Module – 4 Software Engineering Foundations

Chapter - XV

Engineering Foundations

IEEE computer society

IEEE

Engineering Foundations – Content Areas

1	Empirical Methods and Experimental Techniques
2	Statistical Analysis
3	System Development
4	Engineering Design
5	Root Cause Analysis
6	Standards
7	Modeling, Simulation, and Conceptual Prototyping
8	Tool and Platform Selection (not covered)
9	Measurement
10	Theory of Measurement
11	Goal Question Metric (GQM) Paradigm

IEEE computer society

IEEE

Content Area – 1

Empirical Methods and Experimental Techniques

Empirical Methods

- Empirical methods – forming theories based on experimental data
- A Technical Review
  - is a brief, concise evaluation
  - provides measures of effectiveness
  - provides members with empirical data

Experimental Techniques

- Are experiments performed with the goal
  - to understand
  - solve a problem
  - improve a process
- Factors, levels, and replicates
  - factor is a problem to solve
  - level is one model for solving the problem
  - Replicates are a number of experiments or observations
- Variability
  - unaccounted variations in experiments

Improvements are for reduction in
- Variability
- Errors
- Cost
- Design time

Data Sources
- Retrospective
- Observational
- Designed Experiment

Experimental Techniques

- Examples from Software Engineering
  - evaluate design configurations
  - selection of design parameters
  - Programming languages
  - type of database used
  - performance of several algorithms

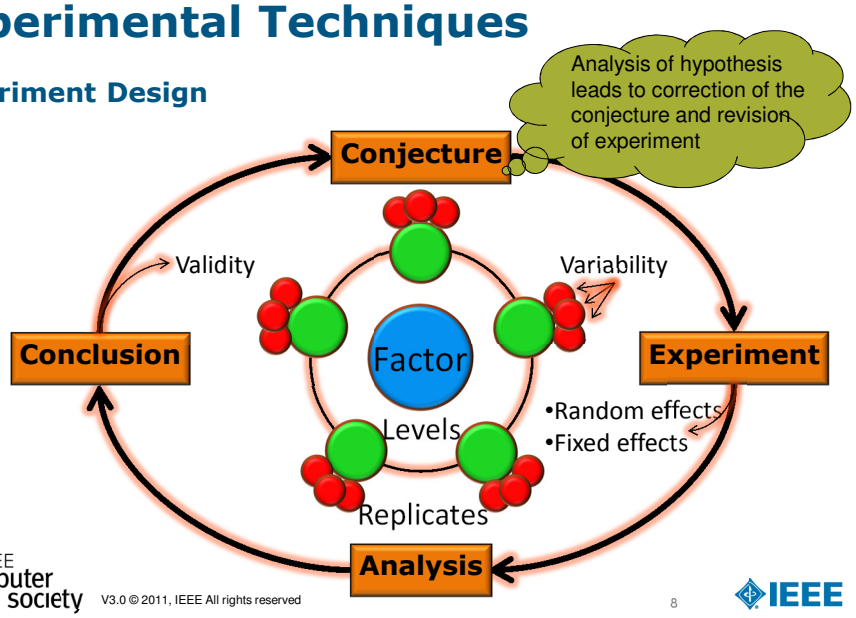
Experimental Techniques

Engineering Scientific Methods

1. Develop unambiguous problem description
2. List elements of problem/potential solution
3. Create/refine model, state limitations/ assumptions
4. Experimental techniques
5. Use model to solve problem
6. Experimental techniques to validate model and prove efficiency
7. Conclusions/Recommendations

Experimental Techniques

Experiment Design



Experimental Techniques

Analysis of Variance (ANOVA)

- ANOVA is designed to test differences between means in many sample cases
- Graphs are plotted to display or summarize the results
- Operating characteristic curves
  - Shows increasing probability of accepting the hypothesis against the number of experiment runs

Example on ANOVA

Two independent variables
- Gender of the respondent
- Misleading notion

	withfallacy	withoutfallacy
male	15	16
male	15	20
male	16	21
male	16	22
male	19	22
male	20	24
male	28	26
male	28	28
female	13	16
female	16	19
female	17	20
female	19	24
female	20	24
female	23	26
female	24	28
female	28	28

Anova: Two-Factor With Replication

SUMMARY	withfallacy	withoutfall	Total			
male						
Count	8.00	8.00	16.00			
Sum	157.00	179.00	336.00			
Average	19.63	22.38	21.00			
Variance	29.98	13.70	22.40			
female						
Count	8.00	8.00	16.00			
Sum	160.00	185.00	345.00			
Average	20.00	23.13	21.56			
Variance	23.43	19.27	22.53			
Total						
Count	16.00	16.00				
Sum	317.00	364.00				
Average	19.81	22.75				
Variance	24.96	15.53				

ANOVA

Source of Variation	*SS*	*df*	*MS*	*F*	*P-value*	*F crit*
Sample	2.53	1.00	2.53	0.12	0.73	4.20
Columns	69.03	1.00	69.03	3.20	0.08	4.20
Interaction	0.28	1.00	0.28	0.01	0.91	4.20
Within	604.63	28.00	21.59			
Total	676.47	31.00				

Assignment

- For the data provided below compute the ANOVA. Discuss the results in the class.

Content Area – 2

Statistical Analysis

Statistical Analysis Terminology

- **Statistical Analysis** is the study of collected data to solve problems, make decisions, and design products and processes
- Few of the commonly used terminologies are
  - **Random Experiment** - an experiment that can have different results even if it is repeated the same way every time
  - **Random Variable** – the function of assigning a real number to each result in a random experiment's sample space. A **discrete random variable** has a value that is finite and countable by a computer

Statistical Analysis Terminology

- **Random Sample** - subset of random variables that consists only of independent random variables
- **Population** - any predefined group of interest
- **Statistic** - an estimate or function of the observations of a random sample
- **Parameter** - an estimate or function of the observations of a population

Statistical Analysis Terminology

Mean, Variance, Standard Deviation

Let *X* be a discrete random variable and probability function *f* maps *x* *X*.

- e.g., *X* ∈ = {1,3}, 1 = 25(1/4)%, 3 = 75% (3/4);

- Mean

$$E(X) = \mu = \sum_x xf(x) = 1\left(\frac{1}{4}\right) + 3\left(\frac{3}{4}\right) = \frac{10}{4} = 2.5$$

- Variance (*It describes how far values lie from the mean*)

$$V(X) = \sigma^2 = E(X-\mu)^2 = \sum_x x^2 f(x) - \mu^2 = 1^2\left(\frac{1}{4}\right) + 3^2\left(\frac{3}{4}\right) - 2.5^2 = 0.75$$

Statistical Analysis Terminology

Mean, Variance, Standard Deviation

- Standard Deviation

$$\sigma = \sqrt{\sigma^2} = \sqrt{0.75} \approx 0.866$$

- Index of Deviation

$$IV = \frac{\text{Standard Deviation}}{\text{Mean}} = \frac{\sigma}{\mu} = \frac{0.866}{2.5} = 0.346$$

Statistical Analysis Terminology

Assignment

- A six-sided fair die can be modeled with a discrete random variable with outcomes 1 through 6, each with equal probability 1/6. Compute the mean, variance, standard deviation and index of deviation.

- Answer
  - Mean = 3.5
  - Variance = 2.92
  - Std Dev = 1.709
  - IV = 0.4882

Simple Hypothesis Testing

Some Definitions

- **Statistical inference** - is the process of drawing conclusions about a population based on a sample from that population
- **Hypothesis** - is a statement about an aspect of a system that makes some assertion
- **Hypothesis testing** - is a procedure for making decisions based on the hypothesis
- **Confidence interval** - is measured as a function of the confidence that the sample set contains the parameters being tested
- **Tolerance interval** - is the maximum and minimum variation for the constant

Simple Hypothesis Testing

Testing hypotheses

- Several methods exist to test hypotheses
  - Screening experiment - seeks to understand the process
  - Comparative experiment - same set of tests are applied to each of the states and results are compared.
  - Optimization experiment - comparative experiment in which there are multiple factors
  - Factorial experiment - determines and tests every possible
  - permutation of variables when there are several variables

Simple Hypothesis Testing

Testing hypotheses

- Experiment validation and error
  - Type I errors, or α.

    This is rejecting a true null hypothesis as false
  - Type II errors, or β.

    This is failing to reject a false null hypothesis

Discussion Question

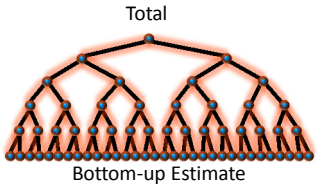
Software engineers do 100 experiments to test the hypothesis that an array (*x*) and a linked list (*y*) performing the same activity have no difference in algorithmic efficiency (*z*). There was no difference in 99 tests; in 1 test *x* was more efficient. Which of the following is true?

a) This is a comparative experiment whose research hypothesis states that *x* and *y* both lead to *z*.
b) The power of the statistical test is 1% (probability that the experiment failed to reject a false null hypothesis).
c) Type I errors or α should be set at 1%.
d) Type II errors or β should be set at 1%.

Answer: c. The results suggest that 1 time in 100 the experimenters may incorrectly reject a correct null hypothesis.

Estimation

- Estimation techniques are historical data based, experience based or intuition
  - **Statistical Estimation** - predictive value comes from historical data
  - **Bottom Up estimation** - identifies the smallest components of a project, from this bottom and rolls them up until all costs or tasks are included
  - **Estimation by Analogy** - estimation to prior similar experiences with allowances for differences
  - **Estimation by expert judgment** - intuitive method that asks one or more experts to guess the outputs



Estimation

- Estimation covers almost every aspect of software development: size, schedule, customer demand, memory usage, execution time, bandwidth use for networking, bugs, lines of code and function points, programmer-hours, and, of course, cost.
- Preparing best, worst, and fair estimates
- Point estimate is used when a mean value for a parameter cannot be known precisely

Regression Analysis

- Regression analysis is a statistical method that explores the relationship between two or more variables
  - Independent variable
  - Dependent Variable
- Linear relationship is expressed as
  - $y = mx + C$
  - $y = (\Delta y / \Delta x)\, x + C$
- C is the intercept and m is the slope, the slope of the line is
  - $m = \Delta y/\Delta x$
- The slope and intercept of this line are called *regression coefficients*

Covariance and Correlation

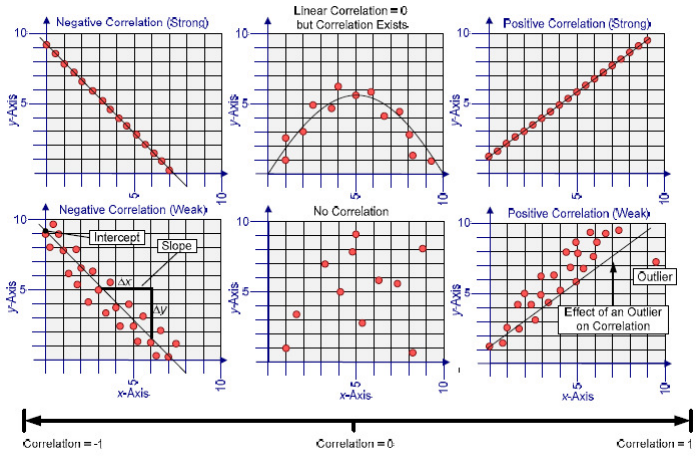
- **Covariance** – the extent to which two random variables vary together is measured by covariance. Two random variables x and y having mean E(x) and E(y), then covariance is defined as

$$\mathrm{Cov}(x,y) = E\{[x - E(x)][y - E(y)]\}$$

  - Covariance can be positive, negative or zero

- **Correlation** – is a measure of the relation between two or more variables
  - Correlation can be either positive, negative or no-correlation

Types of Correlation



Covariance and Correlation

- Covariance measures the linear relationship between two random variables.

  E.g., X= {1, 3}, Y= {2, 6}, μ_X= 2.5; μ_Y= 4.5.

  Odds: 1 = 1/4, 3 = 3/4, 2 and 6 = 1/2; so (1,2) = (1,6) = 1/8;
  (3,2) = (3,6) = 3/8.

$$
\begin{aligned}
Cov(X,Y) = \sigma_{XY} &= E\left[(X-\mu_X)(Y-\mu_Y)\right] \\
&= E(XY) - \mu_X\mu_Y \\
&= (1-2.5)(2-4.5)(0.125) \\
&+ (1-2.5)(6-4.5)(0.125) \\
&+ (3-2.5)(2-4.5)(0.375) \\
&+ (3-2.5)(6-4.5)(0.375) \\
&= 0
\end{aligned}
$$

Covariance and Correlation

- Correlation scales the covariance by the standard deviation of each variable meaning $-1 \le \rho \le 1$.

$$\rho_{XY} = \frac{Cov(X,Y)}{\sqrt{V(X)V(Y)}} = \frac{\sigma_{XY}}{\sigma_X\sigma_Y}$$

Assignment

- Compute the Covariance and Correlation in the following example
  - X={4,3} and Y=(2,6)
  - $\mu_x = 2.5$
  - $\mu_y = 1.5$
  - Odds 4 is ½, 3 is ¾, 6 is ½ and 2 is ¾

Content Area – 3

System Development

System Development

- A *system* is a group of related elements that work together to accomplish a specific objective
- A system can consist of both hardware and / or software.
- *Software system development* is application of software engineering skills to transform stated and unstated (but necessary) requirements into a design

Important Considerations

- Critical Software Systems
- Security systems
- Safety
- Performance
- Effect of Scaling
- Feature Interaction
- Hazards

Critical Software Systems

- Systems that can create physical or economic loss for its operators and users, environment damage, death in some cases
  - **Business-critical systems**
    - Cause lasting economic harm to an organization
  - **Mission-critical systems**
    - Required for success of a mission like aircraft sub-system

Critical Software Systems

- **Safety-critical systems**
  - Failure to respond when required, inadvertent response to stimuli, out of sequence response (rail road signal switching software, nuclear reactor control software)
  - Mitigate result of an accident
  - Recover from the result of an accident
  - Improper retrieval of data – patient records and images

Dependability

- Dependability is comprised of the following:
  - **Reliability:** The software is able to perform services, as specified
    - Ability to withdraw money from an ATM machine
  - **Availability:** The software is able to be used when it needs to be used
    - A Computer Tomography machine
  - **Security:** The software can prevent deliberate or unintentional intrusion
    - Deny credit card usage to invalid users
  - **Safety:** The software can avoid catastrophic failures
    - Switches off the power distribution system when it senses smoke in the vicinity

Implementing dependability
- Repair the system quickly
- Tolerate failures
- Resist and recover from attacks
- Implement Fault tolerance
- Graceful feature degradation
- Error tolerance

Security and Safety

- **Security** - Protecting system from internal and external attacks
  - Confidentiality
  - Authentication
  - Nonrepudiation
  - Integrity
- **Safety** - designed-in to ensure that the system is doing the right things at the right time
  - Anticipate system behavior in every possible critical situation.
  - Expect surprises.
  - Take account of the intended environment.

Security and Safety Analyses

- Software engineers should conduct security and safety analyses including:
- System threat or hazard analysis (predictive)
  - Failure conditions
  - Effects
  - Classification

Security and Safety Analyses

During system design, the following are checked-

- System threat or hazard analysis
- Safety requirements
- Architectural strategies
- System safety modeling
- Software hazard identification
- Hardware hazard identification
- Failure analysis (forensic)

Security and Safety Analyses

A **hazard** is: A source of potential harm or a situation with a potential for harm in terms of human injury, damage to health, property, or the environment, or some combination of these. (***IEEE Std. 1012-2004***)

- Hazard and operability study (HAZOP)
  - Qualitative and systematic
  - Best for processes
  - Used to find the hazards in a design
- Hazards analysis (HAZAN)
  - Quantitative and prioritized
- Hazard vs. failures
  - Hazard analysis (predictive)
  - Failure analysis (forensic)

Security and Safety Analyses

- Modeling for safety
- Errors
  - *Errors of omission* – due to invalid assumptions or failure to consider potential hazards
  - *Errors of commission* – due to deliberate or accidental miscalculation, faulty reasoning or flawed execution
- Type of Hazards
  - Known hazards
  - Unknown hazards

Other points of interest

- Performance – How well the system does the job it is supposed to do, may be dependable but yet performance may be poor
- Action plan is created after analysis of all non-performance analysis are carried out and causes of deviation identified
- Effect of scaling – If more than designed number of users access the system, performance will go down if not taken care of during design
- Feature integration and cross component data transfer – needs to be smooth
- Usage in an environment the system is not designed for will impact performance

Discussion Question

You have been given the responsibility to come up with the system development of a very large safety critical system that has to be used across multiple locations and multiple concurrent users. In your opinion which are the most important parameters that must be kept in mind?

a) Performance and Hazards
b) Dependability
c) Security and Safety considerations
d) Features to be developed

1. Only a
2. Only a and b
3. a, b and c
4. Only d

Answer: 3

Content Area – 4

Engineering Design

Design Process

- As applied to Systems Engineering
  - Projects need to be viewed as a whole
  - Following problem solving skills are critical
    - Problem definition
    - Solution analysis
  - Integrated process – development and support cycle are integrated. Using this knowledge, in case a project is not viable and would not produce positive results, can be cancelled early
  - Team effort – it is every ones job with a common vision - given the constraint of time and money, everyone needs to be working towards a common objective

Design Process

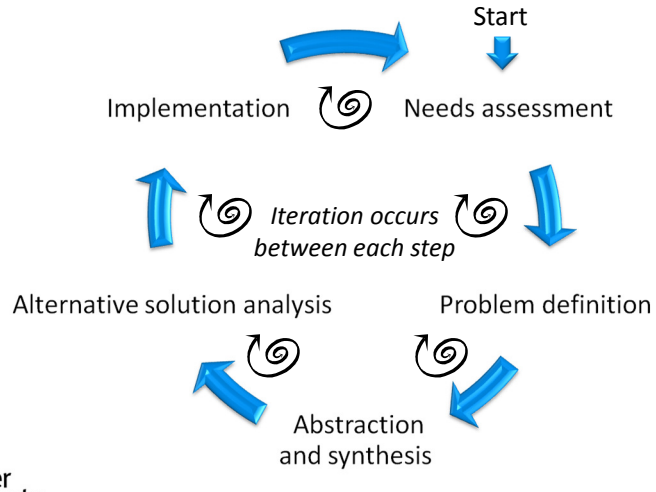
Rigorous Process

- Software Engineers should
  - write down every problem found
  - Known problems which were not documented would get ignored
  - All requirements should have problems formally documented

Design Process is rigorous

- Risks need to be handled effectively
  - *Acceptable risks* are those that are documented and there is a planned response for the event
  - *Unacceptable risks* – takes project by surprise
    - Not previously documented
    - Or risks that were not considered and analyzed

Engineering Design Process



Problem Definition, Alternative Solutions and Feasibility

"*The mere formation of a problem is far more often essential than its solution which may be merely a matter of mathematical or experimental skill*." –Einstein

An engineer must apply proper engineering principles to understand the problem and list all potential solutions before attempting to come to a final solution to a problem

Problem Definition, Alternative Solutions, and Feasibility

- Aspects of Problem Definition
  - Determining if the actual problem
  - Researching the problem
  - Recognizing when a problem was defined
  - Evaluating the reliability of sources
  - Identifying required resources
  - Finding the boundaries and constraints
  - Setting priorities and reviewing the list

Problem definition techniques

- Statement-restatement technique. To distinguish *real* from *stated*:
  - Seek the real problem
  - Seek real constraints
  - Seek the real goals
  - Seek the real links

  The problem is then re-stated

- Revision technique
  - Goal is to improve existing product

Other Aspects

- Alternative solutions
  - Critical design goal focus
  - Efficient
  - Unbiased criteria
  - Documented

- Feasibility
  - Design constraints
    - Quality
    - Time to market
    - safety
  - Implementation constraints
    - Time
    - Cost
    - ROI

Other Aspects

- If a solution violates a constraint, it should
  - Either be discarded
  - Or constraint needs to be re-evaluated
  - A feasible design should satisfy all critical design goals
- If no realistic solution is possible, then the requirements may need to be relooked at as there may be unnecessary requirements
- Trade off among requirements need to be done so that both goals and constraints are addressed

Content Area – 5

Root Cause Analysis

Root Cause Analysis (RCA)

- This is a forensic method used by engineers to detect why a particular problem has occurred and the number of dependencies could be many. Examples include but not limited to
  - Human Error
  - Configuration error
  - Improper data set
  - Incorrect inputs for the system

Root Cause Analysis (RCA)

- The advantage of the root cause analysis is that this is a general approach and can be applied to different types of problems
  - Schedule delays and customer complaints
  - Too many defects in the software and its documentation
  - Improper effort estimation
  - Under utilization of resources

Variability

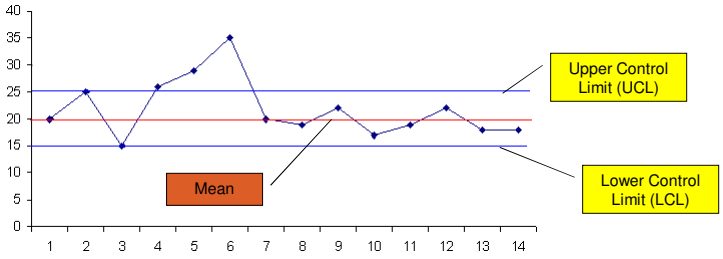
- Repeated measurements of a process output can produce slight variation in the results irrespective of all conditions being maintained same. This variability can be because of
  - Chance causes
  - Variable causes
- Some degree of variability is acceptable but any higher degree calls for conducting a root cause analysis

Some techniques for doing RCA

- Shewhart charts
- Pareto Charts
- Ishikawa cause-and-effect (fishbone) diagram
- Fault Tree Analysis (FTA)
- Failure Mode and Effect Analysis (FMEA)

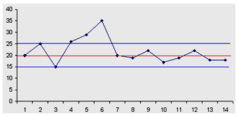
Statistical Process Control

- Makes use of control charts also known as Shewhart charts which shows if the SW development process is in a state of statistical control



- x-axis is time and y-axis represents data samples
- Distance between mean and UCL or LCL is 3σ

Interpretation of Shewhart charts



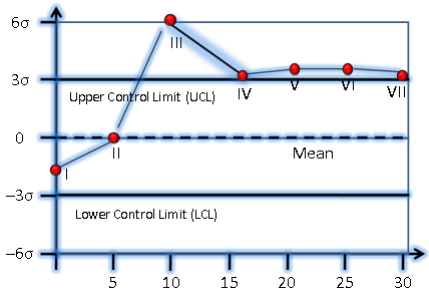
- Data points within the UCL and LCL range are considered to be in control and caused by chance causes
- Outliers (data points falling above the UCL or below the LCL) are considered to be out of control and caused by assignable causes
- If large number of data points lie above or below the control limits (UCL or LCL) the process is said to be random and out of control
- The goal is to early detect out-of-control states quickly and analyse points outside the control limits
- Points that lie beyond the control limits call for doing a RCA

Interpretation of Shewhart charts

- **Advantages of Control Charts**
  - shows the health of the process at any point in time
  - identifies performance improvement opportunities
  - distinguish between inherent variations and variations caused by sources

Discussion Question

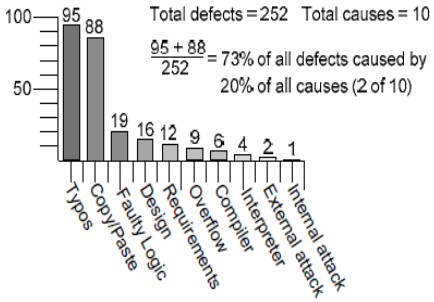
Which of the following points may indicate a nonrandom out-of-control state?



Answer: IV, V, VI, and VII. Also, III is out of control and caused by assignable causes. I and II are in control and caused by chance causes.

Pareto Charts

- Joseph Juran, who theorized that about 80 percent of a system's problems are created by about 20 percent of the causes
- Pareto chart is a bar chart that ranks issues from most frequent to least frequent



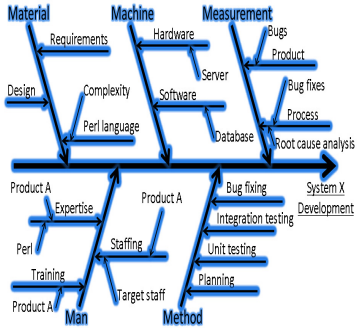
Pareto Chart for Causes of Code Errors

Benefits of Pareto Charts

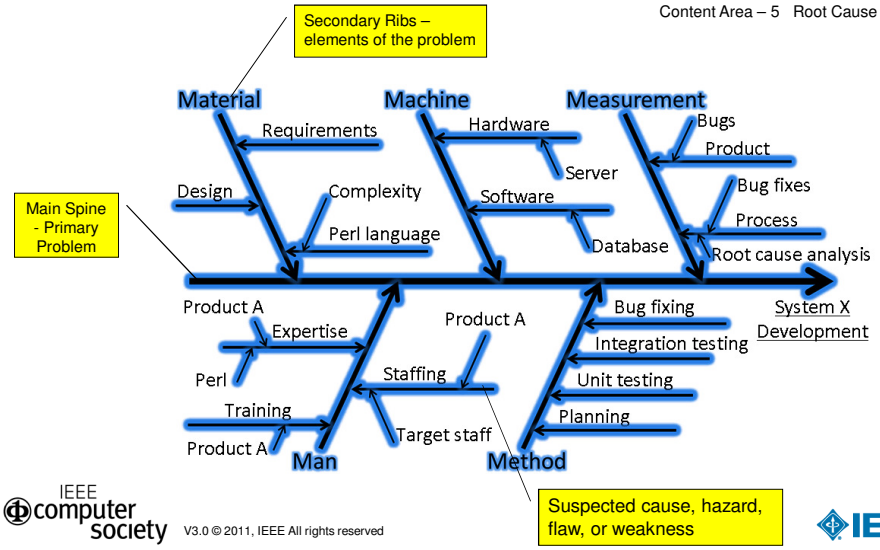
- Can reveal patterns in data that may otherwise escape notice
- Identify problems to analyze
- Assist in the design of new experiments
- Visual comparison of actual vs. budget
- Cumulative effect of multiple problems
- Can be weighted for total cost, effect on schedule, criticality, etc.

Ishikawa cause and-effect (fishbone) diagram

- The primary spine is the main cause or problem
- The secondary ribs are main components of the system or elements of the problem.
- Each secondary rib can have as many tertiary ribs as needed to list every known cause, hazard, flaw or weakness.

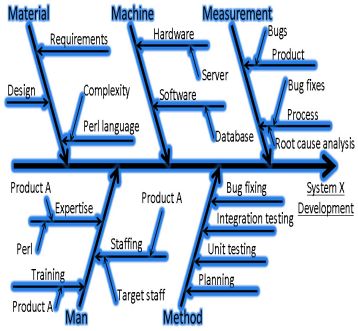


Example of Ishikawa diagram



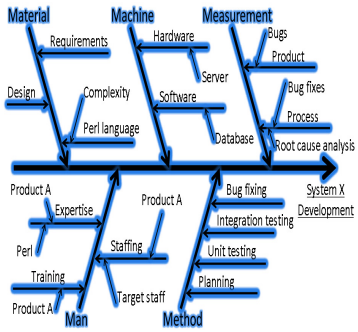
Ishikawa cause and-effect (fishbone) diagram

- A suggestion the secondary ribs made by Ishikawa
  - Man
  - Machine
  - Material
  - Method
  - Measurement



Ishikawa cause and-effect (fishbone) diagram

- Primary purpose is to connect causes to effects
- Primary benefit is its ability to allow factors to be sorted and related
- The diagram can separate the causes from symptoms so that the causes can be analyzed and not the symptoms



Benefits of Ishikawa Diagram

- An Ishikawa diagram can:
  - Be learnt quickly
  - Sort ideas into categories quickly
  - Encourage wide participation across a team
  - Assist with systems thinking or evaluating the relationship of a process to its related processes and causes
  - Avoids too much complexity
  - Yield a theory about a cause to a problem without providing a solution

Fault Tree Analysis

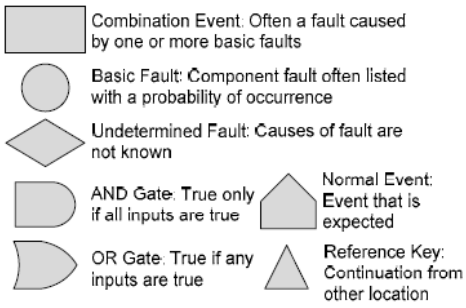
- It is a top-down approach
- A fault tree is used to analyze a single fault event
- Creates a hypothesis of the undesired event and then uses a tree to map out the underlying events
- Symbols used for FTA –
  - OR if many inputs can be true
  - AND if all inputs need to be true.

Fault Tree Analysis

FTA is a five step Process

1. Define the undesired event to study
2. Obtain the understanding of the system
3. Construct the Fault Tree
4. Evaluate the Fault Tree
5. Control the hazard / undesired event

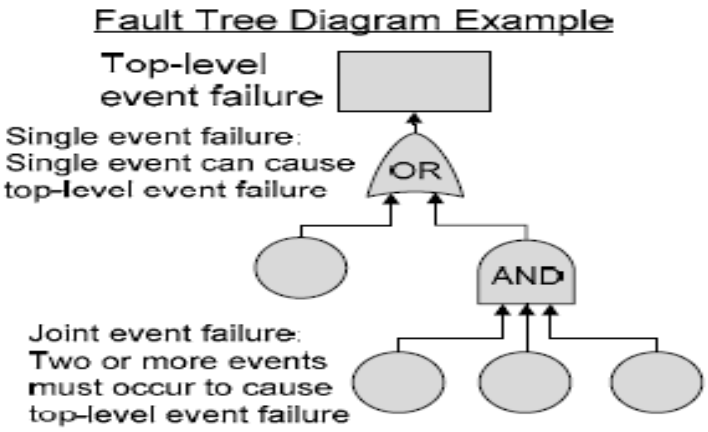
Common Fault Tree Symbols



Fault Tree Diagram

- Benefits of FTA
  - Visually depict the interactions of causes
  - Give an idea of the probability of events
  - Highlight changes to system design
- Why-Why diagram
  - Simplified version of FTA
- Dependency Diagram
  - Equivalent to success tree analysis
- Reliability Block Diagram
  - contributes to the success or failure of a complex system

Fault Tree Diagram



Failure Mode and Effect Analysis (FMEA)

- This is a step-by-step approach for identifying all possible failures in design, manufacturing , assembly process, product or service
- Bottom-up analysis that seeks to correct errors at this level

Failure Mode and Effect Analysis (FMEA)

Goals of FMEA

- Highlight areas of failure and determine their effect on the system
- Discover how to eliminate or reduce the probability and severity of a failure.
- Locate areas of critical failure
- Locate areas that affect safety or total system integrity
- Maintain rigorous documentation on failures and responses

Severity and Probability

- Severity, Probability and Detectability are rated between 1 and 10
  - Severity (S) - damage that may be caused to the system or its users. Rating of 10 is considered Hazardous and rating of 1 is considered as none.
  - Probability (P) - Probability is frequency of occurrence. Probability rating of 10 means failure is inevitable where as rating of 1 means failure is unlikely or remote.

Detectability and RPN

- Detectability (D) - controls can detect and eliminate the failure of 10 is considered as absolutely uncertain where a rating of 1 is considered as almost certain
- Risk Priority Number (RPN) = S * P * D

Contents of a FMEA Analysis

Failure Mode and Effects Analysis

Process/System/Product Name								Prepared by		
Sub Process/Subsystem name								FMEA Date		
Project Manager Name								Revision No.		
Item/Function Name	Potential Failure mode(s)	Potential Effect(s) of Failure	Severity (S)	Potential Cause(s)	Probability of occurrence (O)	Detection (D)	Risk Priority Number (S*O*D)	Recommended action(s)	Responsibility	Target date for completion

Obj. #	Object Name	Failure Mode	Failure Cause	ID Method	Failure Response	Effect	RPN
234	Shopping Cart	Out of stock; backorder not triggered	Obj. 432 interface fault	DBMS exception list	None	Customer billed for goods not sent	26
546	Credit Card Processor	Verification deadlock	Circular wait with Obj. 593	Exception 546	Log card number and retry	Delay in receiving funds	18

Benefits of FMEA

- Failure mode and effect analysis can:
  - Document areas needing special controls.
  - Highlight areas for improvement.
  - Reduce failure rates and improve safety
- Types of FMEA
  - DFMEA – focuses on Design
  - PFMEA – focuses on process

Discussion Question

A client for a software system has a primary goal of very high dependability. Which of the following root cause analyses would be best if there are about 100 items outstanding on the bug list that could affect dependability?

a) Failure mode and effect analysis (FMEA)
b) Fault tree analysis (FTA) with a Pareto chart
c) A Why-Why diagram
d) Ishikawa diagram

Answer: a

Content Area – 6

Standards

Identifying Standards

- *ISO/IEC Std. 24765* states that
  - *Standards* are "mandatory requirements employed and enforced to prescribe a disciplined uniform approach to software development, that is, mandatory conventions and practices are in fact standards"
  - Software engineering standards are normative documents

Identifying Standards

- IEEE standards can either be informative or normative
  - Normative is prescriptive in nature and tells what the software engineer should do
  - Normative documents are validated by consensus among practitioners
- Examples – ISO, IEEE, CMM, HL7, DICOM, TCP/IP, IPV6…

Definition of a standard

1. An object or measure of comparison that defines or represents the magnitude of a unit
2. A characterization that establishes allowable tolerances or constraints for categories of items
3. A degree or level of required excellence or attainment.

Standards are definitional in nature, established either to further understanding and interaction, or to acknowledge observed (or desired norms) of exhibited characteristics or behavior.

Principles and benefits of standards

Principle	Benefits
Consensus and open participation in standard setting:	• Leads to majority agreement and better chances of widespread use. • Reflects the needs of both creators and consumers. • Provides stability of design and interoperability among competitors. • Makes a competitor's design less likely to become a de facto standard. • Validates standards by consensus wisdom when no scientific proof of validity is available. (For example, *IEEE Std. 1061* lists early assessment tools.) • Allows best practices to be truly stated as such (for example, ISO 9001).
Formal processes for setting standards:	• Allow all interested parties to be treated the same. • Mean standards are slow to form but are likely to have high quality.

Principles and benefits of standards … contd

Principle	Benefits
Segregation of technical and managerial approval:	• Means that technical content is reviewed by technical experts. • Means that management adheres to formal processes.

A professional discipline that is voluntary:	• Improves products. A standard gives its name to products that conform to it and distinguish it from products that follow no standards.
	• Engenders organizations that provide certification via an external assessment for:
	  ◦ Protecting buyers: Customers know quality assurance is present.
	  ◦ Protecting sellers: Proof of standards can be a legal defense.
	• Can state minimum requirements that a design should meet.
	• Speeds communications and agreement reaching among customers and suppliers when using process standards such as *IEEE/EIA Std. 12207*.

Making the most out of standards

- Pre – project
- For standards to be useful, they need to be
  - Selected.
  - Communicated.
  - Applied.
  - Reviewed for actual use and usefulness.

- Post – project review
- The extent to which standards were applied.
- Whether or not reported compliance matches verifiable data.
- If the standards resulted in a measurable increase in quality or productivity.

Standards Quality

- Based on reliable empirical data and analysis, not just intuition
- Measureable to show whether the system or organization is in compliance
- Written using quantitative terms or qualitative terms that are defined clearly
- Clear about the effort required for success and have clear success criteria
- Able to provide a vocabulary for buyer and seller communications
- Supported by a baseline study and followed by case studies showing progress

Objectivity within Standards

- From completely effective to less effective
  - **Completely objective:** "…shall be done so that costs result in a positive return on investment."
    - can be measured without bias by calculating a ratio
  - **Partially objective:** "…shall be done within reasonably budgeted cost constraints."
    - depends on how "reasonably" is defined
    - whether the budget was accurate to begin with
  - **Subjective:** "…shall be done efficiently."
    - depends on the definition of the term "efficiently"
  - **Declarative:** "…shall be done."
    - no method of being measured

S2ESC organization of IEEE standards

S2ESC: Software and System Engineering Standards Committee
Standards are organized in several ways to make them user-friendly

Topic	Level of prescription	Objects of software engineering
Documentation	Terminology	A project (agent)
Life-cycle processes	Collection guides	uses resources
Measurement and reliability	Principle standards	that help perform processes
Plans	Element standards	to create a product
Project management	Application standards	by interacting with a customer
Reuse	Technique standards	
Terminology		
Tools		

Adopting Standards

- Standards should be adapted to meet the organization's needs and goals
- *IEEE/EIA Std. 12207 contains guidelines for adopting standards*
  - Identifying project environment
  - Soliciting inputs
  - Selecting processes, activities, and tasks
  - Documenting tailoring decisions and rationale
- Tailoring of standards to suit organizations
  - Standards and processes should not become an overhead for projects
  - Use only that which is necessary

Discussion Question

Match the examples of requirements within standards to the applicable labels:

(a) Declarative

(b) Completely objective

(c)Partially objective

(d) Subjective

1: "…shall minimize faults so any remaining faults rarely cause critical failures."

2: "…shall minimize faults."

3: "…shall minimize faults so that any remaining faults shall result in 0 critical failures."

4: "…shall minimize faults to safe levels."

Answer: a-2, b-3, c-1, d-4

Content Area – 7

Modeling, Simulation and Conceptual Prototyping

Introduction

- Models and prototypes can be created to conduct
  - alternatives analyses
  - reduce implementation risks
  - usually have a research
  - experimentation phase

Introduction

- Benefits Include
  - Finding realistic boundaries or constraints
  - Explaining systems to others
  - Assessing simplifications made
  - Embedding empirical data
  - Comparing two alternative tools
  - comparing the method to preexisting models, simulations, prototypes

Modeling

- Not every idea can or should be implemented in engineering
- Modeling fills in this role for visualization and communication
- Proper level of abstraction is a must
  - Good models should reasonably represent the actual system
  - With minimal complexity
    - model is only as complex as needed to approximate the particular problem at hand

Modeling

- All models are approximations
  - A model with too little information may miss critical interdependencies and hence may not represent actual system
    - Either extreme may be a waste of time
    - Only that amount of modeling is required that would bridge the role

Modeling

- Proper level of abstraction
  - Models show a problem in abstract
  - Occam's razor: neither too much nor too little detail
- Avoid common model errors
  - Does the chosen model help solve the problem?
  - Does the model describe the system or process?

System models

System Models focus on
- Structure
- External Environment
- Behaviour

Deterministic system models always follow an expected pattern and produce an expected result

Stochastic (prediction-based) system models are models that predict system behavior with a degree of uncertainty related to the complexity of the system

Process models

Prescriptive process models show how a process should be done

Descriptive process models describe how a process is actually performed and its results, whether successful or unsuccessful

Simulation

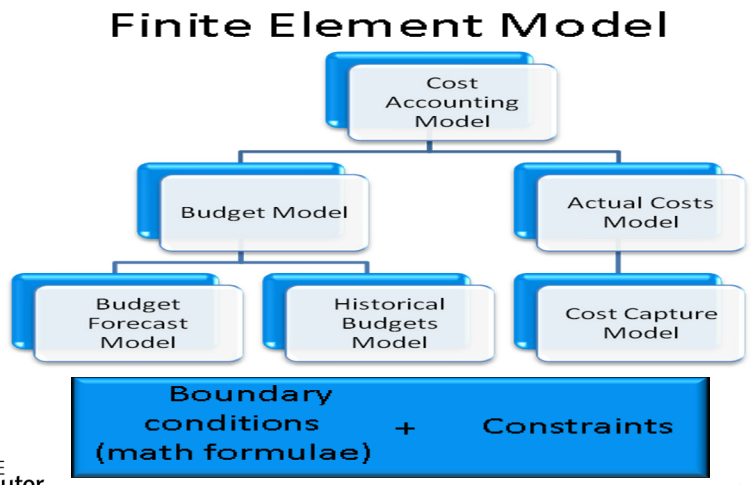
- The term *simulation* refers to:
  - A model that behaves or operates like a given system when provided a set of controlled inputs
  - The process of developing or using a model, as in the above (*ISO/IEC Std. 24765*)
  - The use of a data processing system to represent selected behavioral characteristics of a physical or abstract system (*ISO/IEC Std. 2382-1:1993*)

Finite Element Model

Approach

1. Takes a large system and uses decomposition to break It down into sub-systems
2. Decomposition must not be too broad or tediously detailed
3. Subsystem is programmed in the simulation using mathematical equations
4. Reflect inputs and outputs or predicted behavior
5. Assembles components to determine how entire system interacts

Finite Element Model

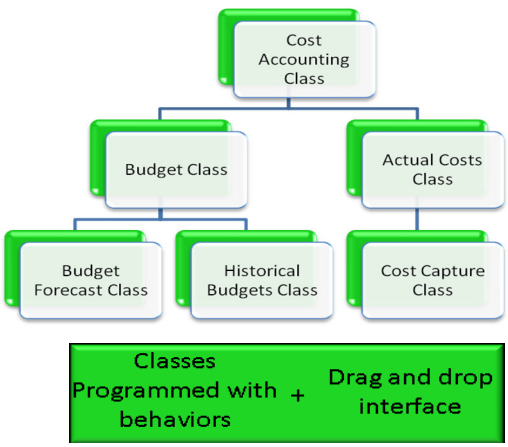


Process Simulation

Approach

1. Breaks a problem down into sub-processes so that each element may be programmed into a computer
2. Classes are defined of particular behaviour
3. Drag and drop functionality

Process Simulation



Conceptual Prototyping

- A prototype is used to get feedback from users for improving and specifying a complex human interface, for feasibility studies, or for identifying requirements
- Definition as per (ISO/IEC Std. 24765) - A preliminary type, form, or instance of a system that serves as a model for later stages or for the final, complete version of the system

Content Area – 8

Tool and Platform Selection (Not available in the text)

Content Area – 9

Measurements

Need for Measurement

- Measurement is the key to achieving control over the software engineering process
- Develop robust data tracking capabilities
- Training in gathering data using quality methods and state-of-the-art tools is essential
- Systematic discipline must start with qualitative data

Proper balance of data gathering

- Need for organized methodology to retaining data in a database
- Measurements provide a benchmark to judge improvement
- Collecting more data than is needed might be beneficial in the long run
- Management of too much data needs to be handled carefully

Prediction based Measurement

- **Probability model**
- **Prediction interval**
- **Extrapolation**

- Using historical data in a predictive model one needs to be careful about how it is relevant

Measurement: Proper Balance of Data Gathering

- Measurement is key to control
- Implement organized methodology for retaining all projects' data in a database
  - Homogeneous
  - Data for estimates is easier to defend than intuition
- Data gathering can be thorough within constraints of cost and need
- Analysis should be restricted to information that will help guide decisions

Measurement in Software Projects

- Measurement should be considered in context of technical and project goals
- Show whether the goals will be met and, if not, by how much the results will vary
- When software standards are stated objectively, measurements are the key to applying them

Measurement as a Form of control - advantages

- Organizations that employ measurements effectively will
  - be able to win more contracts with lower bids
  - yet stay profitable because their estimates prove accurate
- In the long run they provide evidence of
  - improvement or a need to improve
  - they include broad measures such as sales and market share

Product, process, and project measurements

- Empirical data enables communication because
  - everyone must stick to the facts
  - everyone has a common frame of reference
- **Product metric** - "used to measure the characteristics of any intermediate or final product of the software development process" (*IEEE Std. 1061-1998 (R2004)*).

Product, process, and project measurements

- **Process metric** "used to measure characteristics of the methods, techniques, and tools employed in developing, implementing, and maintaining the software system" (*IEEE Std. 1061-1998 (R2004)*).
- **Project control** - "the activities concerned with monitoring the progress of a project, its direction, quality, and resource utilization, as compared with project plans" (*ISO/IEC Std. 2382-20:1990*).

Estimated versus Actual data

- Data can be estimated or actual
- Estimated data is data that cannot be known with certainty:
  - Internally generated data is preferable to external benchmarks.
  - Estimated data is most effective when several independent estimates are combined

Estimated versus Actual data

- Different persons and different estimation methods should be used
  - Many software estimation tools exist and are superior to heuristics.
- Actual data is data collected at the end of milestones and the project when estimation is no longer necessary, for example, total bugs found and repaired.

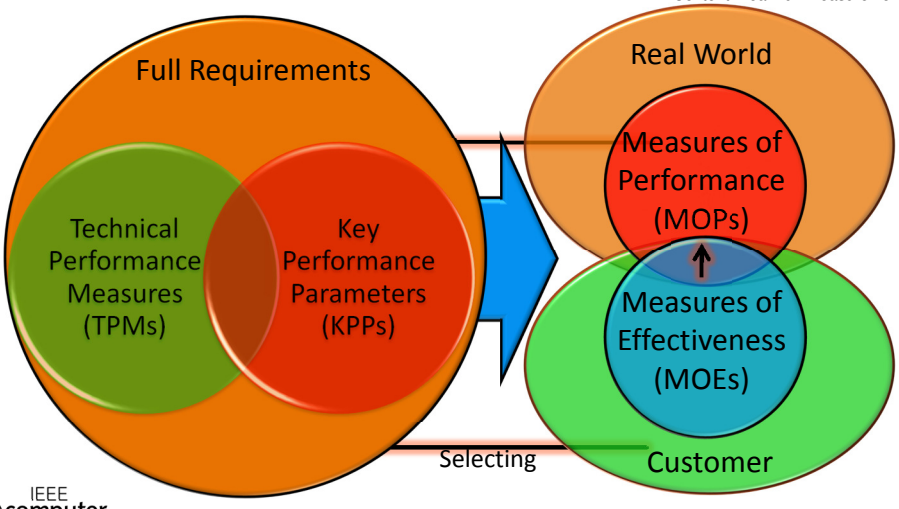
Technical Measurement

- Technical measurement is a group of interrelated measurements that relate to the design and implementation of solutions to technical problems including their risks
  - Technical Performance Measures (TPMs)
    - measure how well technical elements satisfy technical requirements

Technical Measurement

- Measures of Performance (MOPs)
  - measure functional output under actual operating conditions
- Key Performance Parameters (KPPs)
  - measure achievement of functional output requirements that are so critical
- Measures of Effectiveness (MOEs)
  - measure the fit of the solution to the problem from the customer's point of view

Technical Measurement



Measurement in software

- Important software measurements include measurements of quality, design complexity, maintainability, defect tracking, portability, and productivity
  - For example quality measurements include traceability, accuracy, completeness, consistency, and error tolerance
- The measurements chosen for a project should be tailored for the specific needs of the project. One measurement set does not fit all

How measurements are used

Software Size

- Software size is a vital measure because many other estimates are based on it, such as total cost.
- Size is measured by source lines of code (SLOC), function points, or source lines of design (SLOD).
- Extrapolating growth in actual SLOC requires increasing the scope or limiting growth.
- At milestones extrapolate total SLOC plus subcategories: new, reusable, and modified-reusable SLOC.

Software Personnel

- Estimate the number of experienced and total personnel per milestone. (The term *experienced* must be defined.)
- Determine the actual number of experienced and total personnel per milestone.
- Calculate planned versus unexpected losses of personnel.
- At milestones extrapolate the results of personnel changes against planned changes.
- Respond to potential schedule slippage due to a loss of personnel, but be wary of adding personnel to correct slippage for nonpersonnel reasons.

Schedule Estimation

- Schedule estimation is a function of SLOC, historically estimated complexity constant and labor years.
- Alternately, schedule estimation can use cost data for software and work breakdown structure (WBS):
  - Number of months in program schedule (revision at each milestone)
  - Budgeted cost of work scheduled (BCWS)
  - Budgeted cost of work performed (BCWP)

How Measurements are used …

Hardware Resource Usage (Software Expense)

- Estimated CPU, memory, I/O channel usage
- Actual CPU, memory, I/O channel usage
- Comparison indicates expense of system on the target computer and the relative resource usage.
- Most projects set the bar at 50 percent of target computer usage; excess usage requires optimization or new system requirements.

Software Volatility/Stability

- Current total number of requirements
- Number of new software action items (SAIs)
- Cumulative number of changes to requirements (additions, deletions, changes)
- Cumulative number of open SAIs
- Requirements changes (even deletions) after final design review have a significant impact on schedule.
- Measure volatility/stability by comparing total to cumulative requirements and SAIs.
- The slope of requirements changes should start steep and taper off if stable.
- Stable software should show spikes at each review that grow smaller on graphs of SAIs.

Content Area – 10

Theory of Measurements

Theory of Measurements

- Measurement is the basis of science because science is based upon observed phenomena.
- The goal of the theory of measurement is to allow a safe acquirement and reproducibility of measuring characteristics
- To draw conclusions one must take into account the nature of the correspondence between the attribute and the measurements

Theory of Measurements

- Measurement theory is a branch of applied mathematics that is useful in measurement and data analysis
- Measurement theory shows that strong assumptions are required for certain statistics to provide meaningful information about reality
- Measurement theory encourages people to think about the meaning of their data critical assessment of the assumptions behind the analysis

Measurement Terminology

Type	Measurement Term	...Equivalent Term in the Rules of Logic
Abstract terms	Concepts	...Theorems
	Definitions	...Propositions
Empirical terms	Operational definitions	...Hypotheses
	Empirical measurements	...Data analysis

Measurement Terminology

Measurement Prerequisites

- Measurements can be made on different levels or scales
- **Nominal scale** - each element must be mutually exclusive and jointly exhaustive
  - Mutually exclusive - term used for one element in the category cannot be used of another element - Example Python and ADA
  - Jointly exhaustive - anything that could fit in the category has an element, though "other" can be used

Measurement Prerequisites

- **Ordinal scale** - orders items in a nominal scale to provide distinction among the elements based upon some criteria
  - For example, 1 = Excellent, 2 = Good and so forth can show which is higher but not by how much
- **Interval scale** – based on well defined and agreed upon system of measurement. Example the metric system or time
- **Ratio scale** - is an interval scale that has a non-arbitrary zero point. Example Celsius has a arbitrary zero point

Theory of Measurement

Benefits of measurement

- Visibility for management decision making
- Focuses work on key deliverables
- Baseline for the long term
- Boosts morale by highlighting difficulties
- Sets realistic expectations

Theory of Measurement

Risks of measurement

- Reported data gains weight, often a detriment to overall project
- Data collected at too high or too low a level
- Cross project measurements not comparable
- Failure to give staff feedback
- Evaluating staff using measures outside control

Measurement Types

- Basic types of measurements include:
  - Percentage (*x*/100)
  - Ratios (one mutually exclusive element divided by another)
  - Proportions (one element divided by itself plus other members of its category)
  - Rate (Δx per unit of *y* where *y* is usually time)

Content Area – 11

Goal Question Metric (GQM) Paradigm

Goal Question Metric (GQM)

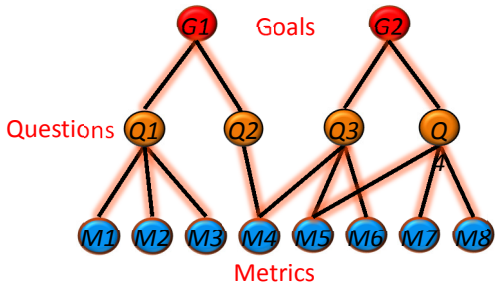
- *IEEE Std. 1061-1998* describes GQM as follows
  - "The GQM paradigm is a mechanism for defining and evaluating a set of operational goals, using measurement. It represents a systematic approach to tailoring and integrating goals with models of the software processes, products, and quality perspectives of interest, based upon the specific needs of the project and the organization."

GQM Paradigm and Goal Tree

- A top-down approach
  - start with the goals of the project or organization and works from there to select appropriate measurements, or metrics
- Goals - reflect the objectives of the organization
- Question – that assist in quantifying the goals

GQM Paradigm and Goal Tree

- Metrics – Measurements that must be gathered to answer each of the question posed to meet the goal

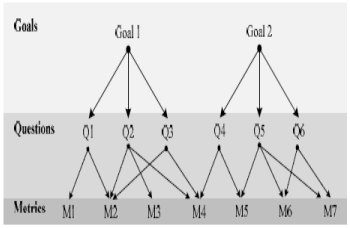


Set Goals

- Clearly defined software or process improvement goals
  - Use an active verb, such as assess or improve.
  - Define what it will do, such as improve quality, risk, productivity, efficiency, effectiveness, or customer satisfaction

Set Goals

- Goals are constructed with basic components
  - Purpose
  - What issue it will address
  - Relationship – model, process, resource
  - Impact on stake holders
  - Environment



Discussion Question on GQM

- An organization sells an application called Finance Pro that helps its customers organize their personal finances. The organization is losing market share, in part, due to a perception that the software is not reliable. Management decides to use a GQM approach to address these issues

How to proceed

Goal 1: Increase (Purpose) the reliability (Issue) of Finance Pro (Product) relative to competitors (Environment) from the point of view of the customer (Stakeholder).

- Ask Questions
  - determine how to achieve the goal
  - how the assessment can be done

How to proceed

- Questions should tie
  - the product or system
  - stakeholders
  - specific environment
  - sometimes initial goals are so broad that a set of sub-goals should be created to aid with question development

Questions

- Q1. How many errors affect the customer?
- Q2. What is the current level of customer satisfaction?
- Q3. What is the response time between a reported error and its resolution?

Questions

- Q4. What is the deviation between actual error fixing time and estimated time?
- Q5. Is the number of errors reported being reduced?
- Q6. How reliable do customers perceive competitors' products to be?

Establish Metrics

- Metrics, or measurements, are established to move from the operational level to the quantitative level
  - **Objective metrics** depend solely upon the object being measured
  - **Subjective metrics** depend on the point of view of the observer
  - Sometimes predictive quality models can be adopted for use

Metrics

- Number of valid error reports received by customer support (Q1, Q2)
- Results of customer satisfaction survey (Q2, Q6)
- Complexity and efficiency of algorithms used in Finance Pro (Q3, Q4):
  - Average lines of code
  - Average expense (processing power required)
  - Average case and worst case of processing time and memory usage
  - Average number of subroutines, and so forth

Metrics

- Time to completion of bug fixes (Q3, Q4, Q5):
  - Average effort per line of code
  - Average time to code algorithm
  - Average time to test algorithm
  - Average time devoted to bug finding and fixing/number of bugs
- Number of software action items (SAI) (Q1, Q3, Q4, Q5):
  - Number of new software action items (SAIs)
  - Number of open SAIs
  - Ratio of new to open SAIs

Evaluation

- Determine whether the potential metrics can gather the data needed answer the questions
- In the example
  - The number of customers who have used competitors' products is too small for an adequate comparison.
  - Customer service doesn't track its issue logs consistently.
  - It is difficult to determine the complexity level of competitors' algorithms.
- Measurement must be possible for the question to be answered

Discussion Question

- Suppose you are employed as a software engineer in a software organization and there is concern about the effectiveness of the software testing process being used for small software projects (5-10 KLOC). Management asks you to use the GQM (Goal-Questions-Metric) paradigm to select metrics that could be used to assess the effectiveness of the organization's testing process.
- What are some goals, questions and metrics?

Module – 4
Chapter – XV Debrief

- Content Domain XIV: Engineering Foundations

Any Questions Regarding Module ?