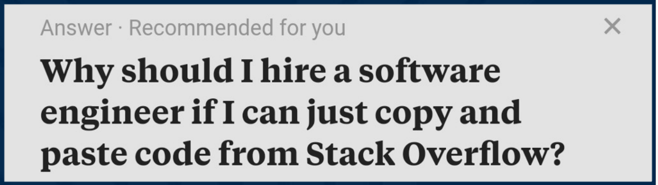
MICHIGAN ENGINEERING
UNIVERSITY OF MICHIGAN

Requirements, Validation, and Risk



Answer · Recommended for you

Why should I hire a software engineer if I can just copy and paste code from Stack Overflow?

SO, YOU DON'T THINK THERE ARE ANY INHERENT BIOLOGICAL DIFFERENCES BETWEEN MEN AND WOMEN?
OF COURSE NOT.
WHAT I THINK IS THAT, WHEN MEASURED, MOST OF THE DIFFERENCES ARE SMALL, AND SOME DISAPPEAR UNDER CROSS CULTURAL ANALYSIS, AND TO THE EXTENT THERE ARE DIFFERENCES THEY TEND TO BE WITHIN A SINGLE STANDARD DEVIATION, WHICH MEANS THEY'RE NOT MUCH USE FOR UNDERSTANDING INDIVIDUALS.
THE AREAS WHERE MEN AND WOMEN DIFFER STRONGLY TEND TO NOT MATTER MUCH IN THE MODERN WORLD. FOR INSTANCE, AN AVERAGE MAN IS ABOUT TWO STANDARD DEVIATIONS BETTER THAN AN AVERAGE WOMAN AT THROWING A ROCK.
WELL YEAH, THAT REQUIRES VECTOR CALCULATION.
I THINK YOU'RE CONSTRUCTING THE NARRATIVE YOU WANT TO BELIEVE.
THEY WERE PROBABLY THROWING MATH BOOKS TO EACH OTHER.
smbc-comics.com

EECS 481 (W24)

The Story so far…

- **Quality assurance** is critical to software engineering
- Ok, so we want to build a quality product.
  - What are we supposed to be building again?
- Remember Design of a system is a process of having a realization from a specification or a requirement.
- Validation is a process of ensuring that a realization satisfies its specification or requirement.
- We should ask the customer!
  - But how?

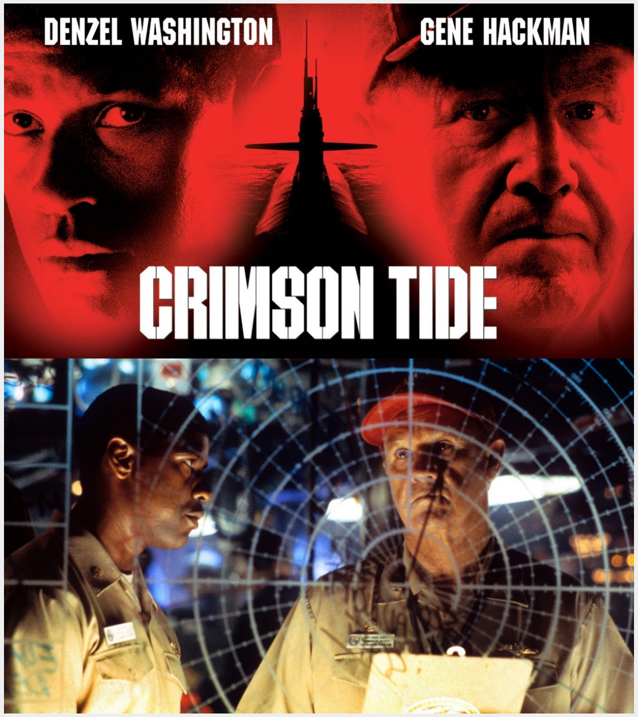
We've got a text from our customer. It says: "The sys must wrk."
What does that mean?
I think that was the requirements we asked for.

One-Slide Summary

- **Requirements elicitation** relies on communication with **stakeholders**. This includes identifying relevant parties, understanding the domain, interviews, and the exploration of alternatives. Requirements often conflict.
- **Validation in SE** checks the correctness of requirements;
- **verification in SE** checks the correctness of software.
- **Risk in SE** includes both the likelihood and the consequence of failure.

Outline (the emotional journey)

- Define Requirements Elicitation Process
- Talk through each step of process
  - Step 1 – Stakeholders
  - Step 2 – Domain Knowledge
  - Step 3 – Discover the real needs
  - Step 4 – Explore Alternatives
- Revisit Risk



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

1. (*knowledge*) describe the steps in requirements elicitation
2. (*knowledge*) provide examples of what can go wrong in interviews
3. (*knowledge*) list types of (requirements) conflicts and strategies for resolving them
4. (*knowledge*) explain the difference between verification and validation with respect to software
5. (*knowledge*) define risk response strategies and describe how to analyze risk

Step 1: Stakeholders

Requirements Elicitation

- **Requirements elicitation** is the process of identifying system requirements through communication with stakeholders Typically:
  - Step 1. Identify Stakeholders
  - Step 2. Understand the domain
    - Analyze artifacts, interact with stakeholders
  - Step 3. Discover the real needs
    - Interview stakeholders, resolve conflicts
  - Step 4. Explore alternatives to address needs

Stakeholder

- A stakeholder is a person or group who has an interest or concern in something, especially a business or an organization.
- Stakeholders can be internal or external to the entity they are involved with or affected by.
- For example, investors, employees, customers, and suppliers are common stakeholders of a corporation.
- They have a stake in the success or failure of the corporation, and they can influence or be influenced by its actions and outcomes.

Stakeholder

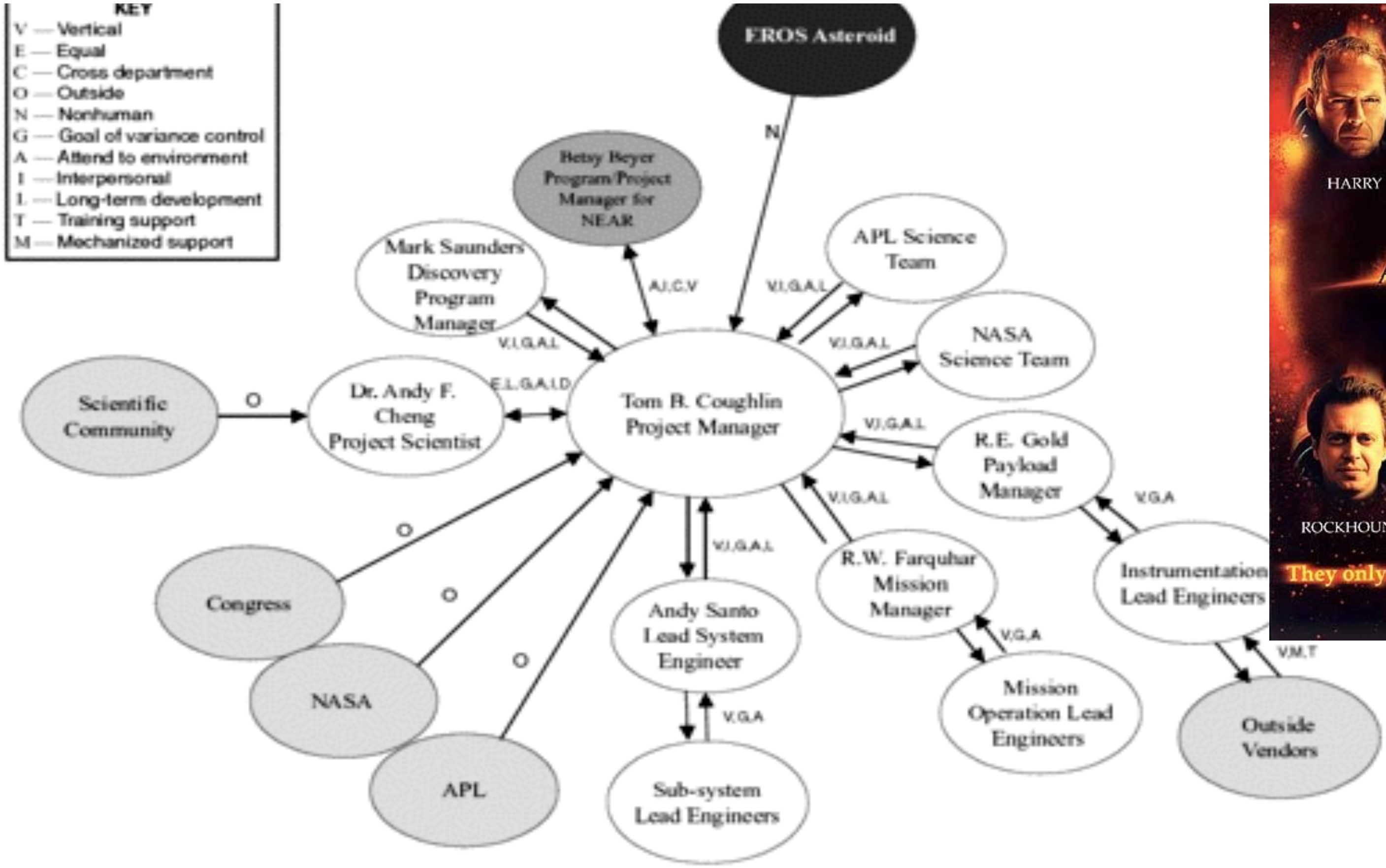
- A **stakeholder** is any person or group who will be affected by the system, directly or indirectly
  - Customers, other parts of your own organization, regulatory bodies, etc.
- Stakeholders may disagree
- Requirements process should trigger negotiations to resolve conflicts.
- (We will return to conflicts)

"Again this year, you get one wish... but please don't waste it on something even I can't grant, like clear business requirements."

Stakeholder Analysis

Common criteria for identifying relevant stakeholders include:

- Relevant positions in the organization
- Effective role in making decisions about the systems
- Level of domain expertise
- Exposure to perceived problems
- Influence in system acceptance
- Personal objectives and conflicts of interest

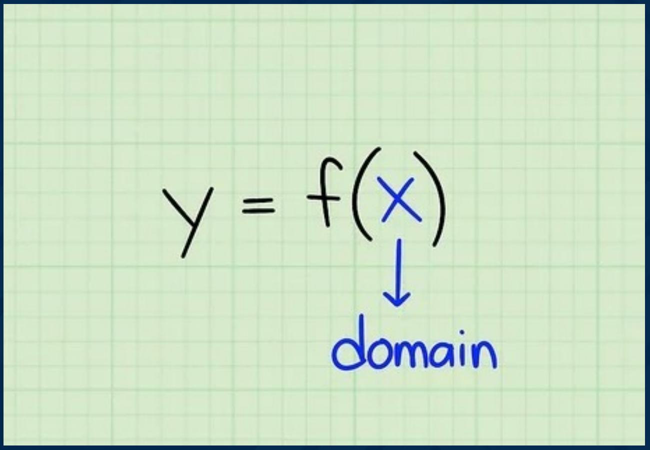

KEY
V — Vertical
E — Equal
C — Cross department
O — Outside
N — Nonhuman
G — Goal of variance control
A — Attend to environment
I — Interpersonal
L — Long-term development
T — Training support
M — Mechanized support
EROS Asteroid
N
Betsy Beyer Program/Project Manager for NEAR
A,I,C,V
Mark Saunders Discovery Program Manager
V,I,G,A,L
APL Science Team
V,I,G,A,L
NASA Science Team
V,I,G,A,L
Scientific Community
O
Dr. Andy F. Cheng Project Scientist
E,L,G,A,I,D
Tom B. Coughlin Project Manager
V,I,G,A,L
R.E. Gold Payload Manager
V,G,A
Instrumentation Lead Engineers
V,M,T
Outside Vendors
V,I,G,A,L
R.W. Farquhar Mission Manager
V,G,A
Mission Operation Lead Engineers
V,I,G,A,L
Andy Santo Lead System Engineer
V,G,A
Sub-system Lead Engineers
Congress
O
NASA
O
APL
O

HARRY
GRACE
A.J.
ARMAGEDDON
AT THEATRE 7th AUGUST 1998
ROCKHOUND
BEAR
TRUMAN
They only have 33 days to save the wo
Do it or die !!

FIGURE 6-3 Role network for National Aeronautics and Space Administration (NASA's) Near Earth Asteroid Rendezvous project.

MICHIGAN ENGINEERING
UNIVERSITY OF MICHIGAN

Step 2: Understanding Domain

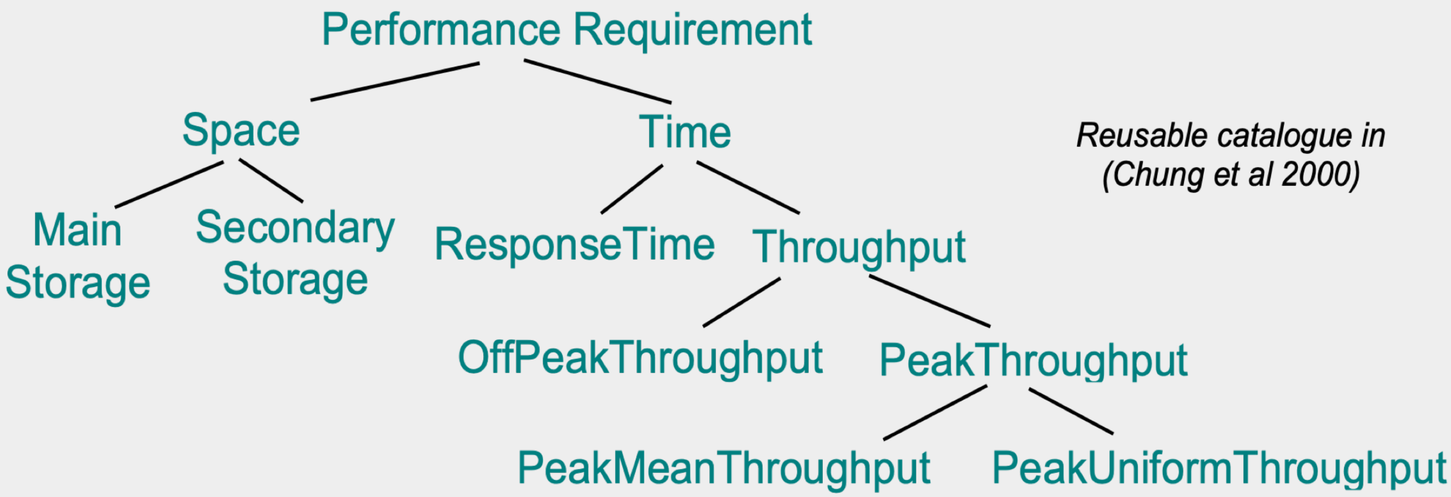


Step 2: Understanding the Domain

- **Content analysis** involves learning about the system domain
  - Books, articles, wikipedia, etc.
- This often focuses on the system to be built or replaced
  - How does it work? What are the problems? Are there manuals? Bug reports?
- But it also involves the organization
- And reusing knowledge from other systems

Domain-Independent Checklist

- Consider the list of qualities (from previous lecture) and select the relevant ones
- Privacy, security, reliability, etc.
- Even “performance” can be complicated



Performance Requirement

Space

Time

Main Storage

Secondary Storage

ResponseTime

Throughput

OffPeakThroughput

PeakThroughput

PeakMeanThroughput

PeakUniformThroughput

Reusable catalogue in (Chung et al 2000)

Step 3: Interviews

Step 3: Discover Real Needs via Interviews

- Having identified stakeholders of interest and information to be gathered…
- Conduct an **interview**

YOUR USER REQUIREMENTS INCLUDE FOUR HUNDRED FEATURES.
DO YOU REALIZE THAT NO HUMAN WOULD BE ABLE TO USE A PRODUCT WITH THAT LEVEL OF COMPLEXITY?
GOOD POINT. I'D BETTER ADD "EASY TO USE" TO THE LIST.
www.dilbert.com
scottadams@aol.com
4/14/01 © 2001 United Feature Syndicate, Inc.

Step 3: Discover Real Needs via Interviews

- Having identified stakeholders of interest and information to be gathered …
- Conduct an **interview**
  - This can be structured or unstructured, individual or group, etc.
  - It may even be a simple phone call
- Record and transcribe interview
- Report important findings
- Check validity of report with interviewee

Requirements Interview Advice

- Get basic facts about the interviewee before (role, responsibilities, …)
- Review interview questions before interview
- Begin concretely with specific questions, proposals: work through prototype or scenario
- Be open-minded; explore additional issues that arise naturally, but stay focused on the system
- Contrast with current system or alternatives
    - Explore conflicts and priorities
- Plan for follow-up questions/sessions

Example: Identifying Problems (1)

- What problems do you run into in your day-to-day work? Is there a standard way of solving it, or do you have a workaround?
  - Why is this a problem? How do you solve the problem today? How would you ideally like to solve the problem?
- Keep asking follow-up questions ("What else is a problem for you?", "Are there other things that give you trouble?") for as long as the interviewee has more problems to describe

Example: Identifying Problems (2)

- So, as I understand it, you are experiencing the following problems/needs …
  - Describe the interviewee's problems and needs in your own words: often **you do not share** the same image. It is very very common to not understand each other even if at first you think you do.
- Just to confirm, have I correctly understood the problems you have with the current solution?
  - Are there any other problems you're experiencing? If so, what are they?

Interview Tradeoffs

- Strengths
  - Reveal what stakeholders do, feel, prefer
  - How they interact with the system
  - Challenges with current systems
- Weaknesses
  - Subjective, yield inconsistencies
  - Hard to capture domain knowledge
  - Organizational issues, such as politics
  - Hinges on interviewer skill

Capturing and Synthesizing

- We acquire requirements from many sources
  - Elicit from stakeholders
  - Extract from policies or other documentation
  - Synthesize from above: estimation and invention
- Stakeholders do not always know what they want (!)
  - Be faithful to stakeholder needs and expectations
  - Anticipate additional needs and risks
  - Validate that “additional needs” are necessary or desired

Problems with stakeholder interviews

- They can be expensive and time-consuming, especially if there are many stakeholders to interview, or if they are located in different places.
- Therefore, it is important to plan and budget the interviews carefully, and to select the most relevant and representative stakeholders to interview.

https://www.epa.gov/international-cooperation/public-participation-guide-stakeholder-interviews

Problems with stakeholder interviews

- They require skilled interviewers who can ask the right questions, listen actively, probe deeper, and build rapport and trust with the interviewees.
- Interviewers also need to be aware of their own biases and assumptions and avoid leading or influencing the interviewees' responses.
- Therefore, it is important to train and prepare the interviewers well, and to use a consistent and structured interview protocol.

Problems with stakeholder interviews

- They may elicit negative or conflicting responses from the interviewees, who may have different opinions, interests, or agendas.
- Some interviewees may also be reluctant or resistant to share information or feedback or may provide inaccurate or misleading information.
- Therefore, it is important to validate and triangulate the data collected from the interviews with other sources and methods, such as surveys, observations, or documents.

https://methods.18f.gov/discover/stakeholder-and-user-interviews/.

Analogy: Ethnography

(Dr. Margaret Mead in Samoa, 1975)

Observation and Ethnography

- Observe people using their current system
- Passive: no interference with task performers
  - Watch from outside, record (notes, video), edit transcripts, interpret
  - Protocol analysis: they concurrently explain it
- Active: you get involved in the task, even become a team member
- Ethnographic studies, over long periods of time, discover emergent properties of social group involved

Margaret Mead: an American Cultural Anthropologist

- In her popular 1928 book, *Coming of Age in Samoa*, Mead presented Samoan culture as a social system that allowed many adolescents to experiment sexually before marriage
  - Based on observations, interviews, ethnographic studies, etc.
- Mead almost certainly had a political agenda (she was a sexual progressive, etc.)
  - But that did not make her wrong

Lessons Learned: Cultural Determinism

Dr. Margaret Mead's studies in Samoa, particularly her work published in "Coming of Age in Samoa" in 1928, provided several key lessons and insights:

- Cultural Determinism: Mead's findings suggested that adolescence and its associated stresses are not solely biological but are significantly influenced by cultural factors.
- Her observations indicated that Samoan girls experienced a more relaxed adolescence compared to their American counterparts, which she attributed to the different cultural expectations and social structures.

Lessons Learned: Nature vs. Nurture Debate

- Nature vs. Nurture Debate: Mead's work contributed to the ongoing discussion about the relative importance of genetic factors (nature) and environmental influences (nurture) in human development.
- She argued for the strong role of nurture, proposing that cultural upbringing plays a crucial part in shaping individual behavior.

Lessons Learned: Sexual Norms and Gender Roles

- Sexual Norms and Gender Roles: The study challenged Western views on sex, family structure, and gender roles by presenting a society with different norms and practices.
- Mead's observations suggested that the openness and fluidity of sexual norms in Samoan culture contributed to the ease of adolescent development.

Lessons Learned: Ethnographic Methodology

- Ethnographic Methodology: Mead's systematic and immersive approach to fieldwork set a precedent for future anthropological studies.
- She emphasized the importance of living among the people being studied to gain a deeper understanding of their culture.

Controversy and Criticism

- Controversy and Criticism: Mead's conclusions were subject to controversy.
- Some anthropologists, notably Derek Freeman, later challenged the accuracy of her findings, arguing that she had been misled by her informants.
- This criticism led to a reevaluation of her work and a broader discussion on the complexities of field research and the interpretation of anthropological data.

Mead vs. Freeman

- In 1983, Derek Freeman's *Margaret Mead and Samoa: The Making and Unmaking of an Anthropological Myth*, suggested that Mead was just gullible. Two of her informants had been lying: "Never can giggly fibs have had such far-reaching consequences in the groves of Academe."
  - This significantly discredited her work
- It seemed his follow-on interviews found very different results. How could that be?

Mead vs. Freeman (Cont’d)

Freeman was lying

- In 1996, Martin Orans used Mead's notes to show that “such humorous fibbing could not be the basis of Mead's understanding. Freeman asks us to imagine that the joking of two women, pinching each other as they put Mead on about their sexuality and that of adolescents, was of more significance than the detailed information she had collected throughout her fieldwork.”

Mead vs. Freeman (Cont’d)

- In 2011, Paul Shankman used Derek Freeman's own notes and found that his interviews were conducted in problematic ways:
  - “The 1987 interview with Fa'apua'a was arranged and carried out by Fofoa's son, a Samoan Christian of high rank who was convinced that Mead had besmirched the reputation of Samoans by portraying his mother, her friend Fa'apua'a, and other Samoans as sexually licentious.”
  - “Fofoa's son told Fa'apua'a "that the purpose of the interview was to correct 'the lies she [Mead] wrote in her book, lies that insult you all.'"

Mead vs. Freeman (Cont'd)

- Shankman notes that "there is no information on the sex from these two women in Mead's field notes": she could not have been fooled by women who were not her informants
  - Instead, she drew her conclusions from data on 25 adolescent girls, of whom over 40% were sexually active, and interviews with men and women
- While she may have downplayed some aspects of Samoan sexuality (e.g., rape and physical punishment for those who violated norms), she did not invent a false narrative

MICHIGAN ENGINEERING
UNIVERSITY OF MICHIGAN

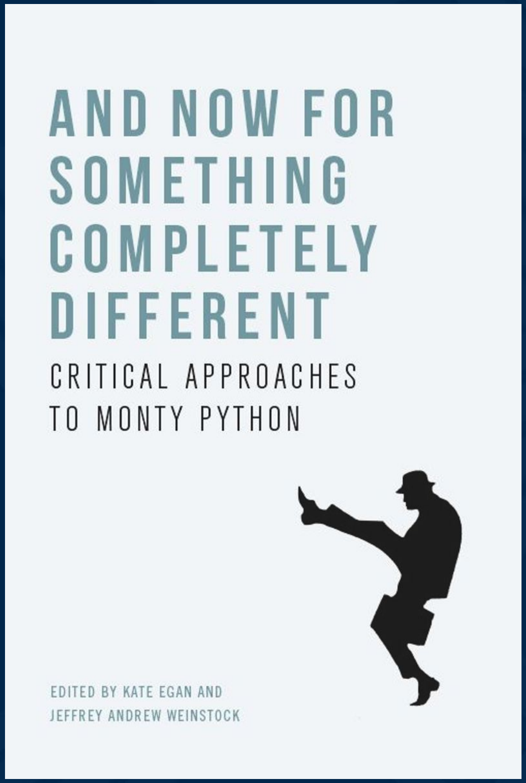
Requirements Interviews vs. Ethnography

- Why am I telling you so much about ethnography and cultural anthropology?
- Want to read more? Try "Sex, Lies, and Separating Science From Ideology": https://www.theatlantic.com/health/archive/2013/02/sex-lies-and-separating-science-from-ideology/273169/

If you were in the middle of the room the whole time, why can we not find a single witness to corroborate your testimony?
BIZARROCOMICS.COM
Facebook.com/BizarroComics
12·4·12
DAN PIRARO

©Dan Piraro.

Trivia Break



AND NOW FOR SOMETHING COMPLETELY DIFFERENT

CRITICAL APPROACHES TO MONTY PYTHON

EDITED BY KATE EGAN AND JEFFREY ANDREW WEINSTOCK

Trivia: Western Philosophy

- Identify the philosopher associated with each quote:
- “Man is by nature a political animal.” (~350 BCE)
- “All human knowledge begins with intuitions, proceeds from thence to concepts, and ends with ideas.” (1781)
- “More natural is our position in politics: We see problems of power, of one quantum of power against another. We do not believe in any right that is not supported by the power of enforcement: we feel all rights to be conquests.” (1888)
- “It is nonsense to assert that revelry, vice, ecstasy, passion, would become impossible if man and woman were equal in concrete matters.” (1949)

Trivia: Countries

- This country unified from three kingdoms into a singular political entity in 676. It gave rise to the world's first metal movable type (13th century) and a lovely constructed alphabet (15th century) but was weakened by Mongol invasions and annexation by Japan. Its largest city is the fourth most economically powerful in the world, measured by GDP.

Conflict Resolution

Conflicts Resolutions

- Conflict resolution in software engineering is the process of identifying, analyzing, and resolving conflicts that arise among software stakeholders, such as developers, managers, customers, and users.
- Conflicts can occur due to various reasons, such as different goals, expectations, opinions, preferences, values, or perspectives.
- Conflicts can also affect various aspects of software development, such as requirements, design, implementation, testing, or maintenance.

https://blog.logrocket.com/handling-conflict-on-a-software-engineering-team/

Conflicts Resolutions (Cont’d)

- Conflict resolution in software engineering is important for ensuring the quality and success of software projects.
- Conflicts can have negative impacts on the software product, such as errors, defects, delays, or failures.
- Conflicts can also have negative impacts on the software process, such as reduced productivity, efficiency, collaboration, or satisfaction.
- Therefore, conflict resolution in software engineering aims to find solutions that satisfy the needs and interests of all parties involved, and that improve the software product and process.

https://leaddev.com/culture-engagement-motivation/managing-conflict-engineering-teams.

Methods for Conflicts Resolutions

Communication: This involves exchanging information and feedback among stakeholders to understand the sources and effects of conflicts, and to express their views and feelings. Communication can be verbal or written, formal or informal, direct or indirect.

https://thesai.org/Downloads/Volume7No10/Paper_44-Software_Requirements_Conflict_Identification.pdf.

Negotiation: This involves discussing and bargaining among stakeholders to reach a mutually acceptable agreement or compromise.

Negotiation can be cooperative or competitive, distributive or integrative.

https://medium.com/swlh/handling-conflicts-in-software-engineering-teams-2e537e9f5d33.

Methods for Conflicts Resolutions (Cont’d)

Mediation: This involves involving a third party who facilitates the communication and negotiation among stakeholders to help them find a solution. The mediator does not impose a solution but rather assists the stakeholders in reaching one.

Arbitration: This involves involving a third party who evaluates the arguments and evidence of stakeholders and makes a binding decision for them. The arbitrator acts as a judge who imposes a solution based on rules and criteria.

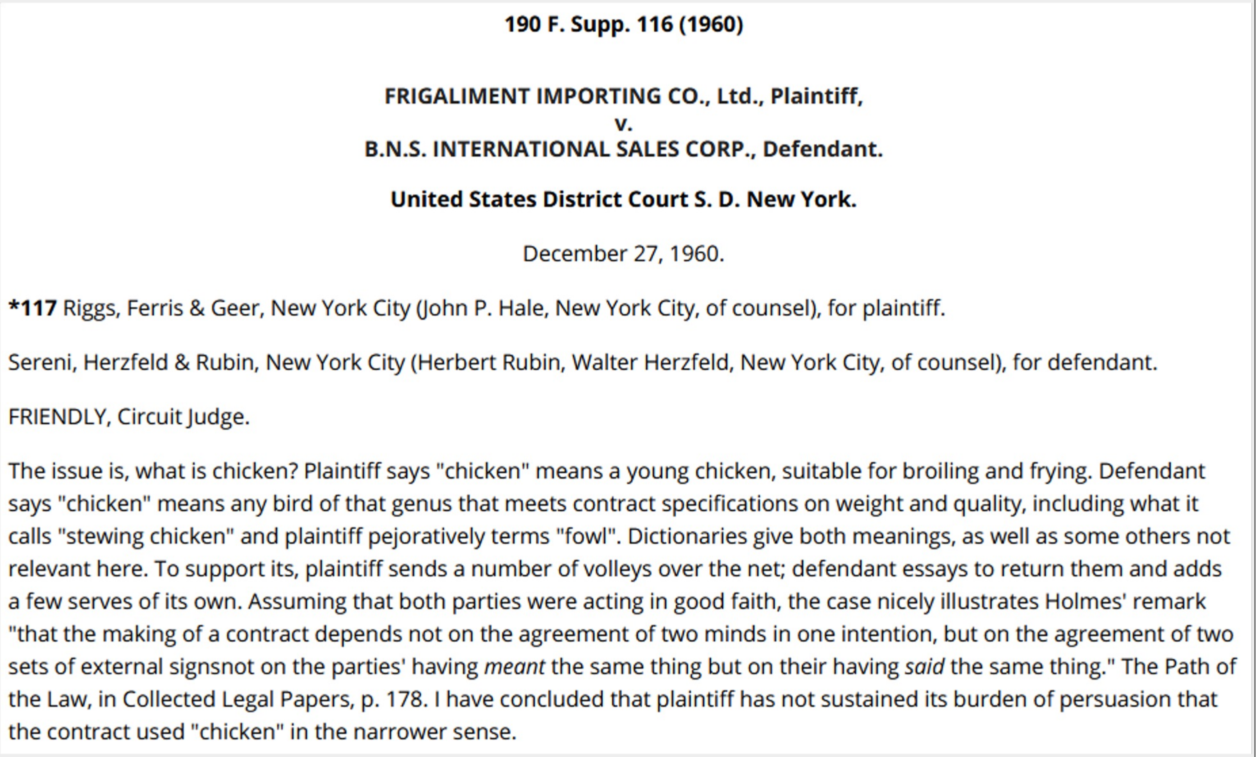
Identifying Conflicts: Inconsistencies

- Terminology clash: same concept named differently in different statements
  - e.g., library: “borrower” vs. “patron”
- Designation clash: same name for different concepts in different statements
  - e.g., “user” for “library user” vs. “library software user”
- Structure clash: same concept structured differently in different statements
  - e.g., “latest return date” as time point (e.g. Fri 5pm) vs. time interval (e.g. Friday)

Conflict Strength

- In a **strong conflict**, statements are not satisfiable together
  - e.g., “participant constraints may not be disclosed to anyone else” vs. “the meeting initiator must know participant constraints”
- In a **weak conflict** (**divergence**), statements are not satisfiable together under some boundary condition
  - e.g., “patrons shall return borrowed copies within X weeks” vs “patrons may keep borrowed copies as long as needed” contradicts only if “needed>X”

Contracts "In Real Life"



190 F. Supp. 116 (1960)

FRIGALIMENT IMPORTING CO., Ltd., Plaintiff,
v.
B.N.S. INTERNATIONAL SALES CORP., Defendant.

United States District Court S. D. New York.

December 27, 1960.

***117** Riggs, Ferris & Geer, New York City (John P. Hale, New York City, of counsel), for plaintiff.

Sereni, Herzfeld & Rubin, New York City (Herbert Rubin, Walter Herzfeld, New York City, of counsel), for defendant.

FRIENDLY, Circuit Judge.

The issue is, what is chicken? Plaintiff says "chicken" means a young chicken, suitable for broiling and frying. Defendant says "chicken" means any bird of that genus that meets contract specifications on weight and quality, including what it calls "stewing chicken" and plaintiff pejoratively terms "fowl". Dictionaries give both meanings, as well as some others not relevant here. To support its, plaintiff sends a number of volleys over the net; defendant essays to return them and adds a few serves of its own. Assuming that both parties were acting in good faith, the case nicely illustrates Holmes' remark "that the making of a contract depends not on the agreement of two minds in one intention, but on the agreement of two sets of external signsnot on the parties' having *meant* the same thing but on their having *said* the same thing." The Path of the Law, in Collected Legal Papers, p. 178. I have concluded that plaintiff has not sustained its burden of persuasion that the contract used "chicken" in the narrower sense.

https://law.justia.com/cases/federal/district-courts/FSupp/190/116/1622834/

Resolving Conflicts

- “No Silver Bullet” (this is why they pay you)
- For Terminology, Designation and Structural conflicts: build a glossary
- For Weak and Strong Conflicts: negotiation is typically required
  - If the cause is different stakeholder objectives, it must be resolved outside of RE
  - If the cause is quality desires (e.g., “Good, cheap, on-time: pick two”), you explore quality tradeoffs

MICHIGAN ENGINEERING
UNIVERSITY OF MICHIGAN

Step 4: Explore Alternatives

Step 4: Explore Alternatives

- Alternative solutions and tradeoffs are typically presented via **prototypes**, **mockups,** or **storyboards**
- Mockups can be low- or high-fidelity
- Rapid prototypes can be throw-away (designed to learn about the problem, not for actual use) or evolutionary (intended to be incorporated into the final product)
- Stories detail who the players are, what happens to them, how it happens, why it happens, and what could go wrong

Prototypes

- Prototypes in software engineering are incomplete or preliminary versions of software applications that are used to test the feasibility, design, functionality, and usability of the software product before developing the final product.
- Prototypes can help software engineers communicate with users and stakeholders, gather feedback and requirements, identify and resolve issues, and evaluate the performance and quality of the software product.
- Prototypes can also help software engineers to reduce the cost and risk of software development, as well as to improve customer satisfaction and loyalty.

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

https://www.geeksforgeeks.org/software-engineering-prototyping-model/

Mockups

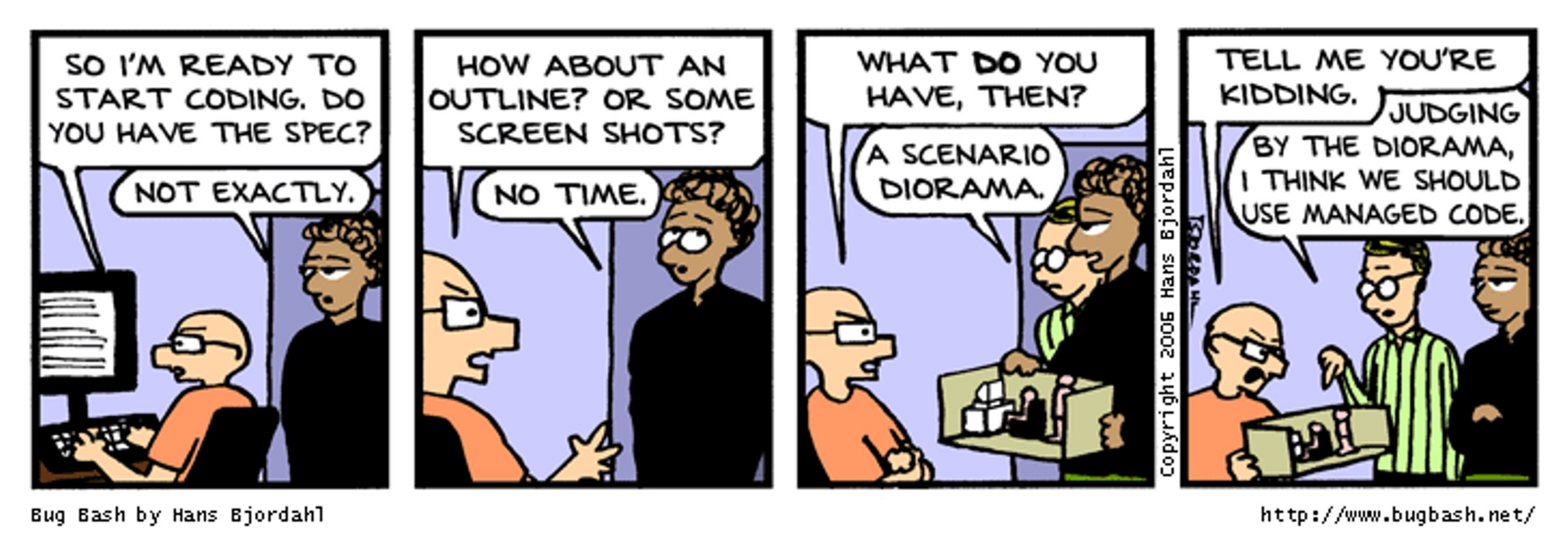
- Mockups in software engineering are a way of designing user interfaces on paper or in computer images, to show how the software product will look like, but without any functionality or interactivity.
- Mockups are used to communicate the design ideas, test the layout, color, typography, and navigation, and gather feedback from users and stakeholders.
- Mockups are usually created after wireframes, which are low-fidelity sketches of the basic structure and content of the software product, and before prototypes, which are high-fidelity simulations of the software product with some functionality and interactivity.
- Mockups can be created using various tools, such as Photoshop, Sketch, Figma, or UXPin.

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

https://www.uxpin.com/studio/blog/what-is-a-mockup-the-final-layer-of-ui-design/

Informality

- Storyboards and mockups definitely do exist, but are often informal and incomplete


SO I'M READY TO START CODING. DO YOU HAVE THE SPEC?
NOT EXACTLY.
HOW ABOUT AN OUTLINE? OR SOME SCREEN SHOTS?
NO TIME.
WHAT DO YOU HAVE, THEN?
A SCENARIO DIORAMA.
Copyright 2006 Hans Bjordahl
TELL ME YOU'RE KIDDING.
JUDGING BY THE DIORAMA, I THINK WE SHOULD USE MANAGED CODE.
Bug Bash by Hans Bjordahl
http://www.bugbash.net/

Exploration

- Humans are better at recognizing and evaluating solutions than facing blank pages
- Mockups and prototypes explore uncertainty in requirements
  - Validate that we have the right requirements
  - Get feedback on a candidate solution
  - “I'll know it when I see it.”
- Stories illuminate the system by walking through real or hypothetical sequences

Requirements Documentation

- Formal standards for writing down requirements exist (e.g., “may” vs. “must”) but are not a focus for this course
- They vary by domain and company (e.g., startup vs. established)

At last, he has found the famous Requirements Document dating back to the Traditional Age.

Requirements Elicitation: Reminder

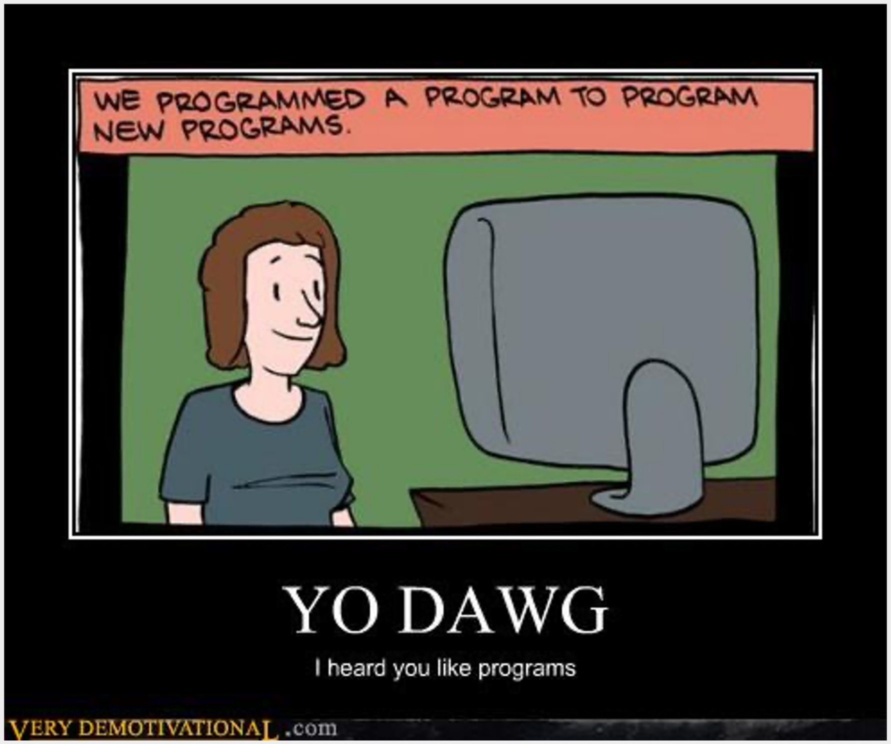
- **Requirements elicitation** is the process of identifying system requirements through communication with stakeholders. Typically:
    - Step 1. Identify stakeholders
    - Step 2. Understand the domain
        - Analyze artifacts, interact with stakeholders
    - Step 3. Discover the real needs
        - Interview stakeholders, resolve conflicts
    - Step 4. Explore alternatives to address needs

MICHIGAN ENGINEERING
UNIVERSITY OF MICHIGAN

Other aspects of Requirements

Requirements for Requirements?

- Correct
- Consistent
- Unambiguous
- Complete
- Feasible
- Relevant
- Testable
- Traceable


WE PROGRAMMED A PROGRAM TO PROGRAM NEW PROGRAMS.
YO DAWG
I heard you like programs
VERY DEMOTIVATIONAL .com

Verification and Validation in SE

- Validation is the task of determining if the requirements are correct
  - Are the requirements complete? Do they reflect the client's problem? Are they consistent?
- Verification is the task of determining if the software is correct (e.g., by testing)
  - Does the software satisfy the specification?
  - Is the specification correct with respect to the requirements, assuming the domain properties hold?

Approaches

Validation

- Interviews
- Reading
- Walkthroughs
- Prototypes
- Scenarios
- Checklists
- Modeling

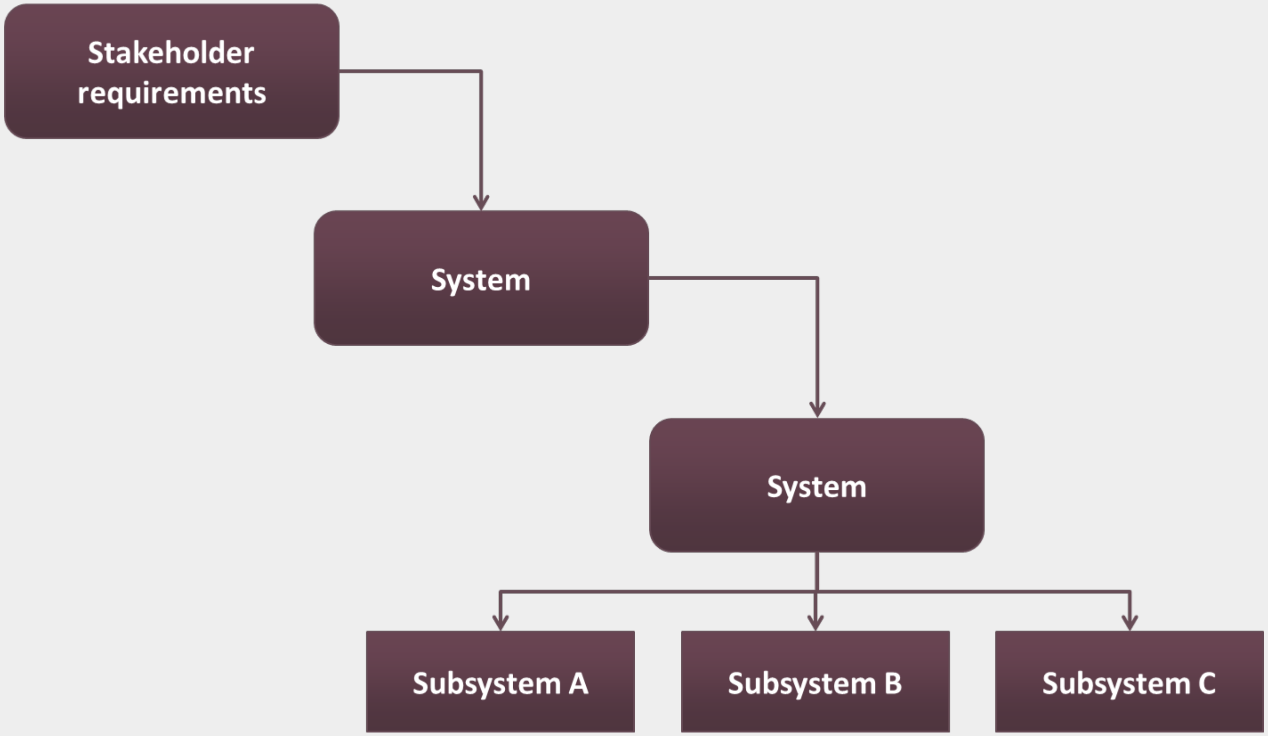
Verification

- Testing
- Mathematical proofs
- Simulation
- Static analysis
- Dynamic analysis
- Checks for unreachable states or transitions (model checking)

Decomposition

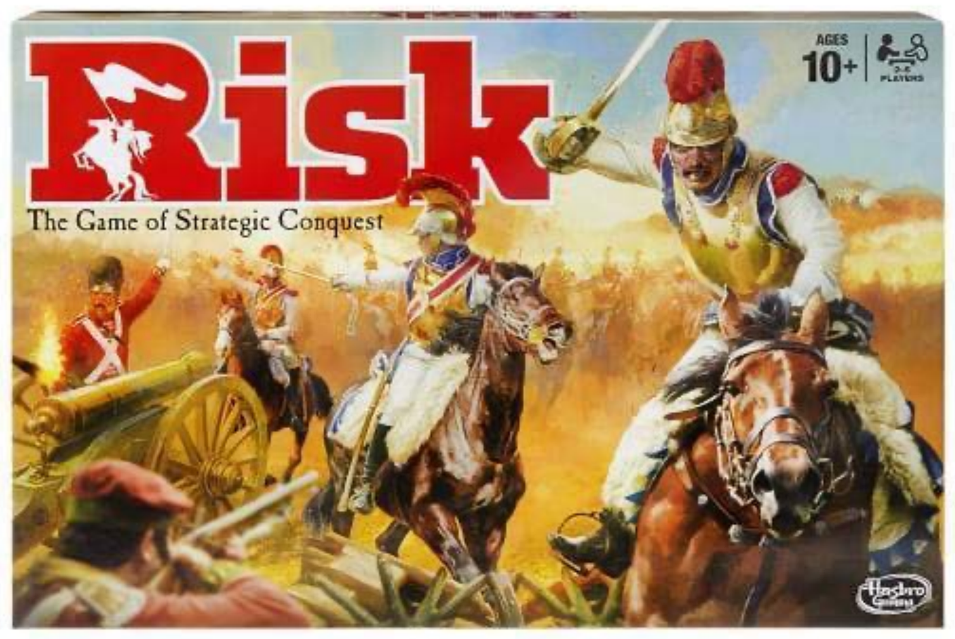
- We recursively **decompose** a system, from the highest level of abstraction (stakeholder requirements) into lower-level subsystems and implementation choices
- This decomposition establishes **traceability**, which identifies relationships between requirements and implementations
- Traceability is important for verification and when requirements change
- Decomposition helps both validate and verify

Decomposition Example



MICHIGAN ENGINEERING
UNIVERSITY OF MICHIGAN

Revisiting Risk

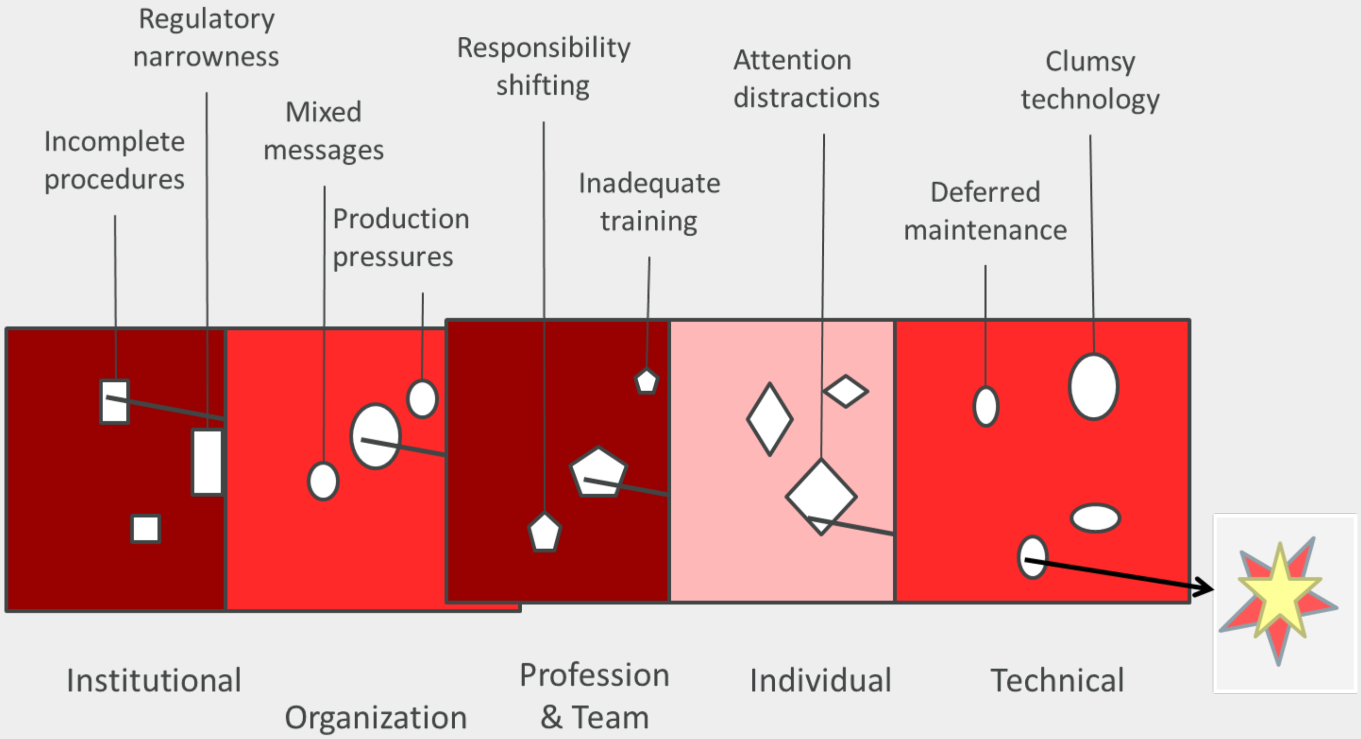


Risks

WE'LL NEED A RISK ANALYSIS ON THIS PROJECT BEFORE I CAN APPROVE IT.
www.unitedmedia.com
S. Adams
RISK 1: INDECISIVENESS
RISK 2: OVERANALYSIS
RISK 3: CLUELESSNESS
RISK 4: MICROMANAGE-MENT...
CLICK CLICK CLICK
© 1997 United Feature Syndicate, Inc.
I DON'T UNDERSTAND THESE RISKS.
THAT'S NUMBER THIRTY-SIX.

- In this context, a **risk** is an uncertain factor that may result in a loss of satisfaction of a corresponding objective
- For example:
  - The system delivers a radiation overdose to patients (Therac-25, Theratron-780)
  - Medication administration record (MAR) knockout (provided inaccurate medication plans hospital-wide)
  - Premier Election Solutions vote-dropping “glitch”

Swiss Cheese Model



Modified from Reason, 1999, by R.I. Crook

Risk Assessment

- Risk consists of multiple parts:
  - The likelihood of failure
  - The negative consequences or impact of failure
  - In advanced models: the causal agent and weakness
- Mathematically,
  Risk = Likelihood · Impact

WE SHOULD GO TO THE NORTH BEACH. SOMEONE SAID THE SOUTH BEACH HAS A 20% HIGHER RISK OF SHARK ATTACKS.
YEAH, BUT STATISTICALLY, TAKING THREE BEACH TRIPS INSTEAD OF TWO INCREASES OUR ODDS OF GETTING SHOT BY A SWIMMING DOG CARRYING A HANDGUN IN ITS MOUTH BY 50%!
OH NO! THIS IS OUR THIRD TRIP!
REMINDER: A 50% INCREASE IN A TINY RISK IS STILL TINY.

Risk Assessment in Software Engineering

- Risk assessment in Software Engineering is the process of identifying, analyzing, and prioritizing risks that could potentially affect the success of a software project.
- It involves evaluating the likelihood and impact of various risks, such as technical challenges, project management issues, and external factors, to determine how they could impede project objectives.
- The goal is to develop strategies to manage or mitigate these risks effectively.

The Main Steps in Risk Assessment in SE

- Risk Identification: Spotting potential risks that could negatively influence the project.
- Risk Analysis: Evaluating the risks to understand their nature, causes, and potential consequences.
- Risk Prioritization: Ranking the risks based on their likelihood and impact to focus on the most critical ones.
- Risk Planning: Developing plans to avoid, transfer, mitigate, or accept risks.
- Risk Monitoring: Continuously tracking identified risks and new risks that may emerge during the project lifecycle.

Common Vulnerability Scoring System (CVSS)

- The Common Vulnerability Scoring System (CVSS) is a framework for rating the severity of security vulnerabilities in software.
- It provides a standardized way to capture the principal characteristics of a vulnerability and produce a numerical score reflecting its severity, which can then be translated into a qualitative representation (such as low, medium, high, and critical) to help organizations properly assess and prioritize their vulnerability management processes.

Scoring CVSS

CVSS scores are determined based on three metric groups:

- Base Metrics: These represent the intrinsic qualities of a vulnerability that are constant over time and across user environments.
- Temporal Metrics: These reflect the characteristics of a vulnerability that may change over time but not among user environments.
- Environmental Metrics: These are customized to reflect the impact of the vulnerability on a particular user's environment.
- The CVSS score ranges from 0 to 10, with 10 being the most severe.

Example: CVSS V2.10 Scoring

- The Common Vulnerability Scoring System consists of:
  - 6 base metrics (access vector, complexity, confidentiality impact, …)
  - 3 temporal metrics (exploitability, remediation, …)
  - 5 environmental metrics; all qualitative ratings (collateral damage, …)
- BaseScore = round_to_1_decimal(((0.6*Impact)+(0.4*Exploitability)–1.5)*f(Impact))
- Impact = 10.41*(1-(1-ConfImpact)*(1-IntegImpact)*(1-AvailImpact))
- Exploitability = 20 * AccessVector * AccessComplexity * Authentication
- f(Impact) = 0 if Impact=0, 1.176 otherwise

https://nvd.nist.gov/vuln-metrics/cvss

https://www.first.org/cvss/v2/guide

Example: DO-178b Aviation Failure Impact Categories

- *No effect* – failure has no impact on safety, aircraft operation, or crew workload
- *Minor* – failure is noticeable, causing passenger inconvenience or flight plan change
- *Major* – failure is significant, causing passenger discomfort and slight workload increase
- *Hazardous* – high workload, serious or fatal injuries
- *Catastrophic* – loss of critical function to safely fly and land

Fault Tree Analysis

- Fault Tree Analysis (FTA) is a type of failure analysis in which an undesired state of a system is examined.
- This analysis method is mainly used in safety engineering and reliability engineering to understand how systems can fail, to identify the best ways to reduce risk, and to determine the probability of a failure event.
- FTA is used in various industries, such as aerospace, nuclear power, chemical, pharmaceutical, and software.

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

Fault Tree Analysis (Cont'd)

FTA is a graphical tool that uses symbols and logic gates to represent the causes and effects of system failures.

The top event is the undesired state or failure of the system, and the basic events are the lowest-level failures or faults that can occur.

The logic gates show how the basic events combine to cause higher-level events until the top event is reached.

FTA can be used to perform qualitative and quantitative analysis of system failures, such as identifying the minimal cut sets, calculating the importance measures, and performing sensitivity analysis.

https://sixsigmastudyguide.com/fault-tree-analysis/

Uses of Fault Tree Analysis (Cont’d)

FTA can help to improve the reliability and safety of systems by providing a clear and structured way to identify and eliminate potential failure modes.

FTA can also help to design and optimize systems by evaluating different scenarios and alternatives.

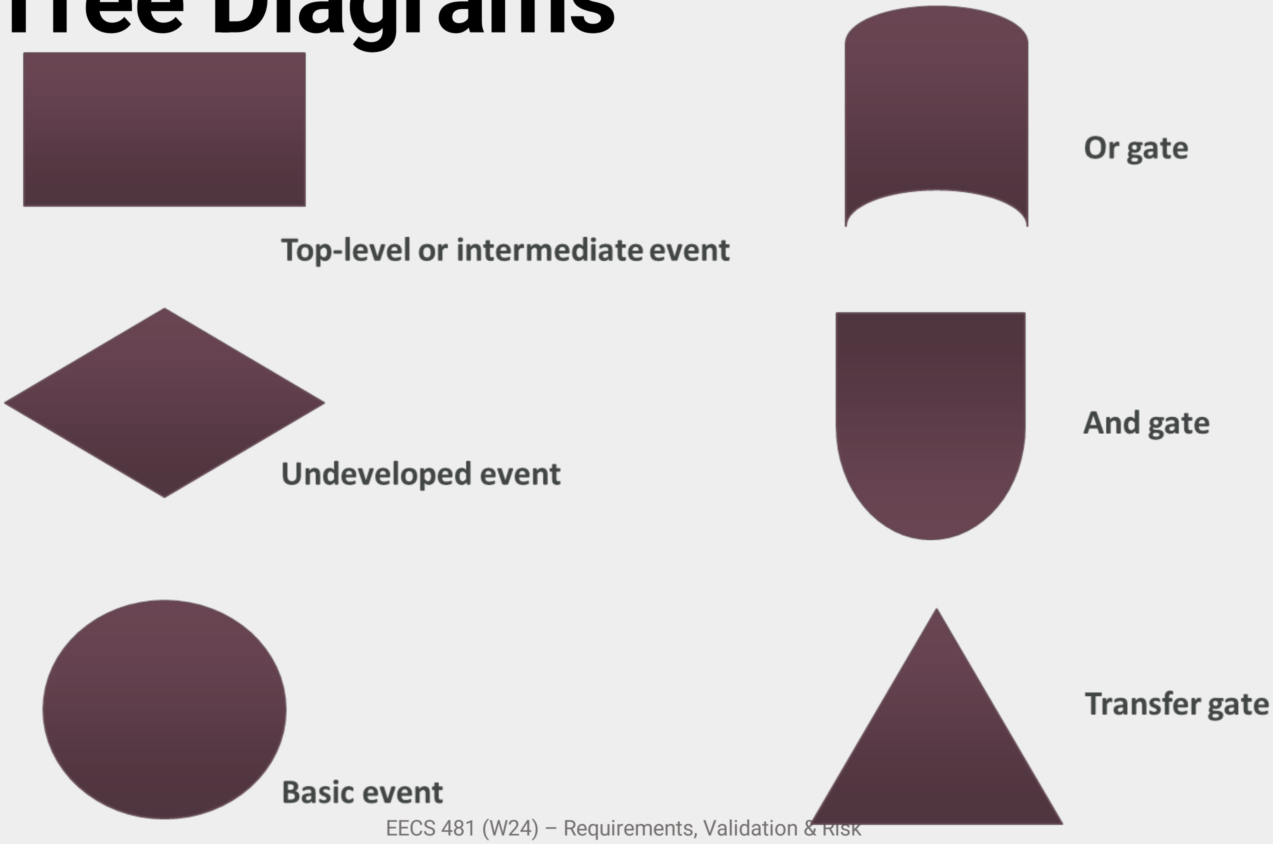
FTA can be used alone or in combination with other methods, such as Failure Mode and Effects Analysis (FMEA) or Reliability Block Diagram (RBD).

https://fiixsoftware.com/glossary/fault-tree-analysis/

Fault Tree Analysis (Cont'd)

- **Fault tree analysis** is a top-down technique to model, reason about, and analyze risk
- A fault tree analysis decomposes a particular type of failure into constituent potential causes and probabilities
- It defines the scope of system responsibilities and identifies unacceptable risk conditions that should be mitigated

Fault Tree Diagrams



Top-level or intermediate event

Undeveloped event

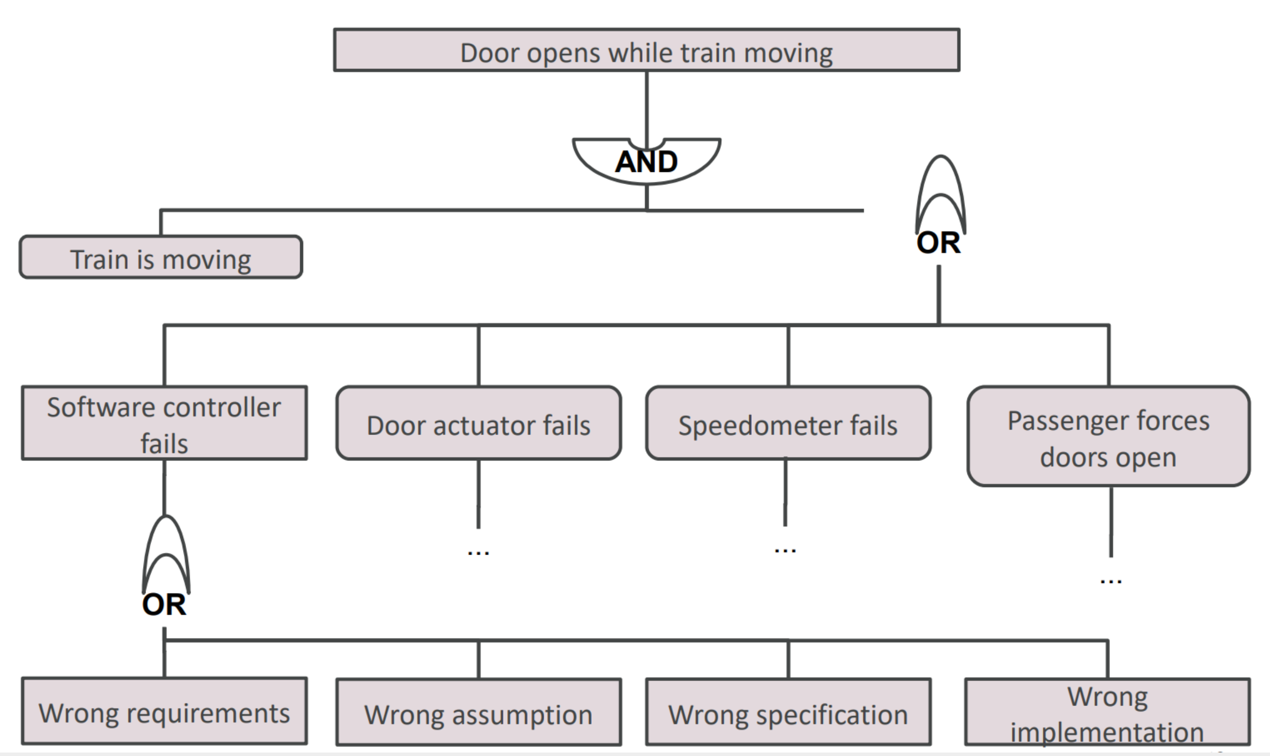
Basic event

Or gate

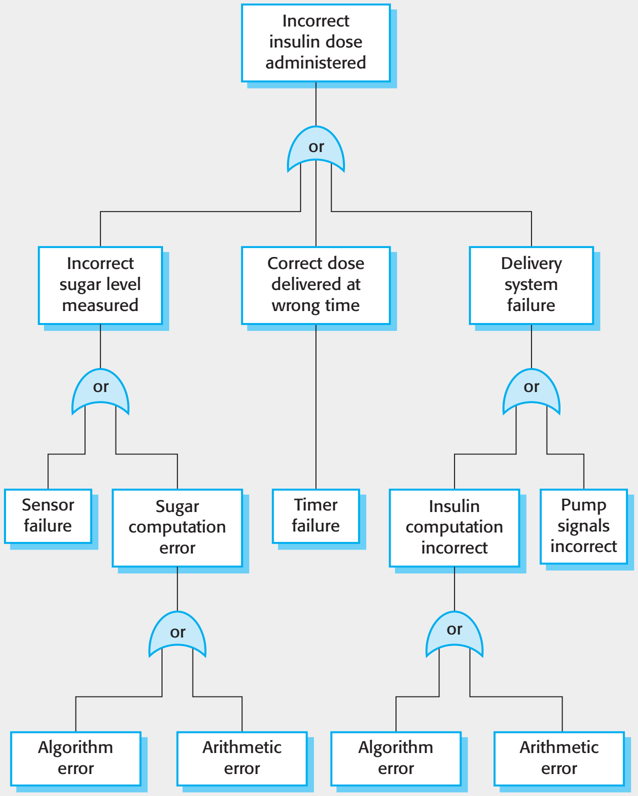
And gate

Transfer gate

Example Fault Tree to Quantify Risk



An example of a software fault tree



Fault tree analysis

- Three possible conditions can lead to the delivery of incorrect doses of insulin
  - Incorrect measurement of blood sugar level
  - Failure of the delivery system
  - The dose was delivered at the wrong time
- By analysis of the fault tree, the root causes of these hazards related to software are:
  - Algorithm error
  - Arithmetic error

Risk reduction

- The aim of this process is to identify dependability requirements that specify how the risks should be managed and ensure that accidents/incidents do not arise.
- Risk reduction strategies
  - Hazard avoidance;
  - Hazard detection and removal;
  - Damage limitation

Insulin pump - software risks

- Arithmetic error
  - A computation causes the value of a variable to overflow or underflow;
  - Maybe include an exception handler for each type of arithmetic error.
- Algorithmic error
  - Compare the dose to be delivered with the previous dose or safe maximum doses. Reduce the dose if too high.

Examples of Safety Requirements

SR1: The system shall not deliver a single dose of insulin greater than a specified maximum dose for a system user.

SR2: The system shall not deliver a daily cumulative dose of insulin greater than a specified maximum daily dose for a system user.

SR3: The system shall include a hardware diagnostic facility executed at least four times per hour.

SR4: The system shall include an exception handler for all exceptions identified in Table 3.

SR5: The audible alarm shall be sounded when any hardware or software anomaly is discovered and a diagnostic message, as defined in Table 4, shall be displayed.

SR6: In the event of an alarm, insulin delivery shall be suspended until the user has reset the system and cleared the alarm.

Risk Response Strategies

- Accept the risk: for low likelihood or low impact risks, or where the cost of mitigation is too high
- Transfer the risk: push the risk outside the system boundary
- Mitigate the risk: introduce active countermeasures
  - Reduce likelihood of failure; reduce severity of impact; change *ors* to *ands*!
- Avoid the risk: redesign so that risk cannot occur

Questions?

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

I'LL DESIGN THE SYSTEM AS SOON AS YOU GIVE ME THE USER REQUIREMENTS.
BETTER YET, YOU COULD BUILD THE SYSTEM, THEN I'LL TELL YOUR BOSS THAT IT DOESN'T MEET MY NEEDS.
I DON'T MEAN TO FRIGHTEN YOU, BUT YOU'LL HAVE TO DO SOME ACTUAL WORK.
THAT'S CRAZY TALK.
www.dilbert.com
scottadams@aol.com
© 2003 United Feature Syndicate, Inc.