Why *not* "how"?
- Requirements describe what is observable at the environment-machine interface
- **Indicative mood** describes the environment (as-is)
- **Optative mood** to describe the environment with the machine (to-be)

Online Shopping Example

- Stories: Scenarios and Use Cases
  - "After the customer submits the purchase information and the payment has been received, the order is fulfilled and shipped to the customer's shipping address."
- Optative statements
  - "The system **shall** notify clients about their shipping status"
- Domain properties and Assumptions
  - "Every product has a unique product code"
  - "Payments will be received after authorization"

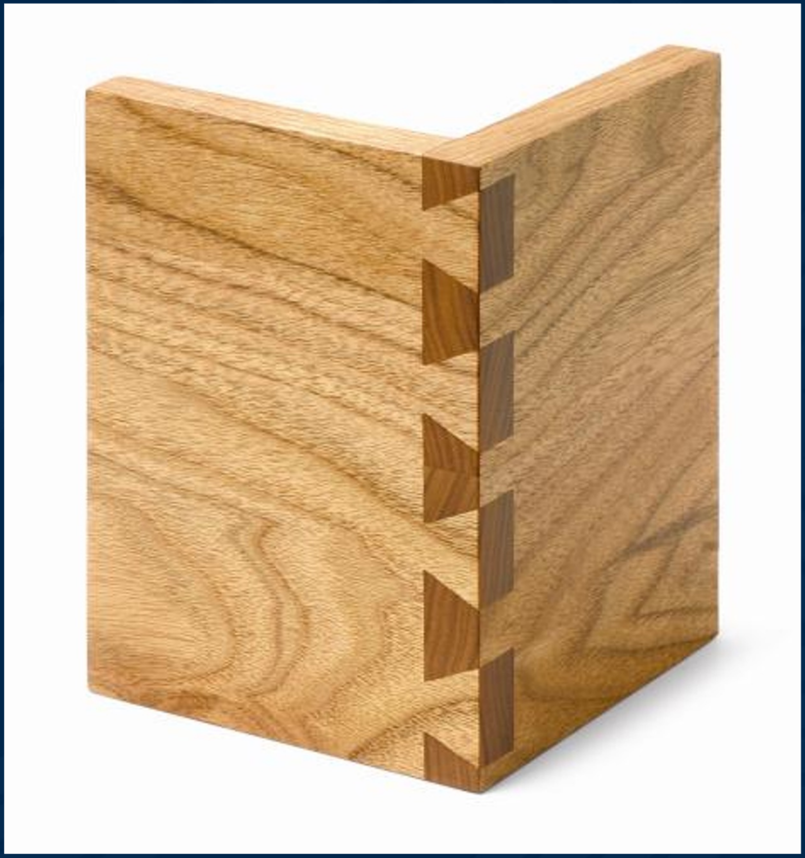
MICHIGAN ENGINEERING
UNIVERSITY OF MICHIGAN

Trivia Break

MICHIGAN ENGINEERING
UNIVERSITY OF MICHIGAN

Trivia: Woodworking

- This type of joinery uses a series of trapezoidal “pins” in one board that interlock with another board to resist being pulled apart. It is believed to predate written history, with examples in the tombs of Chinese emperors and entombed with First Dynasty Egyptian mummies.



Trivia: Chinese History (梁紅玉)

- This Song Dynasty war leader became famous in the Jin-Song wars (ca. 1135). In various accounts she is described as a martial artist, wrestler and archer who bought her own way out of slavery. She married Han Shizhong and later gave important information to him allowing a coup to be crushed and Emperor Gaozong to be restored. She was rewarded with a noble rank of 護國夫人 ("Noble Lady of Hu Guo" or "Lady Protector of the Nation"), which is relatively unique in Chinese history: most noble women obtained ranks through their husbands. Many folk legends, operas, and novels feature this "red jade" figure.

https://en.wikipedia.org/wiki/Liang_Hongyu

MICHIGAN ENGINEERING
UNIVERSITY OF MICHIGAN

Psychology: Belief

- What factors influence our belief in a statement?
  - “You only use 10 percent of your brain. Eating carrots improves your eyesight. Vitamin C cures the common cold. Crime in the United States is at an all-time high.”
- We would like factors such as “evidence” or “validity” to be dominant
- Today we consider “repetition and “ease”

Psychology: Belief

- Subjects were asked to rate how certain they were that 60 statements were true or false
  - "Zachary Taylor was the first president to die in office".
  - "Lithium is the lightest of all metals."
  - "The largest museum in the world is the Louvre in Paris."

- Critically, subjects gave ratings on three successive occasions at two week intervals

MICHIGAN ENGINEERING
UNIVERSITY OF MICHIGAN

Psychology: Illusory Truth Effect

- For both true and false statements, there was a significant increase in the validity of judgements for the repeated statements (and no change for non-repeated ones)

[Lynn Hasher, David Goldstein, Thomas Toppino. Frequency and the Conference of Referential Transparency. J. Verbal Learning and Verbal Behavior, 1977.]

Psychology: Illusory Truth Effect

- Participants were exposed to false news stories portrayed as true news stories. After a five-week delay, participants who had read the false experimental stories rated them as more truthful and more plausible than participants who had not been exposed to the stories. In addition, there was evidence of the creation of false memories for the source of the news story. *Participants who had previously read about the stories were more likely to believe that they had heard the false stories from a source outside the experiment.* These results suggest that repeating false claims will not only increase their believability but may also result in source monitoring errors.

[Danielle Polage. Making up History: False Memories of Fake News Stories. Europe J. Psychology, 2012.]

MICHIGAN ENGINEERING
UNIVERSITY OF MICHIGAN

Psychology: Illusory Truth Effect

- "When people judge the truth of a claim, related but nonprobative information rapidly leads them to believe the claim: an effect called "truthiness". … Across all experiments, easily pronounced names trumped difficult names. Moreover, the effect of pronounceability produced truthiness for claims attributed to those names."

  [People with Easier to Pronounce Names Promote Truthiness of Claims. PLOS ONE, 2014.]

- Implications for SE? Process and requirements decisions are made based on evidence and claims. Who said: "Slogans should be persistently repeated until the very last individual has come to grasp the idea."

MICHIGAN ENGINEERING
UNIVERSITY OF MICHIGAN

Functional and Quality Requirements

Functional & Quality Requirements

- Functional and quality requirements are two types of requirements that specify what a software system should do and how well it should do it.
- Functional requirements describe the features and functions of the system, such as what inputs it should accept, what outputs it should produce, and what behaviors it should perform.
- Quality requirements, also known as non-functional requirements, describe the characteristics and constraints of the system, such as how fast, reliable, secure, or user-friendly it should be.
- Both types of requirements are important for defining the scope and quality of the software system and ensuring that it meets the needs and expectations of the customers and users.

https://en.wikipedia.org/wiki/Functional_requirements

Examples of Functional & Quality Requirements

- Functional requirement: The system must allow users to create, edit, and delete their profiles.
- Quality requirement: The system must respond to user requests within 2 seconds.
- Functional requirement: The system must encrypt the user passwords before storing them in the database.
- Quality requirement: The system must comply with the GDPR regulations.
- Functional requirement: The system must generate a monthly report of sales and expenses.
- Quality requirement: The system must be able to handle up to 100 concurrent users.

Functional Requirements

- Functional requirements describe what the machine should do ("get the right answer")
  - Input, output
  - Interface
  - Response to events
- Criteria
  - Completeness: All requirements are documented
  - Consistency: No conflicts between requirements
  - Precision: No ambiguity in requirements

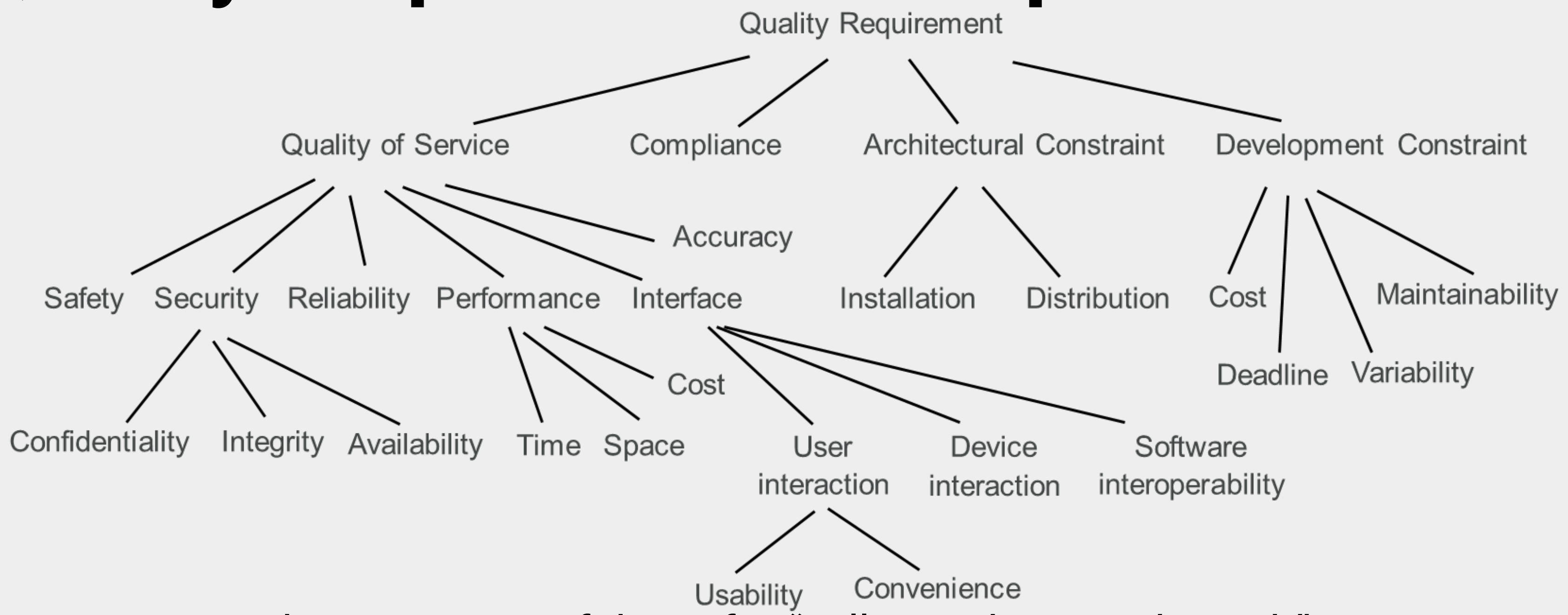
Quality Requirements

- Quality requirements specify not the functionality of the system, but the manner in which it delivers that functionality
- Can be more critical than functional requirements
  - Can work around missing functionality
  - Low-quality systems may be unusable
- Examples?

Framing the Question

- Who is going to ask for a slow, inefficient, unmaintainable system?
- A better way to think about quality requirements is as **design criteria to help choose between alternative implementations**
- The question becomes: *to what extent* must a product satisfy these requirements to be acceptable?

Quality Requirement Examples



What are some of these for “selling videos on the web”?
(possibly: “with your partner ...”)

Expressing Quality Requirements

- Requirements serve as contracts: they should be testable/falsifiable
- An **informal goal** is general intention (e.g., “ease of use” or “high security”)
    - May still be helpful to developers as they convey the intentions of the system users
- A **verifiable** non-functional requirement is a statement using some measure that can be objectively tested.

Informal vs. Verifiable Example

- **Informal goal**: "the system should be easy to use by experienced controllers, and should be organized such that user errors are minimized."
- **Verifiable non-functional requirement**:
  "Experienced controllers shall be able to use all the system functions after a total of two hours training. After this training, the average number of errors made by experienced users shall not exceed two per day, on average."

Quality Requirement Examples

- Confidentiality requirement: A non-staff patron may never know which books have been borrowed by others
- Privacy requirement: The calendar constraints of a participant may never be disclosed to other invited participants without consent
- Integrity requirement: The return of book copies shall be encoded correctly and by library staff only
- Availability requirements:
  - A blacklist of bad patrons shall be made available at any time to library staff.
  - Information about train positions shall be available at any time to the vital station computer.

Quality Requirement Examples

- Reliability req: The train acceleration control software shall have a mean time between failures on the order of 100 hours
- Accuracy req:
  - A copy of a book shall be stated as available by the loan software if and only if it is actually available on the library shelves.
  - The information about train positions used by the train controller shall accurately reflect the actual position of trains up to 4 meters at most.
  - The constraints used by the meeting scheduler should accurately reflect the real constraints of invited participants.
- Performance req:
  - Repsonses to bibliographical queries shall take less than 2 seconds.
  - Acceleration commands shall be issued to every train every 3 seconds.
  - The meeting scheduler shall be able to accommodate up to 9 requests in parallel.
  - The new subscription facility should ensure a 30% cost saving.

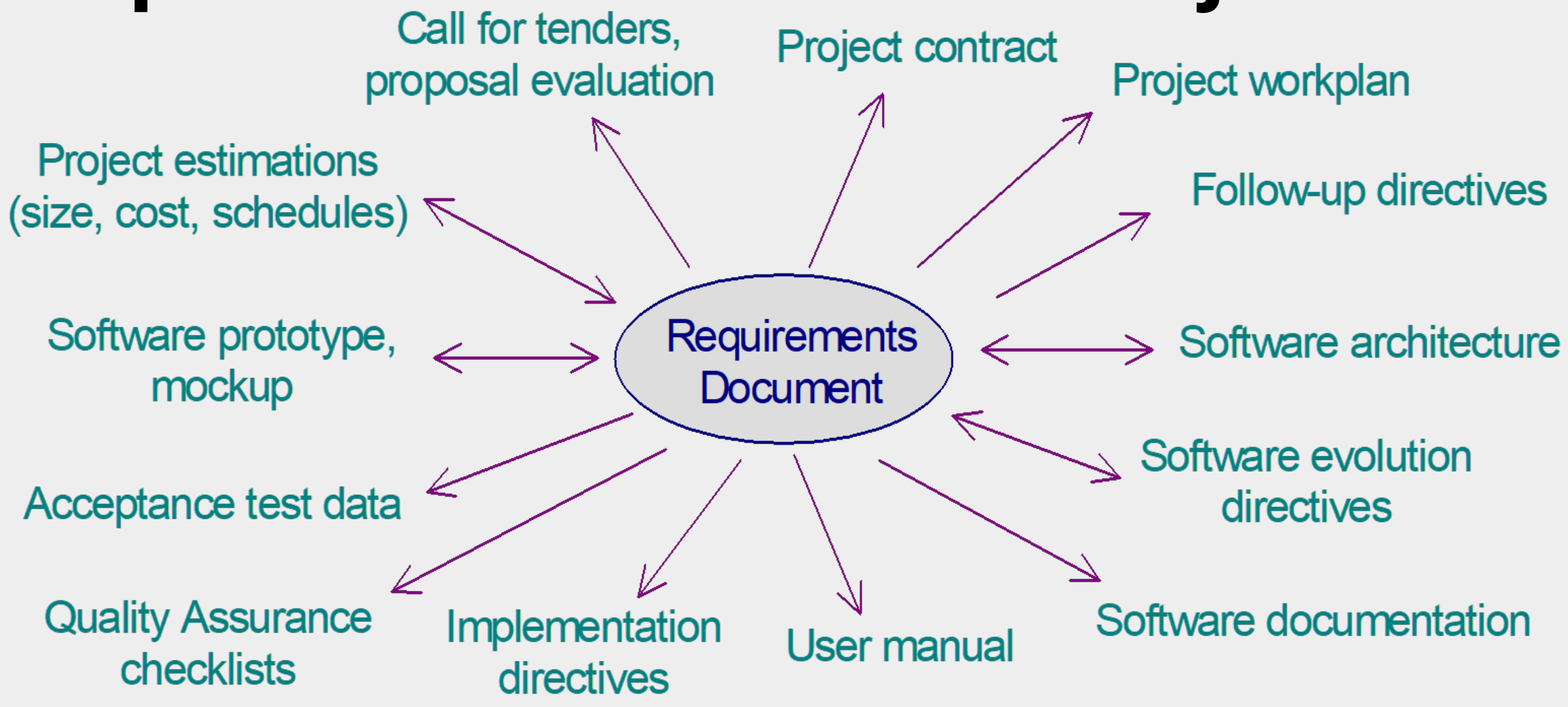
MICHIGAN ENGINEERING
UNIVERSITY OF MICHIGAN

Requirements Engineering

Requirements Engineering

- **Knowledge acquisition**: how to capture relevant detail about a system
  - Is the knowledge complete and consistent?
- **Knowledge representation**: once captured, how do we express it most effectively
  - Express it for whom?
  - Is it received consistently by different people?
- You may sometimes see a distinction between the requirements *definition* and the requirements *specification*

Requirements in Software Projects


Requirements Document
Call for tenders, proposal evaluation
Project contract
Project workplan
Follow-up directives
Software architecture
Software evolution directives
Software documentation
User manual
Implementation directives
Quality Assurance checklists
Acceptance test data
Software prototype, mockup
Project estimations (size, cost, schedules)

Requirements Engineering: Typical Steps (Iterative)

- Identifying stakeholders
- Domain understanding
- Requirements elicitation (interviews, …)
- Evaluation and agreement (conflicts, prioritization, risks, …)
- Documentation and specification
- Consolidation and quality assurance (what?)

Target Qualities for RE Processes

- Completeness of objectives, requirements, assumptions
- Consistency of RD items
- Adequacy of requirements, assumptions, domain props
- Unambiguity of RD items
- Measurability of requirements, assumptions
- Pertinence of requirements, assumptions
- Feasibility of requirements
- Comprehensibility of RD items
- Good structuring of the RD
- Modifiability of RD items
- Traceability of RD items (where did we see this before?)

MICHIGAN ENGINEERING
UNIVERSITY OF MICHIGAN

Types of RE Errors and Flaws

- **Omission**
- **Contradiction**
- **Inadequacy**
- **Ambiguity**
- Unmeasurability
- Noise, over-specification
- Unfeasibility (wishful thinking)
- Unintelligibility
- Poor structuring, forward references
- Opacity

WE MAY BE THE ONLY LAWYERS ON EARTH WHOSE CLIENTS ARE ALL INNOCENT.
THE ANIMAL LEGAL DEFENSE FUND

dizzyotter

when i first saw this i thought the kittens were the lawyers

Omission and Contradiction

- Omission: problem/world feature not stated by any RD item
  - e.g., no req about state of train doors in case of emergency stop
- Contradiction: RD items stating a problem/world feature in an incompatible way
  - “All doors must always be kept closed between platforms”
  - *and* “All doors must be opened in case of emergency stop”

Inadequacy and Ambiguity

- Inadequacy: RD item not clearly stating the problem/world feature (“I need more to go on”)
  - “Panels inside trains shall display flights served at next stop”
  - (Which panels? Which trains? Display how? What does “served” mean? *Flights* vs. Trains?)
- Ambiguity: RD item allowing a problem/world feature to be interpreted in different ways
  - All doors shall be opened as soon as the train is stopped at platform”
  - (When do you start opening the doors?)

Software Requirements Specification (SRS)

- A software requirements specification (SRS) is a document that describes what the software will do and how it will be expected to perform.
- It also describes the functionality the product needs to fulfill the needs of all stakeholders (business, users). An SRS is a blueprint or roadmap for the software you're going to build. It helps you to define your product's purpose, describe what you're building, detail the requirements, and deliver it for approval.
- An SRS can help you to ensure the safety, reliability, and quality of your software product, as well as to comply with regulatory standards and customer expectations. An SRS can also help you avoid misunderstandings, conflicts, and rework during the software development process.

https://relevant.software/blog/software-requirements-specification-srs-document/

https://en.wikipedia.org/wiki/Software_requirements_specification

https://www.perforce.com/blog/alm/how-write-software-requirements-specification-srs-document

SRS Sections

- Purpose: This section explains the main objectives and goals of the software product, and why it is needed.
- Scope: This section defines the boundaries and limitations of the software product, and what it will and will not do.
- Definitions: This section provides the definitions of terms, acronyms, and abbreviations used in the document.
- Background: This section provides the context and background information about the software product, such as its history, motivation, or related projects.
- System overview: This section gives a high-level overview of the software system, its components, interfaces, and interactions.

SRS Sections (Cont'd)

- References: This section lists the sources of information that are used or cited in the document, such as standards, specifications, or documents.
- Overall description: This section describes the general characteristics and features of the software product, such as its perspective, functions, user characteristics, constraints, assumptions, and dependencies.
- Specific requirements: This section details the specific requirements of the software product, such as external interface requirements, performance requirements, logical database requirements, and software system attributes. It also specifies the functional and non-functional requirements of the software product, such as what it should do and how it should behave.
- Appendices: This section contains any additional or supplementary information that is relevant to the document, such as diagrams, tables, charts, or examples.

SRS Standards

- There are different formats and standards for writing an SRS document, such as IEEE 29148 or ISO/IEC/IEEE 24765.
- You can choose the one that best suits your project needs and preferences.
- However, regardless of the format or standard you use, an SRS document should be clear, concise, consistent, complete, correct, traceable, verifiable, modifiable, and organized.

https://www.computer.org/resources/software-requirements-specifications/

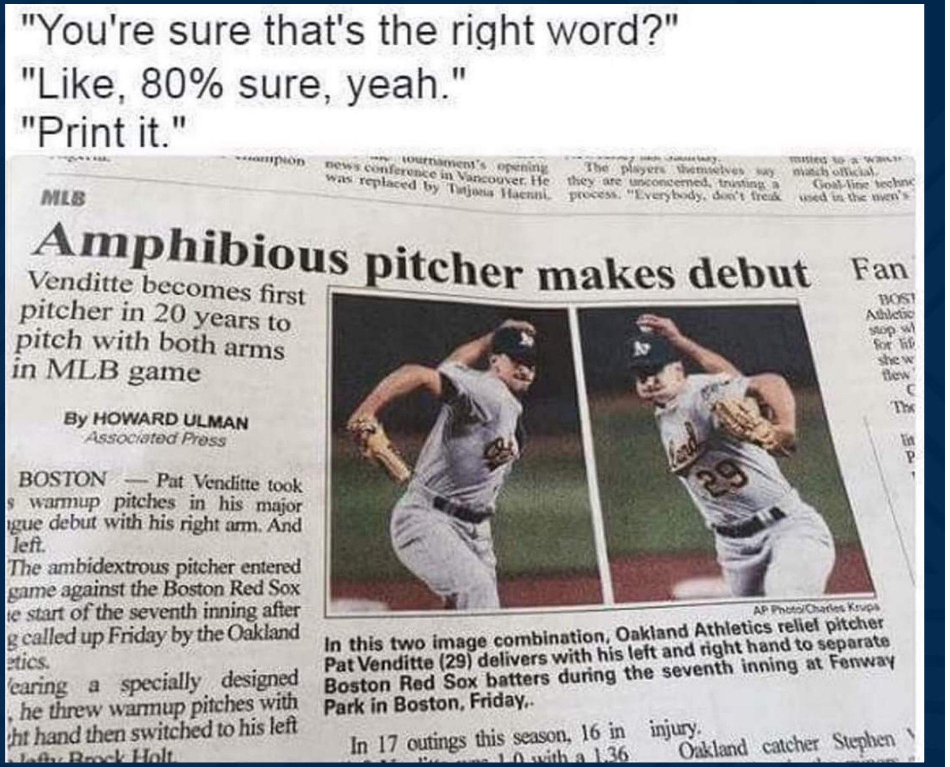
https://www.microtool.de/en/knowledge-base/what-is-a-software-requirements-specification/

Example

- System Design Mock Interview: Design TikTok ft. Google TPM
  - Shows requirements discussion, esp. first 10 min.
- https://www.youtube.com/watch?v=Z-0g_aJL5Fw

Questions?

- HW4 is due this Wednesday!
- .. and consider starting to work on HW5 and HW6a.



"You're sure that's the right word?"
"Like, 80% sure, yeah."
"Print it."

news conference in Vancouver. He was replaced by Tatjana Haenni.

The players themselves say they are unconcerned, trusting a process. "Everybody, don't freak

match official.
Goal-line techno used in the men's

MLB

Amphibious pitcher makes debut

Vénditte becomes first pitcher in 20 years to pitch with both arms in MLB game

By HOWARD ULMAN
Associated Press

BOSTON — Pat Venditte took s warmup pitches in his major gue debut with his right arm. And left.
The ambidextrous pitcher entered game against the Boston Red Sox e start of the seventh inning after g called up Friday by the Oakland etics.
earing a specially designed , he threw warmup pitches with ht hand then switched to his left

AP Photo/Charles Krupa

In this two image combination, Oakland Athletics relief pitcher Pat Venditte (29) delivers with his left and right hand to separate Boston Red Sox batters during the seventh inning at Fenway Park in Boston, Friday,.

In 17 outings this season, 16 in

injury.
Oakland catcher Stephen

Fan