CSQE Exam Preparation

Douglas Hoffman
BACS, MSEE, MBA, ASQ-CSQE, ASQ-CQMgr, ASQ Fellow
Software Quality Methods, LLC.
24646 Heather Heights Place
Saratoga, California 95070-9710
doug.hoffman@acm.org
www.SoftwareQualityMethods.com
408-741-4830

Copyright Notice

These notes include examples which are context specific. In class discussions I may respond to questions and provide information on approaches and solutions to students' problems. I may answer questions in a way that I believe would "normally" be true but our answer could be completely inappropriate for your particular situation. I cannot accept any responsibility for any actions that you might take in response to my comments in this course. Courses such as this one would not be needed if the problems addressed were simple enough to analyze and resolve in a few minutes.

About Doug Hoffman

- I advocate and provide advice and services in software testing and quality assurance.
- Software quality assurance, and especially software testing, have a reputation of being where failed programmers or programmer "wanta be's" congregate. I don't believe it's true, and it's through courses like this that we can change the perception. I gravitated into quality assurance from engineering. I've been a production engineer, developer, support engineer, tester, writer, instructor, and I've managed manufacturing quality assurance, software quality assurance, technical support, software development , and documentation. Along the way I have learned a great deal about software testing and measurement. I enjoy sharing what I've learned with interested people.
- <u>Current employment</u>
  - President of Software Quality Methods, LLC. (SQM)
  - Management consultant in strategic and tactical planning for software quality
  - Adjunct Instructor for UCSC Extension, teaching Black Box Software Testing and Test Automation Design
- <u>Education</u>
  - MBA
  - MS in Electrical Engineering, (Digital Design and Information Science)
  - B.A. in Computer Science
- <u>Professional</u>
  - Past Chair, Silicon Valley Section, American Society for Quality (ASQ)
  - Founding Member and Past Chair, Santa Clara Valley Software Quality Association (SSQA, 1992-1997)
  - Certified in Software Quality Engineering (ASQ-CSQE, 1995)
  - Certified Quality Manager (ASQ-CQMgr, 2003)
  - Previously a Registered ISO 9000 Lead Auditor, (RAB 1993)
  - I also participate in the Los Altos Workshop and several others on Software Testing

About This Course

The CSQE Exam Preparation Class provides a survey of the topics included in ASQ's CSQE Body of Knowledge to familiarize experienced software quality practitioners with its terminology and basic concepts. The practices recommended and discussed in this course are useful as an overview of software quality engineering. There is not time to cover any of the topics in depth and therefore the class is not intended to teach specific job skills, techniques, or tools.

The class content is led by the slide set, but the presentation of materials is heavily influenced by the specific questions and interests of the students in each session. I also include much material from my and the students' experience which often does not correspond with the CSQE BOK or would necessarily provide the expected ("correct") answers to CSQE Exam questions.

In the slides I have tried to identify the "non-BOK materials" with a red asterisk (*) to footnote them as not specifically being part of the BOK, and I try to make it clear in the discussions when ideas are not likely to be on the exam or may possibly be contrary to expected exam responses.

The Course and The Exam

In this class I explicitly present information from the BOK I believe is likely to be included on the exam, ideas about interpreting the BOK to pass the exam, and specific test taking techniques. This is all based upon my experience and understanding of the material, the history of the CSQE BOK, ASQ certification policies and procedures, and feedback from other CSQEs. I do not have any special insider information about the BOK, exam materials, or the exam itself. Students earning their CSQE do so based upon their own knowledge and capabilities. The class does help substantially improve the chances of passing the exam by reviewing the technical material, explaining vocabulary, setting expectations, and providing techniques for maximizing scores and making the best use of the examination time.

As a result of combining information from the formal BOK with experience, students from prior classes have reported learning very practical, real world concepts immediately applicable to their work. Past students have also achieved a pass rate on the CSQE Exam more than double the overall National pass rate.

CSQE Body of Knowledge

1. General Knowledge
2. Software Quality Management
3. Software Engineering Processes
4. Program and Project Management
5. Software Metrics, Measurement, and Analytical Methods
6. Software Verification and Validation
7. Configuration Management

Percent of Time in Phase

Defect Severity Breakdown

Number of Compiles

Software Quality

Path Coverage

Lines of Documentation

Number of Procedures

Code Complexity

Defect Density

Number of Comments

Test Coverage

Software Development Life Cycle

Number of Defects

Lines of Code

Milestone Scheduled Dates

Test Effort per Defect

Fault Insertion Rates

Defect Discovery Rate

CSQE Class Schedule

(1)	Introduction; General Knowledge
(2)	Software Quality Management
(3-4)	Software Engineering Processes
(4-5)	Program and Project Management
(6)	Software Metrics, Measurement, and Analytical Methods
(7)	Software Verification and Validation
(8)	Configuration Management; Exam Strategies; Review

Introductions

- Me
- You

Class Objectives

- Survey the SQE Body of Knowledge
- Prepare for CSQE Examination
- Share quality related experiences
- Gain information on selected SQ topics
-
-

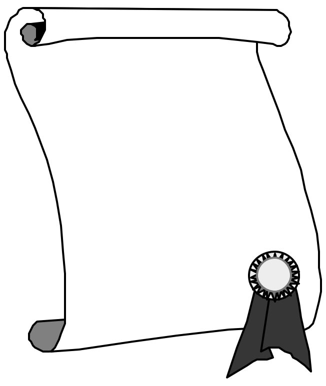
Expectations

- Mine
- Yours

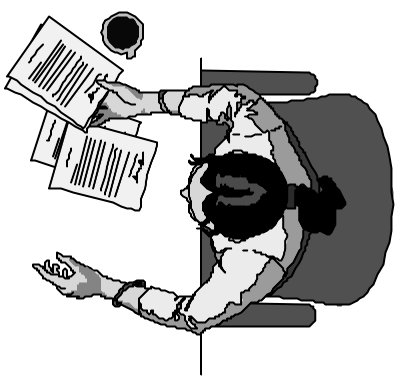
General Knowledge

- **General Knowledge, Conduct, and Ethics**
- Software Quality Management
- Software Engineering Processes
- Program and Project Management
- Software Metrics, Measurement, and Analytical Methods
- Software Verification and Validation (V&V)
- Software Configuration Management

CSQE Requirements



- Obtaining CSQE Certificate
- Maintaining Certification
- Bloom's Levels Of Cognition
- CSQE Subject Areas



Certification Requirements

- Education and/or Experience
  - 8 years in quality field
  - up to 5 years credit for degrees
- Proof of Professionalism
- Examination
  - proctored, open book exam
  - 160 questions

Recertification Requirements

- Recertify every 3 years
- 18 points needed
  - Professional Development
  - Employment
  - Instructor/Student
  - Meetings
  - Committees
  - Certifications
  - Proctoring
  - Publishing

CSQE Body of Knowledge

- General Knowledge (10% - 16 questions)
- Software Quality Management (19% - 30)
- Software Engineering Processes (16% - 26)
- Program and Project Management (15% - 24)
- Software Metrics (15% - 24)
- Verification and Validation (V&V) (15% - 24)
- Configuration Management (10% - 16)

CSQE BOK Subject Areas

- General Knowledge, Conduct, and Ethics
- Software Quality Management
- Software Engineering Processes
- Program and Project Management
- Software Metrics, Measurement, and Analytical Methods
- Software Verification and Validation (V&V)
- Software Configuration Management

Levels of Cognition[1]

- Knowledge
- Comprehension
- Application
- Analysis
- Synthesis
- Evaluation

[1] Bloom, B. S., Engelhart, M. D., Furst, E. J., Hill, W. H., & Krathwohl, D. R. (1956). Taxonomy of educational objectives handbook 1: Cognitive domain. New York: McKay.

Levels of Cognition

Level		*Verbs*
Knowledge	-	write, list, name, define, label, state
Comprehension	-	explain, describe, summarize, illustrate, paraphrase
Application	-	use, solve, apply, construct, demonstrate, compute
Analysis	-	analyze, compare, contrast, separate
Synthesis	-	create, design, invent, develop
Evaluation	-	judge, recommend, critique, justify

General Knowledge, Conduct, and Ethics

- Quality philosophy and principles
- Standards, specifications, and models
- Leadership tools and skills
- Ethical conduct and professional development

Quality Philosophies and Principles

- Benefits of software quality
- Prevention vs. Detection philosophies
- Software TQM principles and applications
- Organization and process benchmarking

Benefits of software quality

To Customers:

– Satisfaction

– Improved Reliability

– Reduced Errors in Operations

– Matching with Requirements

Benefits of software quality

To the Organization:

- Meeting Customer Requirements
- Stable Requirements
- Verification that Requirements are Met
- Consistent Application of Processes
- Improvement Over Time
- Quality of Life *

* Not part of CSQE 2002 BOK

Quality philosophies*

- Philip Crosby (Conformance to Requirements)
- Dr. W. Edwards Deming
  (Never Ending Improvement)
- Dr. Joseph Juran (Fitness for Use)

* Not part of CSQE 2002 BOK

Philip Crosby*

- Four Absolutes:
  - Quality Means Conformance to Requirements
  - Quality Comes from Prevention
  - Zero Defects
  - Quality Measurement is the Price of Nonconformance
- 14 Steps to Improvement

* Not part of CSQE 2002 BOK

Dr. W. Edwards Deming*

- Never Ending Improvement
- Fourteen Points
- Seven Deadly Diseases

* Not part of CSQE 2002 BOK

Deming's Fourteen Points*

- Constancy of Purpose for Improvement.
- Adopt the New Philosophy.
- Cease Dependence on Mass Inspection.
- Cease Doing Business on Price Tag Alone.
- Continual Improvement of Process.
- Institute Training on the Job.
- Institute Leadership.
- Drive Out Fear.
- Break Down Barriers Between Departments.
- Eliminate Slogans, Exhortations, and Targets.
- Eliminate Numerical Quotas.
- Allow Pride in Workmanship.
- Institute a Program of Self-Improvement.
- Do It.

* Not part of CSQE 2002 BOK

Deming's Seven Deadly Diseases*

- Lack of Constancy of Purpose
- Emphasis on Short-Term Profits
- Personal Evaluation Appraisal
- Mobility of Management (Job Hopping)
- Use of Visible Figure for Management
- Excessive Medical Costs
- Excessive Costs of Warranty

* Not part of CSQE 2002 BOK

Dr. Joseph Juran*

- Fitness for Use
- Revolutionary Rate of Improvement
- Top Management is in Charge
- Specific Goals for Quality Improvement in the Business Plan
- Assign Responsibility for Improvements to Individuals
- Train People
- Empower the Workforce

* Not part of CSQE 2002 BOK

Prevention vs. Detection

- Detection
  - Identify
  - Correct
- Prevention
  - Start Earlier
  - Look Upstream for Improvements

Software TQM Principles*

- Continuous Improvement
- Management by Facts
- Measurement of Progress
- Quality Teams
- Management of Resources
- Leadership

* Not part of CSQE 2002 BOK

Types of Quality Teams*

- Quality Council
- Cross-Functional Team
- Quality Action Team
- Tiger Team

* Not part of CSQE 2002 BOK

Software TQM Applications*

- Assessments
- Activities
  - Plan-Do-Check-Act
  - SEI CMM
  - Goal/Question/Metric Paradigm

* Not part of CSQE 2002 BOK

Types of Organizational Benchmarking

- Process Benchmarking
  - macro (organizational) level
  - micro (process and project) level
- Performance Benchmarking
- Project Benchmarking
- Strategic Benchmarking

The Benchmarking Process

- Determine Own Current Practices
- Identify Industry Best Practices
- Analyze Best Practices
- Model Best Practices

Standards, Processes, and Models

- Software standards
- Software quality and process initiatives, ventures, and consortia
- Quality management system models
- Software assessment models

Domestic and International standards and specifications

- ISO
  - ISO 2382 (Data Processing Vocabulary)
  - ISO 9000 (Quality Management System)
  - ISO 15504 (SPICE)
  - IEEE/EIA 12207 (Software Life Cycle Processes)
- IEEE
  - 40 Standards in a 4 volume set (1999)
- CMM and CMMI

ISO 9000

- Family of Standards and Guides
- First approved in 1987, latest revision 2000
- Process (not product) focus
- Covers all processes affecting quality of goods and services of organization
- Open ended
- Descriptive (not prescriptive)

ISO 9000-1987

- 9000 — Guidelines for Selection and Use
- 9001 — Model for Design, Development, Production, Installation, and Servicing
- 9002 — Model for Production, Installation, and Servicing
- 9003 — Model for Final Inspection and Test
- 9004 — Guidelines for Quality Management and Quality System Elements

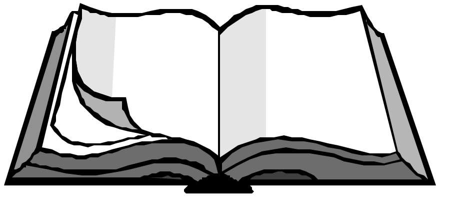
OBSOLETE

ISO 9000:2000

- 9000 — Quality Management Systems — Fundamentals and Vocabulary
- 9001 — Quality Management Systems — Requirements
  - Adds process improvement
  - Combines 9001, 9002, and 9003
- 9004 — Quality Management Systems — Guidelines for Performance Improvements

ISO 9000 Part 3

- 9000-3 Guidelines for Software
- Development, Supply, and Maintenance of Software
- Interprets ISO 9001



ISO/IEC JTC 1

- ISO and International Electrotechnical Commission
- Develops standards for information technology
- Subcommittee SC7 for software engineering standards
- Working Groups for specific standards
- ISO/TC176 to harmonize activities

Domestic and International standards and specifications

- IEEE
  - Standard Collection for Software Engineering
  - ANSI/IEEE 610.12: Standard Glossary of Software Engineering Terminology
  - ANSI/IEEE 730: Software QA Plans
  - ANSI/IEEE 828: Software CM Plans
  - ANSI/IEEE 830: Requirements Specifications
  - ANSI/IEEE 1028: Reviews and Audits
  - ANSI/IEEE 1012: Software V & V Plans
  - ANSI/IEEE 1074: Life Cycle Processes

Other Domestic and International (ISO) Standards and Specifications

- ISO 9126 (Software Product Evaluation)
- ISO 10006 (Project Management)
- ISO 10007 (Configuration Management)
- ISO 10011 (Guidelines for Auditing)

Software Process Models

- Bellcore TR-179
- SEI’s CMM and CMMI
- Trillium Model
- BOOTSTRAP
- ISO SPICE

Bellcore TR-179

- Based on ISO 9001 and 9000-3
- Bellcore Capability Assessment
- Bellcore Model
  - Quality System Framework
  - Quality System Life Cycle Activities
  - Quality System Supporting Activities
- Includes Additional Requirements for Telecommunications Suppliers

TL 9000 Standards

- Quality Excellence for Suppliers of Telecommunications (QuEST) Forum
- Set of common standards including ISO 9000 (both 1994 and 2000)
- Quality System Requirements
- Quality System Metrics
  - Common
  - Software
  - Hardware
  - Service

SEI's Capability Maturity Model

Levels of Maturity

- Level 1: Initial
- Level 2: Repeatable
- Level 3: Defined
- Level 4: Managed
- Level 5: Optimizing

Capability Maturity Model

- Key Process Areas (KPA)
- Software Capability Evaluation (SCE)
- Software Process Assessment (SPA)
- Interim Profile
- CMM-Based Appraisal for Internal Process Improvement (CBA IPI)

Capability Maturity Model Integration

- Combines SE-CMM and SW-CMM
- Continuous or Staged process models
- Software Process Assessment (SPA)
- Appraisal Requirements for CMMI (ARC)
- Standard CMMI Appraisal Method for Process Improvement (SCAMPI)

CMMI Common Process Areas

Category	Process Areas
Process Management	Organizational: Process Focus, Process Definition, Training, Process Performance, and Innovation and Deployment
Project Management	Project Planning and Monitoring and Control Supplier Agreement Management Integrated Project Management Risk Management Integrated Teaming Quantitative Project Management
Engineering	Requirements Development and Management Technical Solution Product Integration Verification and Validation
Support	Configuration Management Process and Product Quality Assurance Measurement and Analysis Organizational Environment for Integration Decision Analysis and Resolution Causal Analysis and Resolution

CMMI Model Representations

Continuous	Staged
• Allows selecting the order of improvement • Enables comparisons • Easy comparison with ISO/IEC 15504 • Easy migration from EIA/IS 731	• Proven sequence of improvements • Permits comparisons of maturity levels • Single summary rating • Easy migration from SW-CMM to CMMI

SEI's CMMI - Continuous

- Improvement across organization and by process area
- Generic Goals (GGs) and Practices (GPs)
- Specific Goals and Practices
- Capability Level Profile

SEI's CMMI - Continuous

Levels of Maturity

- Level 0: Incomplete
- Level 1: Performed
- Level 2: Managed
- Level 3: Defined
- Level 4: Quantitatively Managed
- Level 5: Optimizing

SEI's CMMI - Staged

- Improvement across organization by capability level
- Generic and Specific Goals and Practices
- KPAs mapped from CMM

SEI's CMMI - Staged

Levels of Maturity

- Level 1: Initial
- Level 2: Managed
- Level 3: Defined
- Level 4: Quantitatively Managed
- Level 5: Optimizing

CMM - CMMI Levels

	CMM	CMMI Staged	CMMI Continuous
Level 0			Incomplete
Level 1	Initial	Initial	Performed
Level 2	Repeatable	Managed	Managed
Level 3	Defined	Defined	Defined
Level 4	Managed	Quantitatively Managed	Quantitatively Managed
Level 5	Optimizing	Optimizing	Optimizing

ISO SPICE

- Software Process Improvement for Capability Determination
- ISO 15504
- To Harmonize CMM-Based Efforts
- Used For:
  - Process Assessment
  - Process Improvement
  - Capability Determination
  - Qualification and Training of Assessors

Trillium Model

Trillium Scale

- Level 1: Unstructured
- Level 2: Repeatable and Project Oriented
- Level 3: Defined and Process Oriented
- Level 4: Managed and Integrated
- Level 5: Fully Integrated

Trillium Model

Capability Areas

- Organizational Process Quality
- CHR Development and Management
- Process
- Management
- Quality System
- Development Practices
- Development Environment
- Customer Support

Trillium Model

- Capability Evaluation/Joint-Assessment
- Capability Assessment and Improvement
- Capability Self-Assessment
- Continuous Improvement (CI) Program
- Capability Profile, Levels, Road Maps, and Practices

BOOTSTRAP

- AKA ESPRIT
- Bootstrap Assessment
- Bootstrap Evaluation
- Process Improvements

BOOTSTRAP Assessment

- Assesses Organization and Projects
- Quality-Attribute Hierarchy
  - Clusters
  - Elementary Attributes
- Process Dimensions
  - Organization
  - Methodology
  - Technology

Leadership Tools and Skills

- Organizational leadership
- Analyzing Current Situations
- SWOT Analysis
- Implementing and Managing Change
- Quality Initiatives
- Cross-Functional Collaboration
- Knowledge Management
- Motivation Techniques

SWOT Analysis

- Strengths and Weaknesses
  – Internal to organization
- Opportunities and Threats
  – External sources

Management Science

- Fredrick Taylor (time and motion)
- Hawthorne Studies (cause and effect)
- Abraham Maslow (need hierarchy)
- Douglas McGregor (Theory X and Y)
- Frederick Herzberg (2 motivation factors)
- Organizational Empowerment
- Motivating the Work Force

Team Roles and Responsibilities

- Champion
- Sponsor
- Facilitator
- Coach
- Leader
- Team member
- Recorder
- Timekeeper

Team Life Cycle

Bruce Tuckerman's Stages:

- Forming [Build Phase]
- Storming [Build Phase]
- Norming [Develop Phase]
- Performing [Optimize Phase]

Team Applications

Team Type	Structure	Applications
Improvement	8-10 from a department	Quality or productivity issues
Quality	8-10 from a department	Initially quality or performance; can evolve to self directed teams
Project	ranges; may be part or all managers	Focus on a specific project; disbands when done
Cross Functional	8-12 from different areas	Like project teams but for policies and operational issues
Self Directed	6-15; usually a natural area team	Can work from objectives; requires training

Team Dynamics

- Selecting Team Members
- Team Size
- Team Diversity
- Dominant or Disruptive Team Members
- Common Team Problems
- Dealing with Team Problems

Team Tools

- Nominal Group Technique (NGT)
- Multivoting
- Brainstorming
- Joint Application Development (JAD)
- Rapid Application Development (RAD)
- Facilitated Application Specification Technique (FAST)

Facilitation skills

- Team leader
- Facilitator
- Team participants
- Avoiding chaos
- Maintaining balance
- Keeping focused

Conflicts

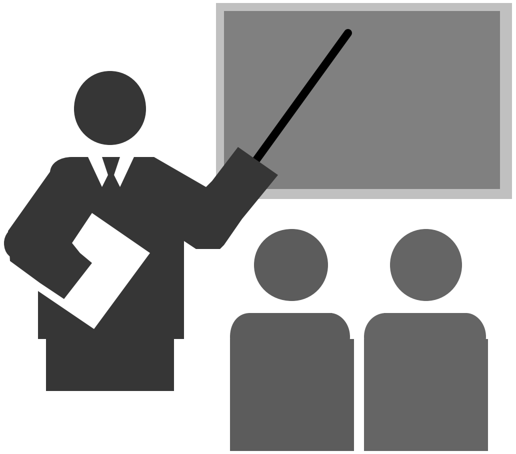
- Expect them
- Manage conflicts
- Resolve conflicts
- Handle conflicts
- Force Field Analysis
- Negotiation

Meetings

- Meeting Management
- Team performance
- Operating guidelines
- Meeting structure

Verbal Communication Skills

- Purpose of Presentation
- Type of Presentation
- Analyze Audience Needs
- Determine Objective or Outcome
- Structure to Support Main Idea
- Get the Audience Attention
- Meet Audience Objectives
- Strong Conclusion
- Practice Delivery
- Use Visual Aids



Written Communication Skills

- Memo
- Report
- Letter

Interviewing Skills

- Organize the Interview
  - Have a Plan
  - Simple Questions
  - Stay Focused
  - Don’t Threaten
  - Ask Permission
  - Avoid Ambiguity
  - Avoid Manipulation
- Closed and Open Questions
- Stay In-Bounds

Effective Listening Skills

- Non-Listener
- Marginal Listener
- Evaluative Listener
- Active Listener

Ethical Conduct

- Professional codes of ethics
- ASQ Code of Ethics
- Professional conduct and competence
- Conflicts of interests
- Software Licensing
- Software Copyrights

Professional Conduct Terminology

- Conflict of interest
- Ethical
- Ethics
- Legal
- Malpractice
- Negligence
- Notification
- Recall
- Regulation

Conflicts of Interest Examples

- Providing recommendations on the purchase of products or services while owning an interest in the producer
- Presenting independent assessment results to a client with recommendations to use your services
- Participating in awarding a contract to someone who manages you in another context
- Evaluating the corrective actions of another party which your organization has performed for them

Software Liability and Safety Issues

- Legal issues involving software
  - product liability
  - product safety
  - negligence
  - customer notification requirements
- Other legal or regulatory issues

Professional Development

- Software Quality Training
- Quality Auditor Training
- Software Engineering Training
- Professional Certification
- Training Needs Analysis

Professional Training and Development

- Subject areas for SQE
- Available training resources, materials, and providers
- SQE Professional
  - Societies
  - Technical Associations
  - Organizations

SQE Subject Areas

- Design of Experiments
- Failure Mode Analysis
- Internal Auditing
- Reliability Engineering
- Statistical Process Control
- Software Quality Engineering
- Strategic Quality Planning

SQE Training Resources

- RAB Accredited ISO Courses
- ASQ
- SEI
- Colleges and Universities
- Private Organizations and Individuals

Professional Societies for SQEs

- ASQ (SQE Certification)
- SEI (SPINs)
- Deming User Group
- ACM
- IEEE

Quality Management

- General Knowledge, Conduct, and Ethics
- **Software Quality Management**
- Software Engineering Processes
- Program and Project Management
- Software Metrics, Measurement, and Analytical Methods
- Software Verification and Validation (V&V)
- Software Configuration Management

Software Quality Management

- Goals and objectives
- Principals
- Policies
- Strategic quality goals
- Tactical quality goals
- Process quality
- Tools and methods
- Audits

Goals and Objectives

- Quality goals and objectives
- Outsourced services
- Planning
- Software quality management systems documentation
- Customer requirements

SQA Tasks

- Generation of QA Documentation
- Review of Project Materials
- Auditing
- Monitoring of Project Status
- Inspection of Delivered Items
- Monitoring Corrective Actions
- Participation in Project Activities
- Guidance of Project
- Testing Oversight

Outsourced Services

- Process work (operations)
- Project work
- Global (strategic)
- Tactical (partial)
- Subcontracting (targeted)
- Offshore outsourcing
- Subcontractor management (SW-CMM)

Outsourcing Pros and Cons

Forces for:

- Lower costs
- Risk sharing or reduction
- Economies of scale
- Greater skill pool
- Greater focus
- More control
- More professional
- Cash infusion

Forces against:

- Higher costs
- Risk exposure
- Dis economies of scale
- Limited access to knowledge
- Loss of intellectual capital
- Conflicting agendas

Software Quality Management

- Planning
- Tracking
- Organizational and Professional Software QA Training

SQ Management Planning

- Product and Project SQ Goals and Objectives
- Customer Requirements for Quality
- Quality and Customer Support Activities
- Software Security, Safety, and Hazard Analysis

Planning Issues

- Software Development is Private
- Traditional Programmer Training Misses Planning and Measurement
- Standards not Followed
- “Cowboy Culture”
- QE Considered an Outsider

Planning Solutions

- On the Job Training
- Positive Experience with Standards and Metrics
- Professional Attitude
- Discipline
- Planning Initiative (WWWWWH)

Planning Initiative Focus

- Defect Prevention
- Identification of Problem Areas
- Risk Analysis and Mitigation
- Correct Reviews and Inspections
- Documentation of Required Procedures

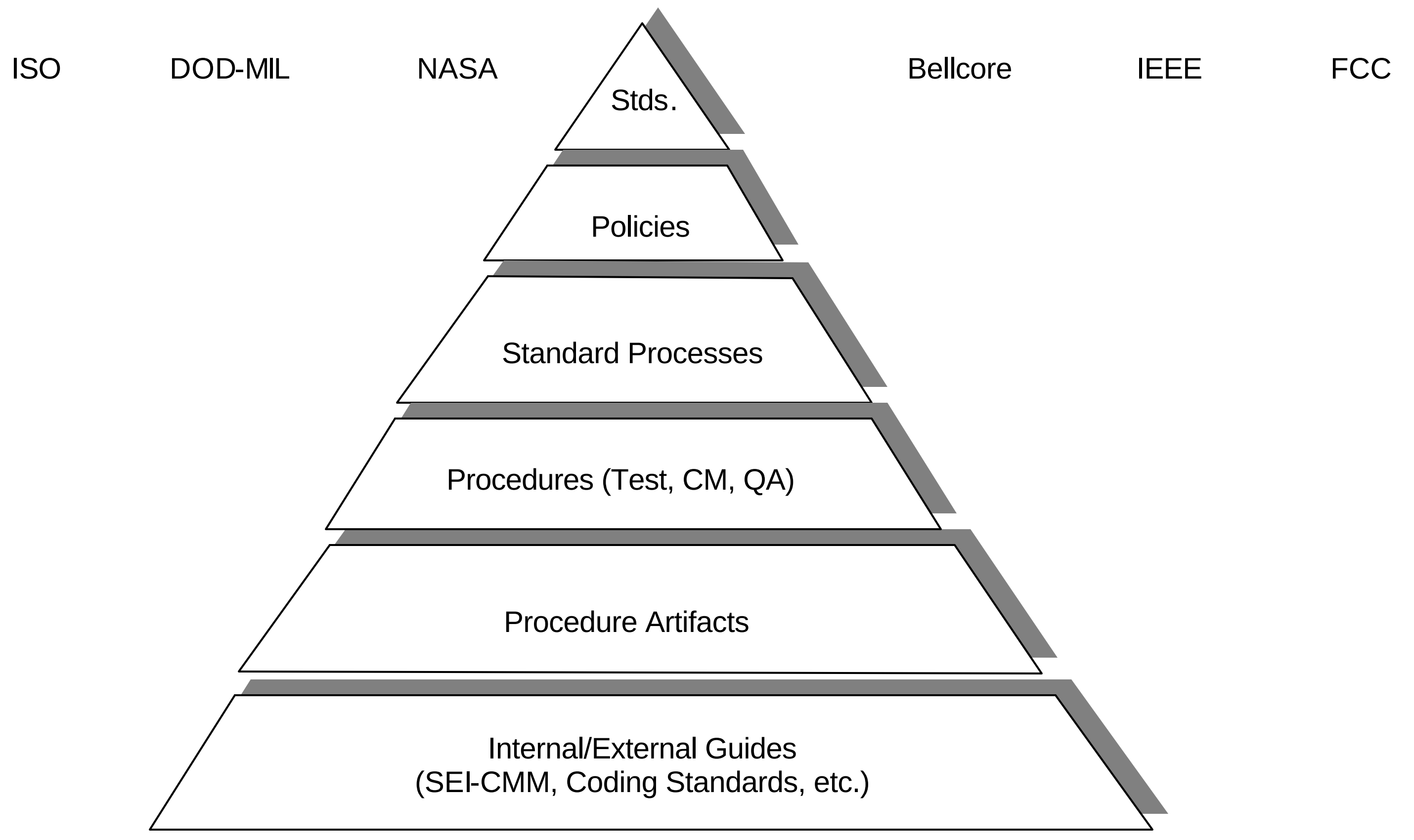
Planning Initiative Levers

- Contractually Imposed Standards
- Analysis of System Requirements
- Schedules
- Development Costs
- User Concerns

Product and Project SQ Goals and Objectives

- Predictability
- Consistency
- Objective Measurements and Evidence

Quality Planning System



Customer Requirements for Quality

- Reliability (MTTF)
- Pass/Fail Criteria and Evidence
- Traceability
- Software Baseline Control
- Product Consistency

Planning for Quality

- Phase Based Activities
- Software Quality Assurance Plan (SQAP)
- ISO 9000-3 requirements
  - Planning
  - Tracking
- Documentation
  - SQM Plan (IEEE 730)
  - Requirements

Requirements*

- Solve a problem
- Must be met to satisfy the contract

- Requirements analysis vs.
  Requirements specification

- Problem analysis vs.
  Product description

* IEEE 610.12

Requirements Analysis

- Object Oriented
- Function Oriented
- State Oriented
- Behavioral Analysis
- Nonbehavioral Analysis

Nonbehavioral Quality Factors

- Efficiency
- Reliability
- Security
- Testability
- Understandability
- Maintainability
- Human engineering
- Portability
- Visibility
- Capacity
- Standards compliance

Requirements Elicitation

- Problem recognition
- Evaluation and synthesis
- Modeling
- Specification
- Review

Customer/User Reviews

- JAD
- RAD
- QFD
- FAST
- Context-free questions

Requirement Analysis and Specification

- Domains of the problem
  – Information
  – Function
  – Behavior
- Problem partitioning
- Representing requirements

Quality Records*

- Records and Data Collection
- Use
- Changes
- Storage
- Maintenance
- Retention

* Not part of CSQE 2002 BOK

Issues for Quality Related Data

- Data Entry
- Change Control
- Data Availability
- Data Integrity
- Data Security

Quality Management Methods

- Reviews, Inspections, and Testing
- Change Management
- Cost of Quality (COQ)
- Quality Data Tracking
- Problem Reporting and Corrective Actions
- Quality Improvement Processes

SQA Tools

- Requirements Tracer
- Database Analyzer
- Complexity Analyzer
- Logic Analyzer
- Reliability Model
- Simulators
- Standards Analyzer
- Data-flow Analyzer
- Interface Analyzer
- Test Generator
- Test Management System

Process and Technology Change Management

- Software process modeling
- SEPGs
- Software process assessment and evaluation
- Software process and technology change management theory and methods
- Barriers to the implementation or success of quality improvement efforts and quality systems

Audit and Assessment Management

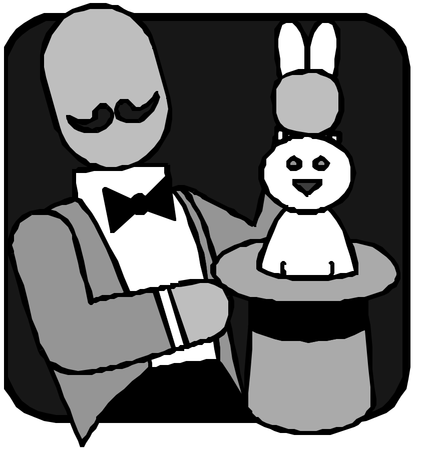
- Program development and administration
- Audit preparation and execution
- Audit reporting and follow up

Software Quality Factors

- SQFs
- Definitions
- Metrics

Software Quality Methods

- "Bag of Tricks"
- Inspections
- Tests
- Assessments



Cost of Quality

- Traditional Concepts
- Levels
- Locations

Quality Cost Analysis

- ***Quality Cost Measurement*** is a cost control system used to identify opportunities for reducing the controllable quality-related costs.
- The ***Cost of Quality*** is the total amount the company spends to achieve and cope with the quality of its product.
- This includes the company's <u>investments</u> in improving quality, and its <u>expenses</u> arising from inadequate quality.
- A key goal of the quality engineer is to help the company minimize its cost of quality.

Quality-Related Costs

•*Prevention*	•*Appraisal*
•Cost of preventing customer dissatisfaction, including errors or weaknesses in software, design, documentation, and support.	**•Cost of inspection (testing, reviews, etc.).**
•*Internal Failure*	•*External Failure*
•Cost of dealing with errors discovered during development and testing. Note that the company loses money as a user (who can’t make the product work) and as a developer (who has to investigate, and possibly fix and retest it).	**•Cost of dealing with errors that affect your customers, after the product is released.**

Examples of Quality Costs

•*Prevention*	•*Appraisal*
• **Staff training** • **Requirements analysis & early prototyping** • **Fault-tolerant design** • **Defensive programming** • **Usability analysis** • **Clear specification** • **Accurate internal documentation** • **Pre-purchase evaluation of the reliability of development tools**	• **Design review** • **Code inspection** • **Glass box testing** • **Black box testing** • **Training testers** • **Beta testing** • **Usability testing** • **Pre-release out-of-box testing by customer service staff**
•*Internal Failure*	•*External Failure*
• **Bug fixes** • **Regression testing** • **Wasted in-house user time** • **Wasted tester time** • **Wasted writer time** • **Wasted marketer time** • **Wasted advertisements** • **Direct cost of late shipment** • **Opportunity cost of late shipment**	• **Lost sales and lost customer goodwill** • **Technical support calls** • **Writing answer books (for Support)** • **Investigating complaints** • **Supporting multiple versions in the field** • **Refunds, recalls, warranty, liability costs** • **Interim bug fix releases** • **Shipping updated product** • **PR to soften bad reviews** • **Discounts to resellers**

Customers' Quality Costs

•Seller: external costs	•Customer: failure costs (seller's externalized costs)
•*These illustrate costs absorbed by the seller that releases a defective product.* –**Lost sales and lost customer goodwill** –**Technical support calls** –**Writing answer books (for Support)** –**Investigating complaints** –**Refunds, recalls, warranty, liability costs** –**Government investigations** –**Supporting multiple versions in the field** –**Interim bug fix releases** –**Shipping updated product** –**PR to soften bad reviews** –**Discounts to resellers**	•*These illustrate costs absorbed by the customer who buys a defective product.* –**Wasted time** –**Lost data** –**Lost business** –**Embarrassment** –**Frustrated employees quit** –**Failure during one-time-only tasks, e.g. demos to prospective customers** –**Cost of replacing product** –**Reconfiguring the system** –**Cost of recovery software** –**Tech support fees** –**Injury / death**

The $q / $Q Debate

Topic	"Little Q"	"Big Q"
Products	Manufactured goods	All products, goods, and services
Processes	Process directly related to production	All processes, support, business, etc.
Customer	Clients who buy the products	All who are affected
Industries	Manufacturing	All industries
Cost of Poor Quality	Costs associated with deficient manufactured goods	All costs that would disappear if everything were perfect

Quality Data Tracking

- Principles
  - Understanding
  - Evaluation
  - Control
  - Prediction
- Models
- Impact on the organization
- Management Support

Problem Reporting and Corrective Actions

- Reporting Procedures
- Corrective Actions
- Quality Improvement Processes
  - Defect prevention
  - Defect detection and removal
  - Trend analysis
  - Pareto Analysis
  - Reviews
  - Testing
- Barriers to Quality Improvements
  - Ill-defined process
  - Poor process management

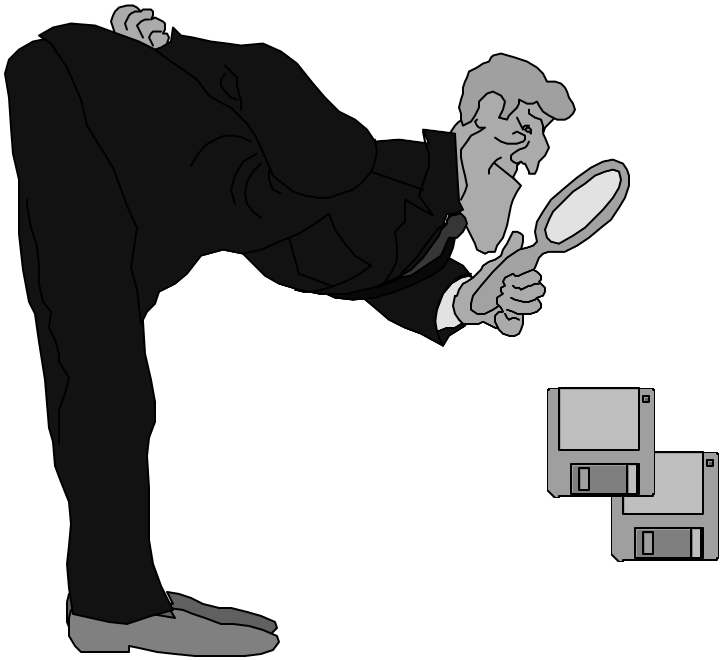
Problem Reporting and Corrective Action Procedures

- Defect Tracking Systems
- Event Recording Systems
- Development Needs
- QA Requirements
- Support Issues

Supporting Activities

- Defect Tracking
- Technical Support
- Change Control Board (CCB)

Software Audits



Software Audits

- Program Development and Administration
- Audit Preparation and Execution
- Audit Reporting and Follow-up

Audit Program Objectives

- Auditing stages
  - Planning and preparation
  - Performance
  - Reporting
  - Corrective action and follow-up

Audit Objectives

- Compliance with standards
- Effectiveness of controls
- Opportunity for improvement
- Regulatory requirements
- Permit registration

Reasons for Audits

- Contractually required
- Verify QMS meets a standard
- Confirm QMS implementation
- Verify QMS effectiveness
- Evaluate compliance with standards.

Audit Responsibilities

- Audit team member
- Lead Auditor
- Client
- Auditee

Auditing Standards

- ANSI/ISO/ASQC Q10011-1994 Guideline for Auditing QMS
- ISO 9001:2000
  - [ISO 9000-3 Guideline]
- IEEE 1028
- TickIT
- SEI (CMMI)

Software Audits

- Terms
- Types
- Methods
- Steps
- Process
- Planning

Audit Terms

- Audit
- Checklists
- Compliance
- Conformance
- Customers
- Interviews
- Objective Evidence

Audit Types

- First party
- Second party
- Third party
- Internal
- External

Audit Types
(Continued)

- System
- Process
- Product
- Compliance
- Regulatory

Audit Types
(Continued)

- Management
- Quality
- Functional configuration
- Physical configuration
- Administrative Audit
- etc.

Audit Methodology

- Purpose
- Perspectives
- Frequency
- Criteria
- Procedures

Audit Precepts

- Function of management ☑
- Qualified auditors ☑
- Measures against standards ☑
- Objective evidence ☑
- Focus on control system ☑

Management Audits

To determine that:

- Adequate controls exist
- Controls are being implemented
- Controls really work

Audit Perspectives

- Customers
  - Auditee
  - Client
  - Organization
- Auditor

Audit Process

- Objectives
- Scope
- Entry criteria
- Assessment criteria
- Exit criteria

Audit Frequency

- Project milestone
- Quality milestone
- External requirements
- Internal request
- Major organizational change

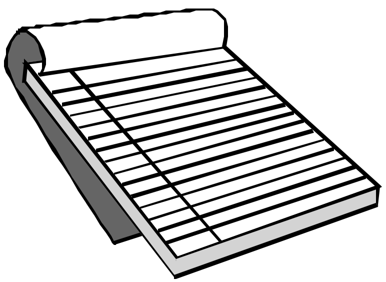
Audit Tools

- Checklists
- Authoritative Documents
- Flowcharts
- Interviewing
- Data Collection

Data Collection

- Interviews
  - Data gathering
  - Not objective evidence
  - Conducting an interview
- Sampling
- Trace Forward
- Trace Back

Audit Data Sources



- Physical properties
- Observations
- Documents and records
- Interviews
- Patterns of information

Audit Phases

- Preparation
- Performance
- Reporting
- Closure

Audit steps

- Planning
- Preparation
- Execution
- Reporting
- Corrective action
- Verification
- Follow-up

Engineering Processes

- General Knowledge, Conduct, and Ethics
- Software Quality Management
- **Software Engineering Processes**
- Program and Project Management
- Software Metrics, Measurement, and Analytical Methods
- Software Verification and Validation (V&V)
- Software Configuration Management

Software Engineering Processes

- Environmental conditions
- Requirements management
- Requirements engineering
- Analysis, design, and development methods and tools
- Maintenance management

Software Environmental Conditions

- Life cycles
- Systems architecture

Why Worry About SDLC?

- "Cosmic Glue"
- Where Quality Goes In
- Basis for Planning
- Basis for Communication
- Set Milestones
- Reduces Ambiguity
- Defines Roles
- Surfaces Issues

Development Life Cycles

- Three generic phases
  - Definition
  - Development
  - Maintenance
- Many models
  - Balance among tradeoffs
  - Choose one that adjusts to expected changes

Generic Life Cycle Phases

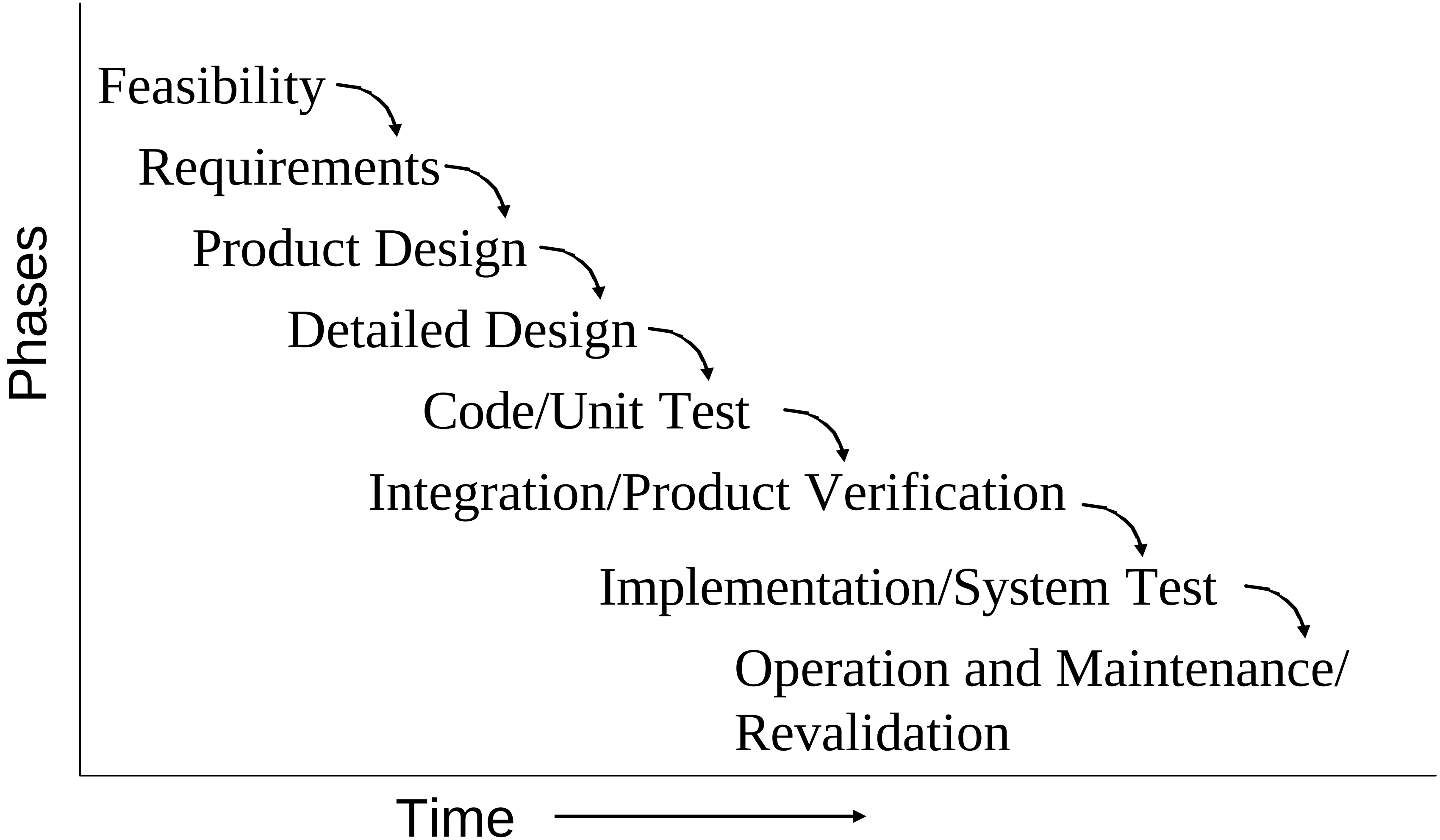
- Definition
  - What information
  - What functions
  - What interfaces
  - What design constraints
  - What validation criteria
- Development phase
  - How data is structured
  - How system is architected
  - How procedures work
  - How programmed
  - How tested
- Maintenance
  - Error corrections
  - Adaptations evolved
  - Enhancements made
- Related activities
  - Quality assurance
  - Configuration management
  - Project monitoring
  - Measurement

Software Development Life Cycle Models

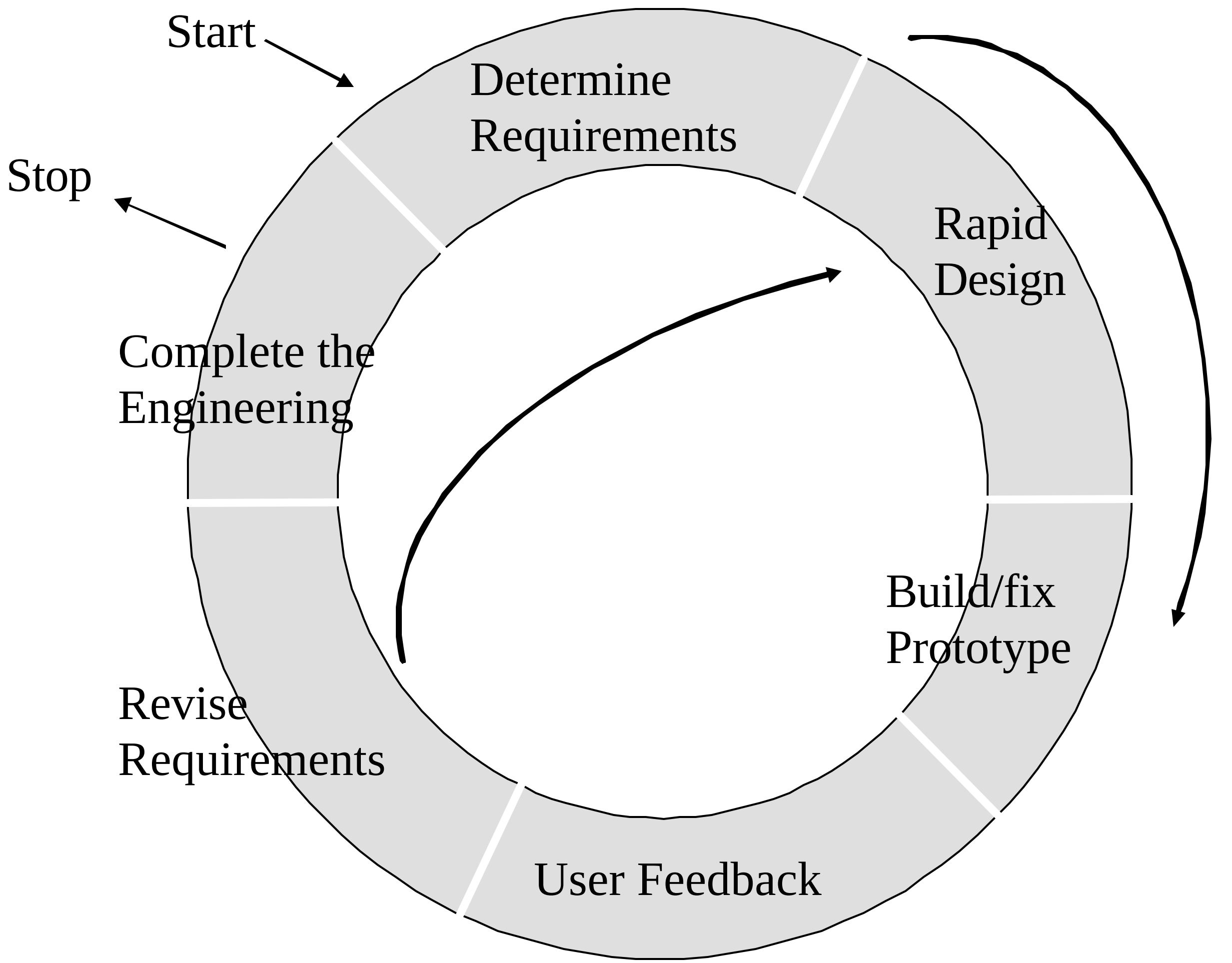
- Waterfall
- Prototyping* and Mock-ups
- Spiral
- Incremental Development
- Decomposition/Integration
- Cleanroom*
- Fourth Generation Techniques*
- Object-Oriented
- Hybrid Models

* Not included in Primer

Waterfall


Feasibility
Requirements
Product Design
Detailed Design
Code/Unit Test
Integration/Product Verification
Implementation/System Test
Operation and Maintenance/
Revalidation
Phases
Time

Prototyping



Mock-Up Approach

Revise Requirements

Revise Prototype

Evaluate Prototype

Detailed Design

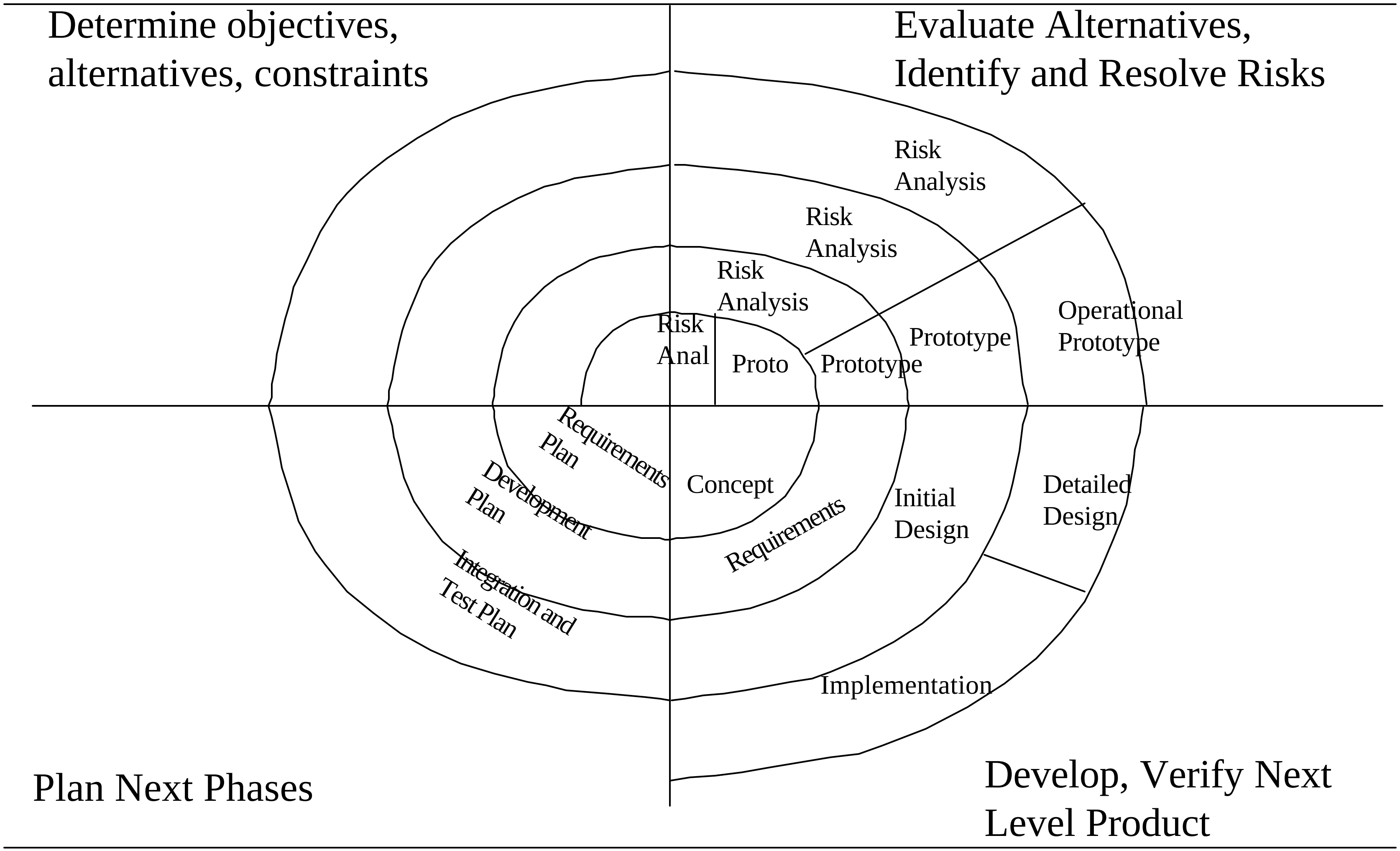
Code/Unit Test

Integration/Product Verification

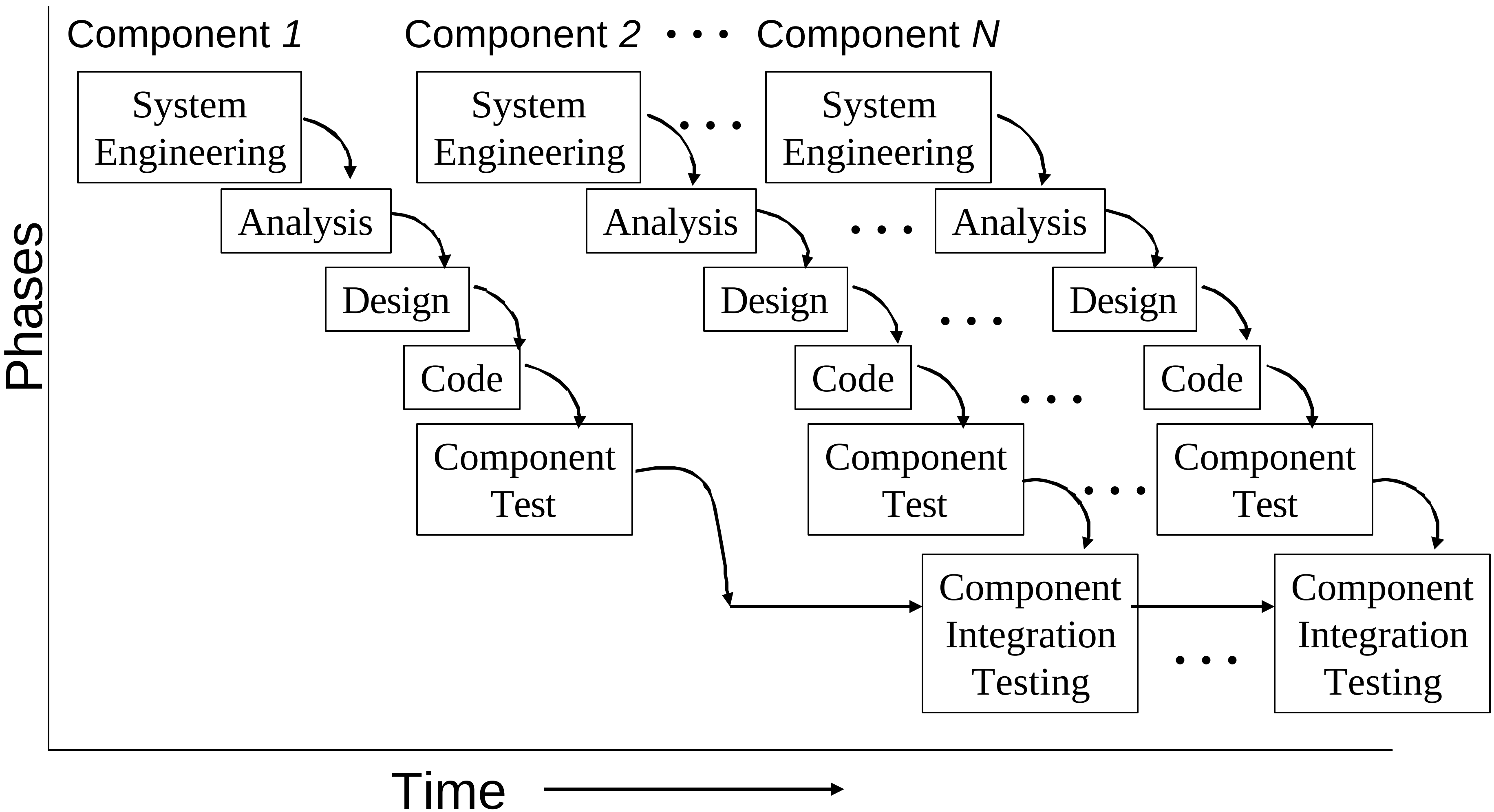
Implementation/System Test

Operation and Maintenance/ Revalidation

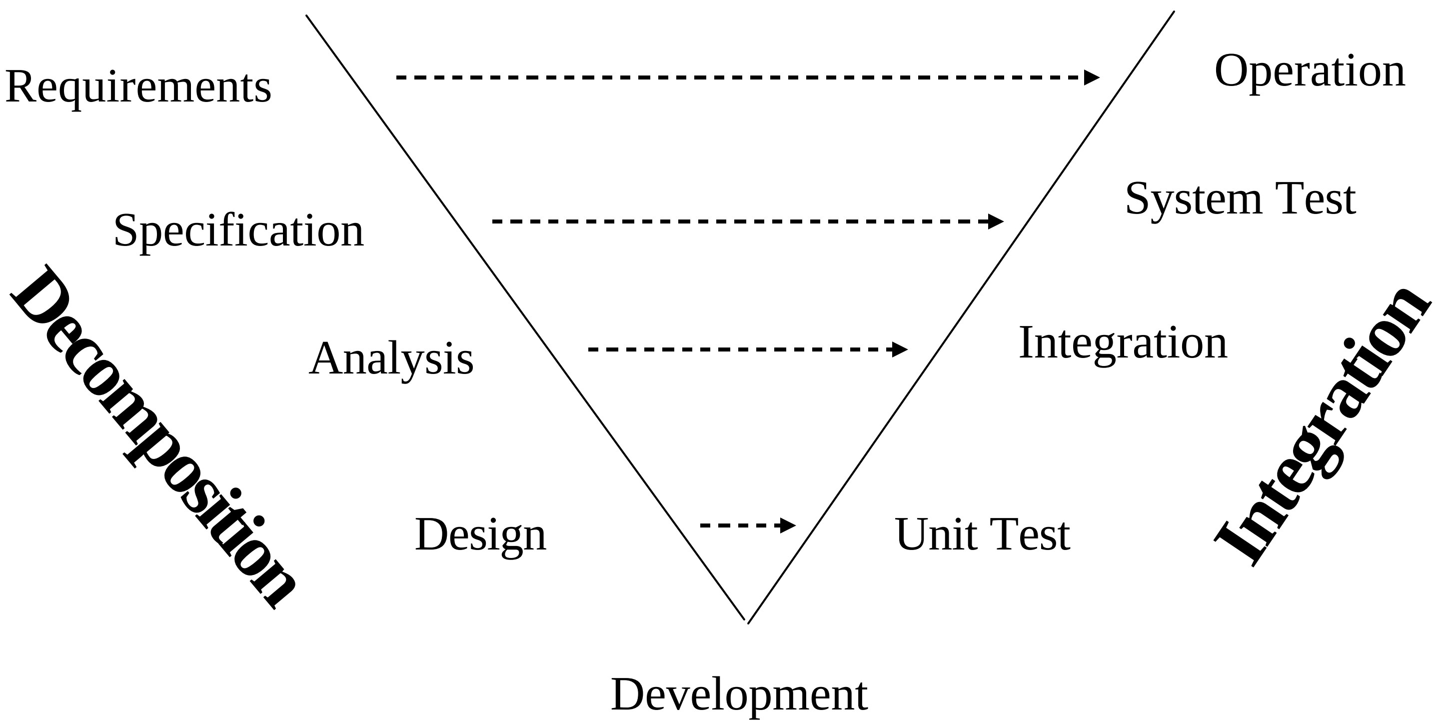
Spiral Model



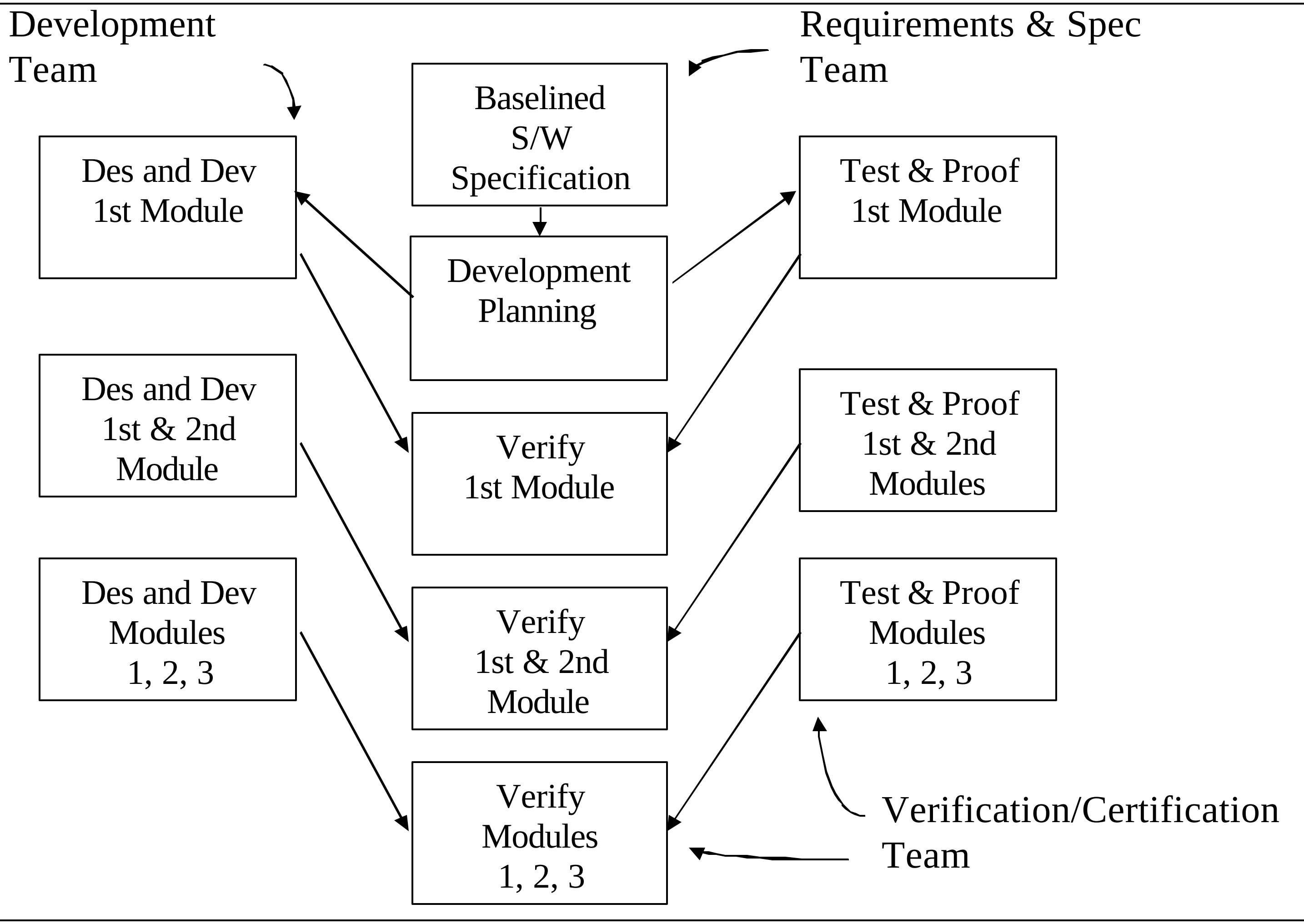
Incremental Development



Decomposition/Integration

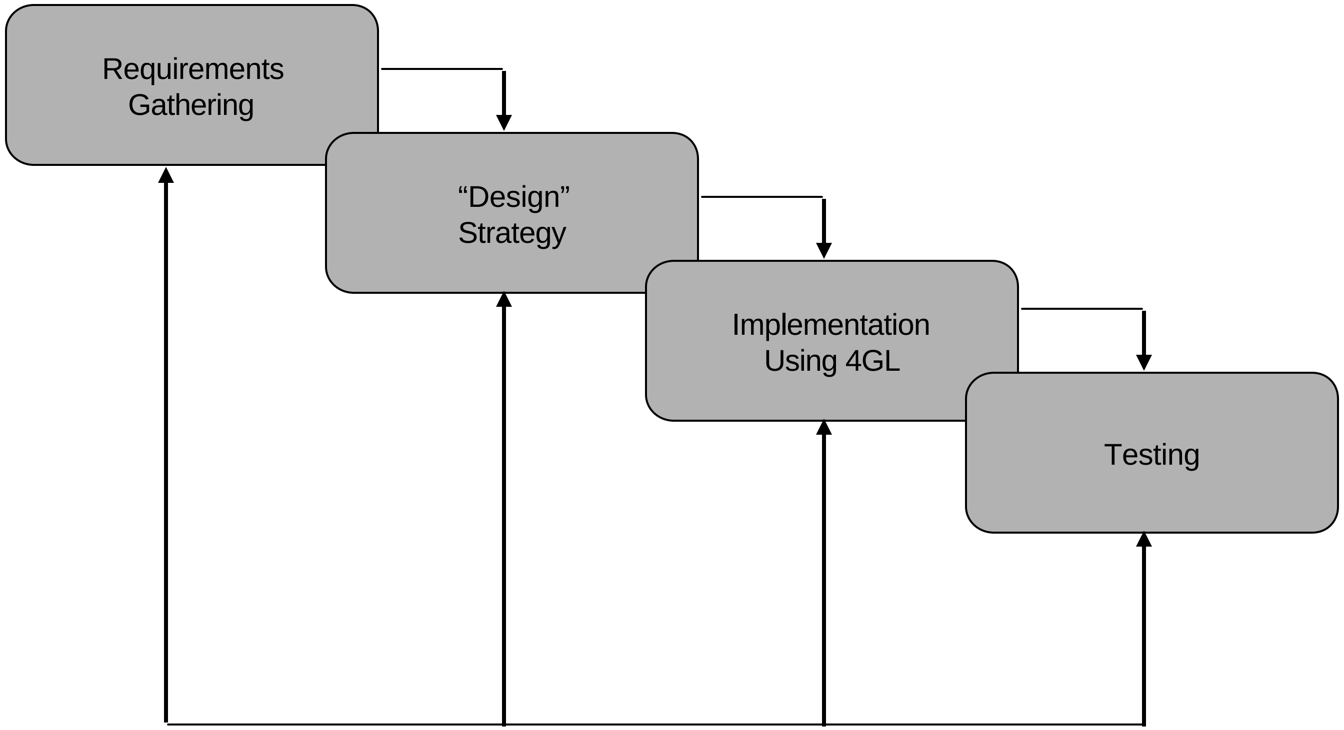


Cleanroom

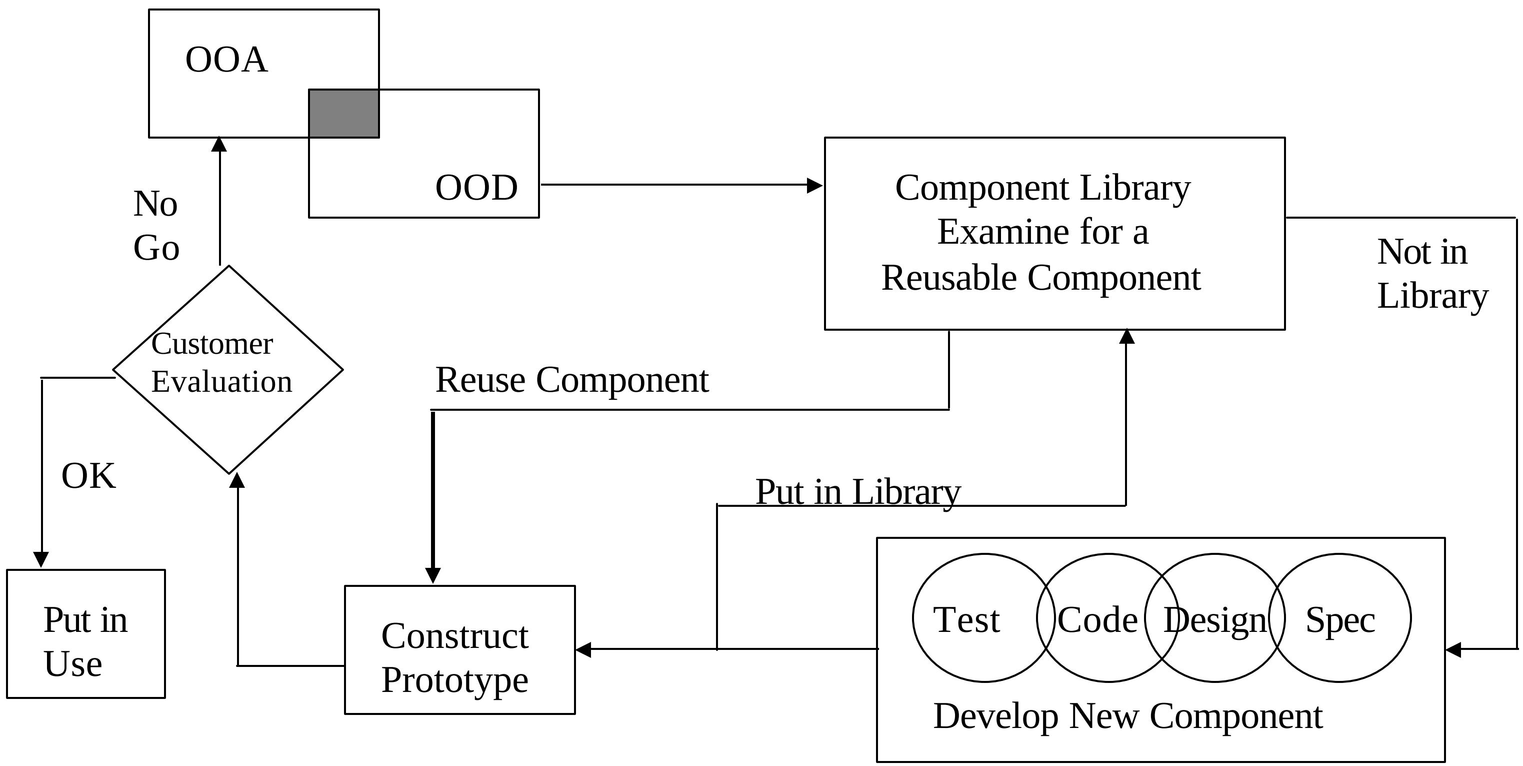


Development Team
Requirements & Spec Team
Baselined S/W Specification
Des and Dev 1st Module
Development Planning
Test & Proof 1st Module
Des and Dev 1st & 2nd Module
Verify 1st Module
Test & Proof 1st & 2nd Modules
Des and Dev Modules 1, 2, 3
Verify 1st & 2nd Module
Test & Proof Modules 1, 2, 3
Verify Modules 1, 2, 3
Verification/Certification Team

Fourth Generation



Object-Oriented



Lifecycle Selection

Life Cycle	Project Requirements
Waterfall	Known, unchanging
Decomposition/ Integration	Known, unchanging
Prototyping	Unknown, changing
Spiral	Unknown, unchanging
Cleanroom	Known, provable

Hybrid Life Cycles

- Should be planned
- Based on Project Requirements
- Focus on Milestone deliverables

Pitfalls In SDLCs

- Poor Choice of SDLC
- Lack of Understanding
- Conflicts of Interest
- Violation of Process
- Poor Feedback

Systems Architecture

- Collection of components
- Interactions between components
- Patterns of similar components and interactions
  - Client Server (C/S)
  - N-tier
  - B to B (B2B)
  - B to C (B2C)
  - B to E (B2E)
  - Web based

Web Architectures (Internet/Intranet/Extranet)

- Wireless
- Messaging
- Collaboration Software
- Market to Market (M2M)
- B to Government (B2G)
- B to Anyone (B2A)
- Application to Application (A2A)

Software Requirements Management

- Requirements prioritization and evaluation
- Requirements change management
- Bi-directional requirements traceability

Requirements Prioritization

- Critical, Important, Desirable
- Required, Conditionally Required, Objective
- Assignment to future releases
- Requirements should be:
  - Unambiguous
  - Correct
  - Verifiable
  - Consistent
  - Complete

Requirements Change Management

- Submitting
- Evaluating
- Reviewing
- Scheduling
- Implementing

Bi-directional Traceability

Of:

- Customer Requirements
- Engineering Requirements
- Design Components
- Code
- Tests
- V & V Results

- Built one way (top-down)
- Bottom-up trace derived

Verifying:

- Requirements to Design
- Design to Code
- Requirements to Tests

Validating:

- Requirements to Test Results

Software Requirements Engineering

- Requirements types
- Requirements elicitation
- Requirements analysis and modeling
- System and software requirements specifications

Types of Requirements

- Input
- Output
- Reliability
- Availability
- Maintainability
- Performance
- Accessibility
- Environmental conditions
- Ergonomic
- Safety
- Security
- Facility requirements
- Transportability
- Training
- Documentation
- External interfaces
- Testing
- Quality provisions
- Policy and regulatory
- Compatibility to existing systems
- Standards and technical policies
- Conversion
- Growth capacity
- Installation
- Contractual
- Regulatory

Quality Requirements*

Product description

- General requirements
- Identifications and indications
- Functionality
- Reliability
- Usability
- Efficiency
- Maintainability
- Portability

User Documentation

- Completeness
- Correctness
- Consistency
- Understandability
- Ease of overview

Programs and Data

- Functionality
- Reliability
- Usability
- Efficiency
- Maintainability
- Portability

* ISO/IEC 12119: 1994(E) (IEEE 1465-1998)

Requirements Elicitation

- Problem recognition
- Evaluation and synthesis
- Modeling
- Specification
- Review

Requirements Gathering Processes

- Interviews
- Document Analysis
- Brainstorming
- Requirements Workshop
- Prototyping
- Use Cases
- Storyboards
- Interfaces Analysis
- Modeling

Requirements Gathering Practices*

1. Project vision and scope
2. Project glossary
3. Joint requirements w/users
4. Requirement rationales
5. Train analysts and users about requirements
6. Have requirements CM
7. Prioritize requirements
8. Incremental development
9. Use reviews and inspections
10. Use a requirements tool
11. Use proven techniques
12. Include domain experts
13. Evolve mechanisms
14. Use CPI
15. Involve users throughout
16. V&V requirements

* R.R.Young,"Effective Requirements Practices (Addiston-Wesley, 2001)

Requirements Obstacles

- User procedures
- Current capabilities
- Formal business rules
- Gold plating

User Procedure Issues

- Invalid Practices
- Workarounds
- SOPs
- Local Policies
- Antiquated Business Practices

Current Capability Issues

- Current system functionality
- Planned enhancements
- Offline processes
- Rogue applications
- Current system limitations

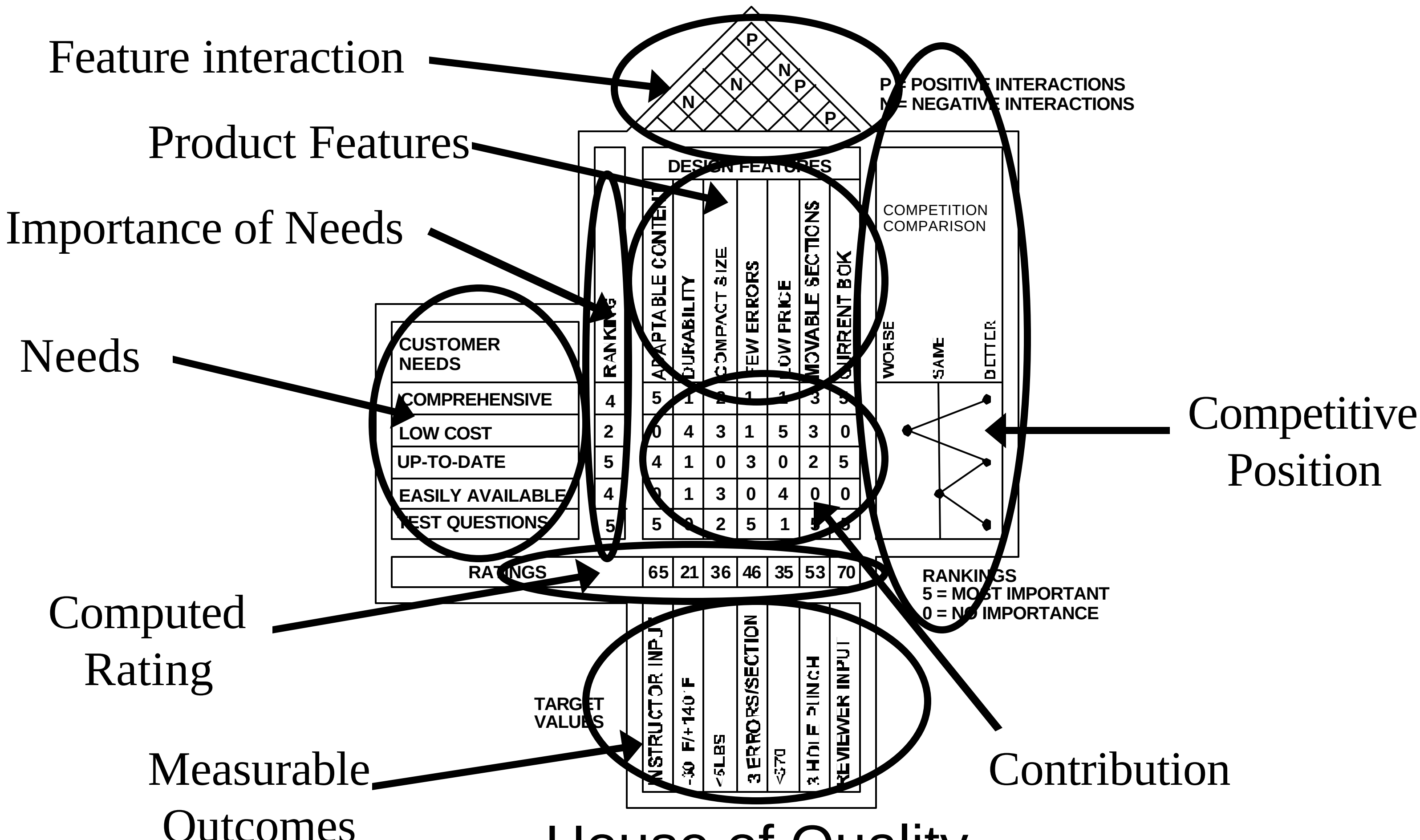
Formal Business Rules

- Legislation
- Local policy
- Industry standards
- Partnering agreements
- Regulations

Gold Plating

- Cutting-edge technology
- User desired features
- Management information
- Technical staff desire to provide better products
- Requirements creep

Quality Function Deployment (QFD)



House of Quality

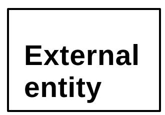
Requirements Analysis and Modeling

- Represent the data domain
- Model
  - System data
  - Functions
  - Behaviors
- Partition into layers
- Move from essential descriptions toward implementation details

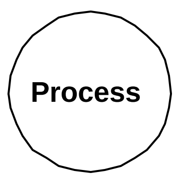
Requirements Analysis and Modeling

- Data Flow Diagram (DFD)
- Control Flow Diagram (CFD)
- Data Dictionary
- Entity Relationship Diagrams (E-R)
- State Transition Diagram
- Data Content Model
- Process Specifications
- Control Specifications

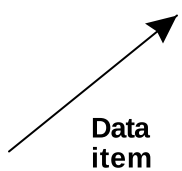
Data Flow Diagram (DFD)
Notation



A producer or consumer of information that resides outside the bounds of the system to be modeled.



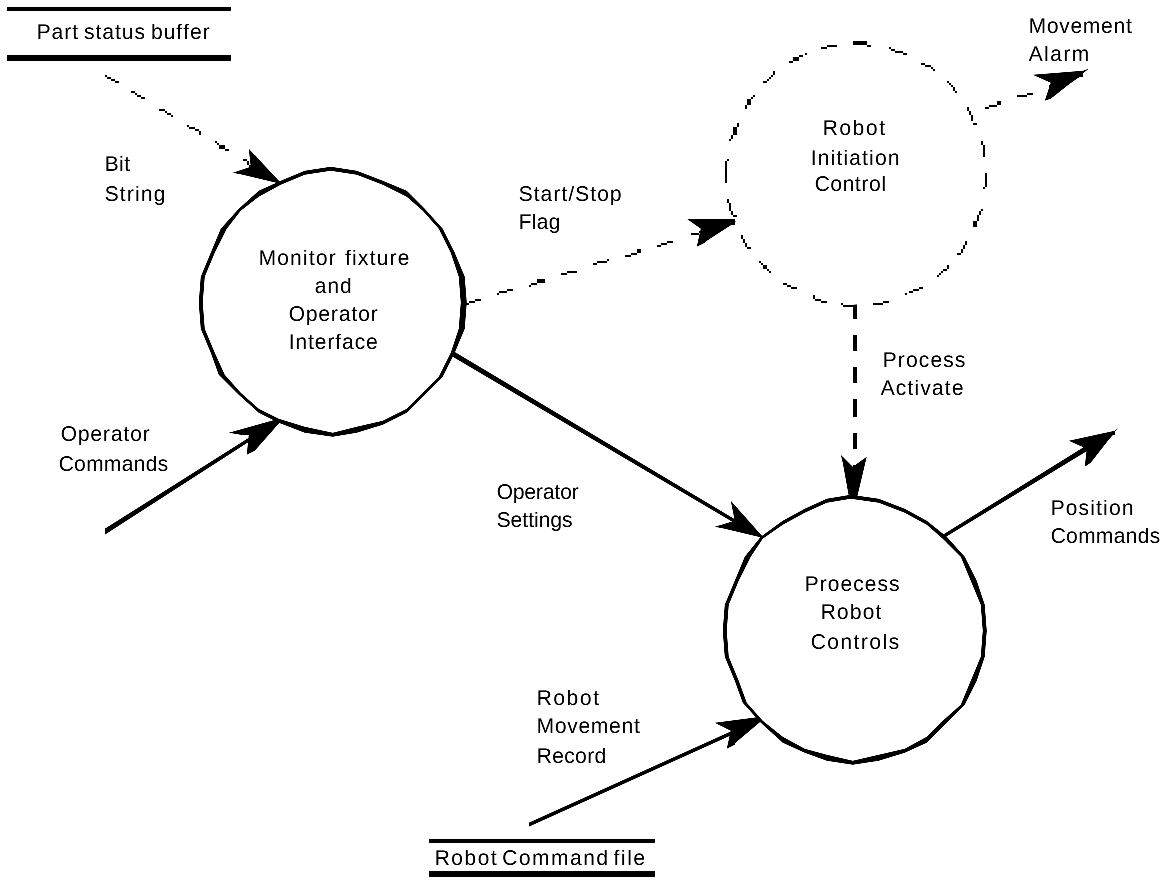
A transformer of information that resides within the bounds of the system to be modeled.



A data item or collection of data items; the arrowhead indicates the direction of data flow.

A repository of data that is to be stored for use by one or more processes; may be as simple as a buffer or queue or as sophisticated as a relational data base.

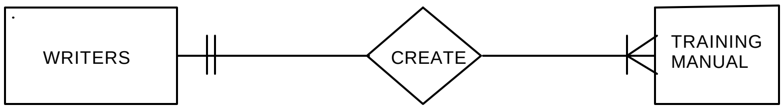
Control Flow Diagram (CFD) Illustration



Data Dictionary

- Represents data content of objects
- Lists all data elements
- Rigorously defines
  - Inputs
  - Outputs
  - Data Stores
  - Intermediate calculations

Entity Relationship Diagrams (E-R)



E-R Diagram Illustration

TITLE	SECTIONS	QUESTIONS	EDIT	REVIEW	ETC.

Data Object Table Illustration

Requirements Analysis Common Characteristics

Methods:

- Support analysis
- Hierarchical representation
- Force careful consideration of interfaces
- Provide foundation for design and implementation

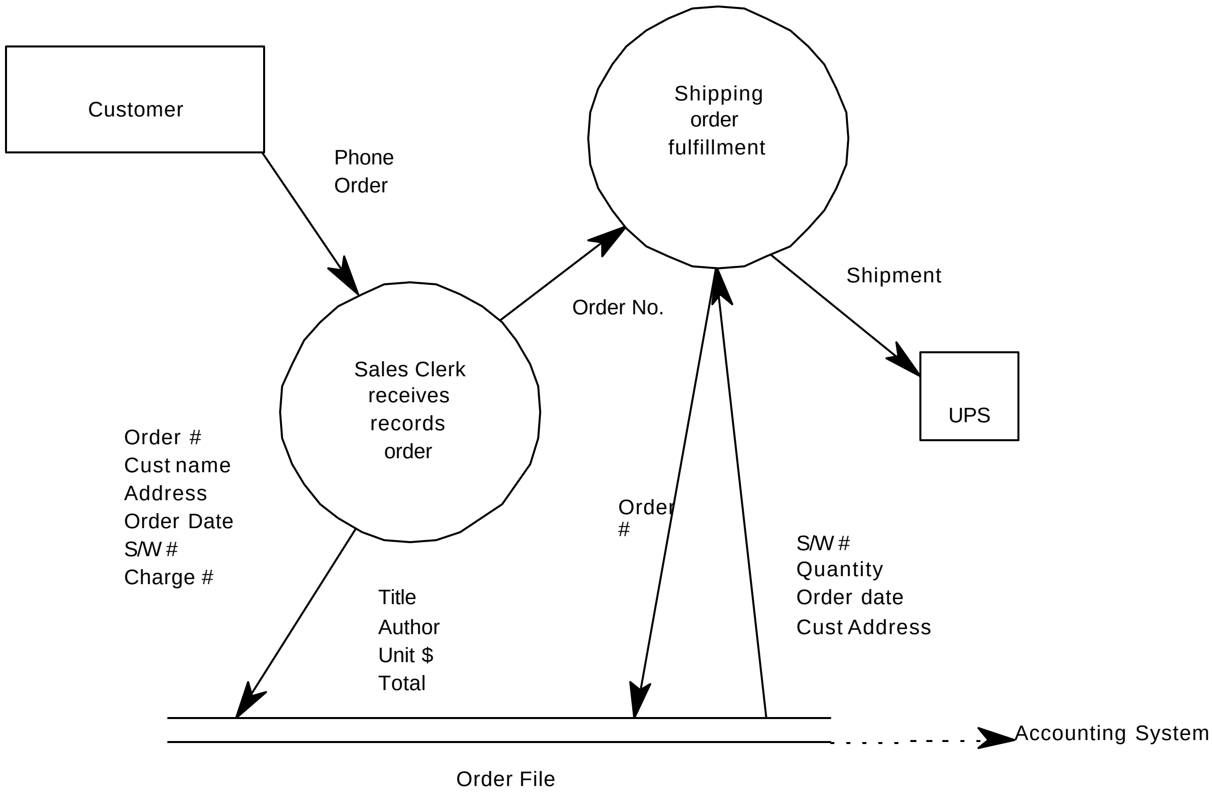
Characteristics:

- Mechanisms for analysis
- Represent functions and/or behaviors
- Define interfaces
- Partition the problem
- Support abstraction
- Represent essential and implementation views

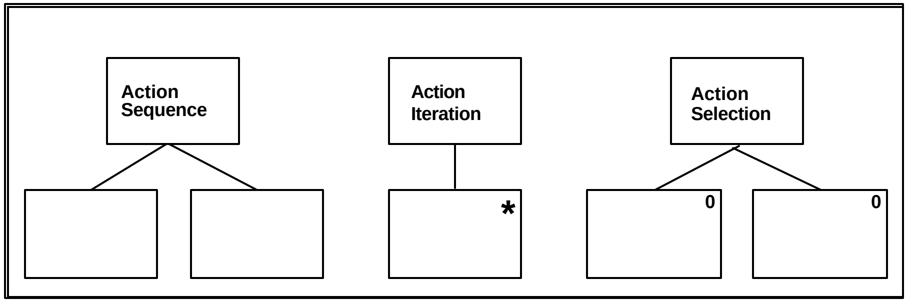
Data Structure Oriented Analysis

- Common characteristics:
  - Identify key information objects
  - Assures hierarchical information structure
  - Allows mapping the data into a program
- Warnier-Orr Method (DSSD)
- Jackson System Development (JSD)
- Object Oriented Requirements Analysis

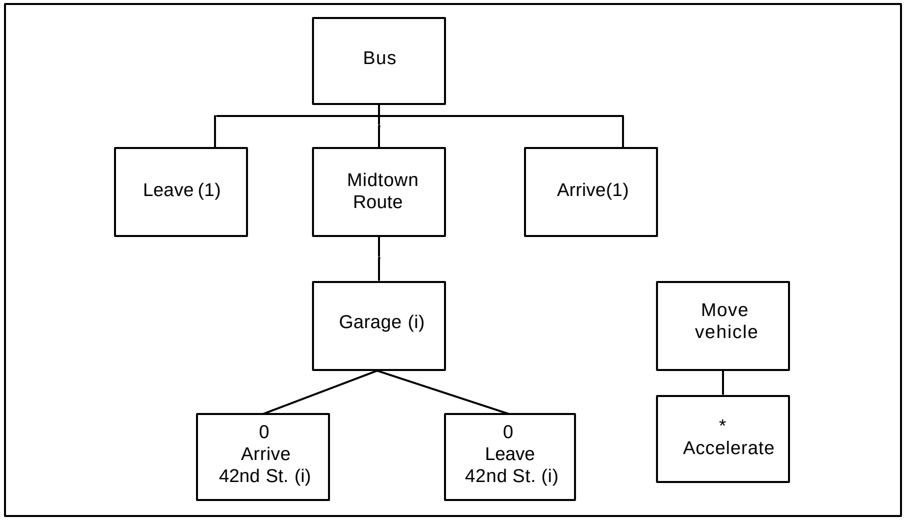
Data Structure Systems Development (DSSD) a.k.a. Warnier-Orr Method



Jackson System Development (JSD)



Notation



Illustration

Approaches for Object Oriented Requirements Analysis

- Categorical
- Behavioral
- Domain
- Use-Case
- Textual
- Structured

Requirements Specification Process

- Systems and Software Engineering
- Iterative process
- Top down synthesis
- Functions and behaviors uncovered, analyzed, and allocated to components
- Defines the scope (limits of the product)
- Trade-offs
- Becomes the System Specification

Requirements Trade-off Criteria

- Project considerations
- Business considerations
- Technical analysis
- Manufacturing evaluation
- Human considerations
- Environmental interfaces
- Off-the-shelf solutions

Systems Requirements Specification*

- Functions and capabilities
- Business, organizational and user requirements
- Safety, security, and ergonomic requirements
- Operational, interface and maintenance requirements
- Constraints
- Qualification requirements

* IEEE 1233 Guide
IEEE 1220 Process
ISO/IEC 12207 Life Cycles

Software Engineering Processes

- Analysis, design, and development methods and tools
  - Software design methods
  - Types of software reuse
  - Clean room and other formal methods
  - Software development tools

Information Domain

- Information flow
  - Transformations
  - Introduced data
  - Output data
- Information content
  - Individual data and control items
  - Combined into records
- Information structure and relationships

Models

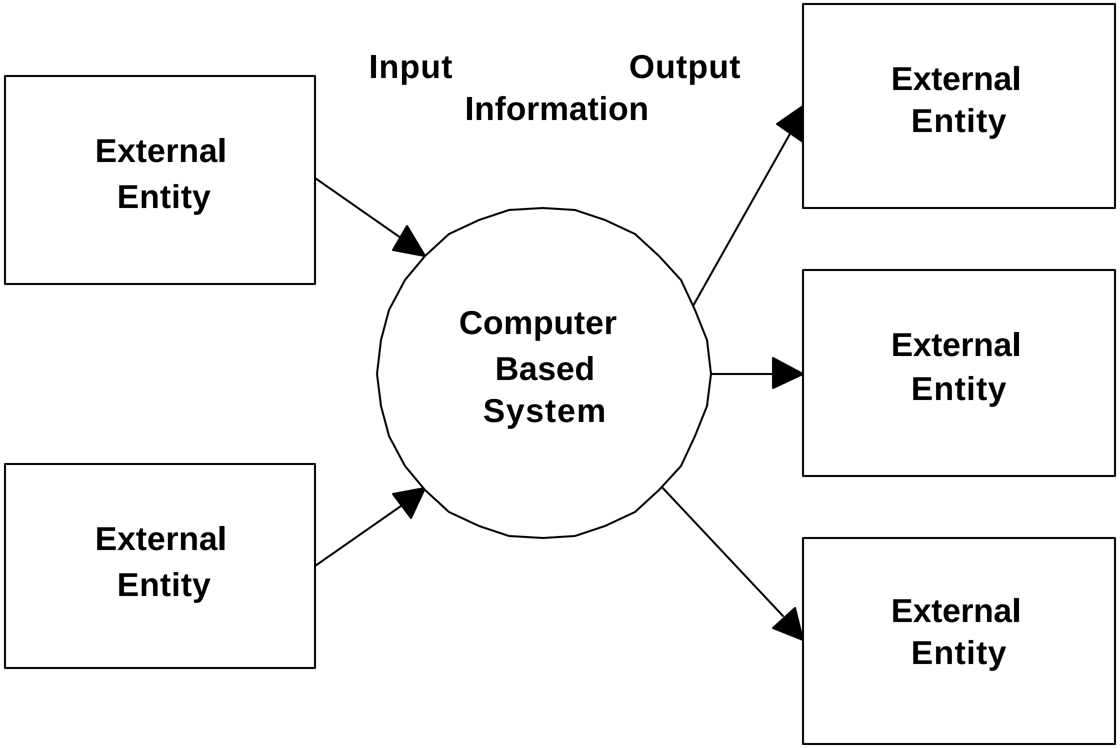
- Aid in understanding system
  - Information
  - Functions
  - Behavior
- Become the focal point for review and testing
- Are the foundation for the design

(Pressman, 1993)

Object Oriented Design

- Find the Classes and Objects
- Identify Structures
- Identify Subjects
- Define Attributes
- Define Services

Structured Analysis and Design



Unified Modeling Language (UML)
Four Layer Architecture

Layer	Description	Example
meta-metamodel	The infrastructure for a metamodeling architecture. Defines the language for specifying metamodels.	*MetaClass, MetaAttribute, MetaOperation*
metamodel	An instance of a meta-metamodel. Defines the language for specifying a model.	*Class, Attribute, Operation, Component*
model	An instance of a metamodel. Defines a language to describe an information domain.	*StockShare, askPrice, sellLimitOrder, StockQuoteServer*
user objects (user data)	An instance of a model. Defines a specific information domain.	*<Acme_Software_Share_98789>, 654.56, sell_limit_order, <Stock_Quote*_Svr_32123>

UML Representations

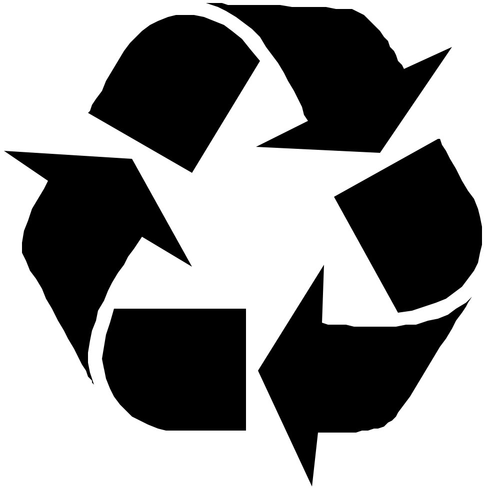
- Static Diagrams; classes and objects
- Use-Case Diagram; systems functions and actors
- Sequence Diagrams; sequential interactions and messages
- Collaboration Diagram; objects' interactions that together perform a function non-sequentially
- Statechart Diagram; the state machine
- Activity Diagram; a state machine where states are activities, and operations and transitions are triggered on activity completion (a.k.a "swimlane diagram")
- Implementation Diagram; component source code structure and runtime processing descriptions

UML Perspectives

- Logical View; the structure of the model (for analysts and designers)
- Process View; the performance, scalability, and throughput (for system integrators)
- Implementation View; the software components (for programmers)
- Use-Case View; the system functionality (for end-users)
- Deployment View; the system topology and installation (for system engineers)

Software Reuse

- Development activity
- Components designed for more than one use
- Benefits
  - Cost savings
  - Reliability
  - Efficiency



Software Reengineering

- Maintaining Functionality
- Inventory Analysis
- Documentation Analysis
- Code Restructuring
- Data Restructuring
- Reverse Engineering

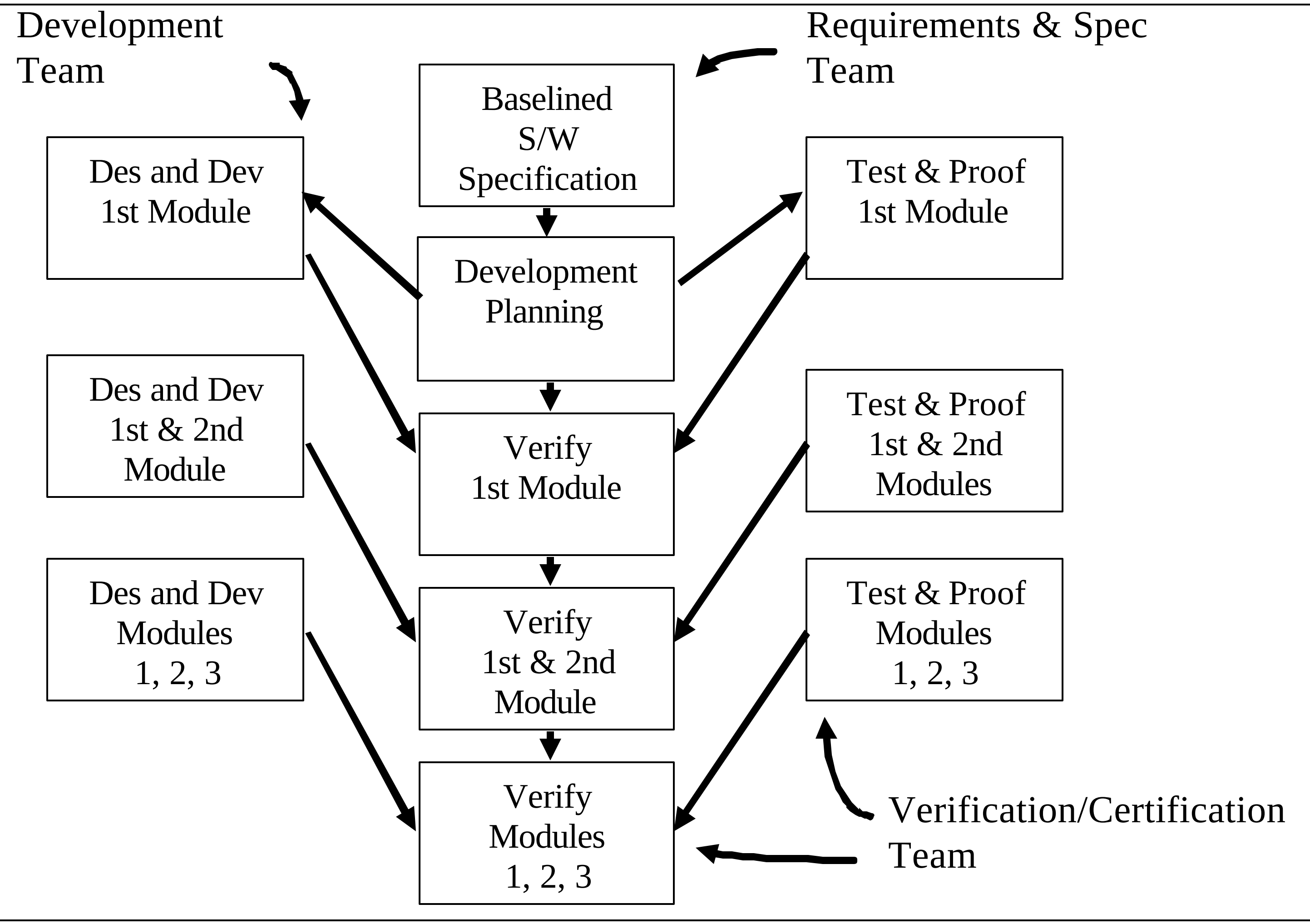
Reverse Engineering

- Design Recovery
  - Code and Data structures
  - Code and Data flows
  - E-R Models
- Re-implementation

Defect Prevention

- Use Standards
  - Life Cycle Models
  - CM
  - Documentation
  - Testing

Classical Cleanroom



Modified Cleanroom

- Explicit component and functional requirements
- Explicit detailed design
- Explicit correctness arguments for design
- Formal inspections of code

Defect Removal

- Detection plus Repair
- Testing
- Reviews
  - Peer or independent
  - Formal or informal
  - Document or code

Technical Reviews

- Three to five people
- Advance preparation
- Maximum of two hour sessions
- Peers only

Development Tools

- Management
- Modeling
- Design
- Code analysis and Testing
- Documentation
- Databases

Maintenance Processes

- Steps
  - Defect report
  - Impact analysis
  - Implement the fix
  - Test
  - Issue revisions
- Types
  - Corrective
  - Adaptive
  - Perfective
  - Preventative

Maintainability

- Factors that control maintainability
  - Qualified staff
  - System structure
  - System handling (CM)
  - Standards use
  - Structure of documentation
  - Test cases
- Side effects
  - Code
  - Data
  - Documentation

Operational Maintenance

- Framework
  - User
  - Environment
  - Maintenance processes
  - Software product
  - Personnel

Maintenance Documents

- Requirements
- System architecture
- Program architectures
- Specifications and designs
- Source code
- Validation documents
- Maintenance guide

Maintenance Factors

- System documentation
- "Lehman's Laws"
  - Continuing change
  - Increasing complexity
  - Large program evolution
  - Organizational stability
  - Conservation of familiarity
- Program understandability
- Maintenance personnel
- Tools

Maintenance Tools

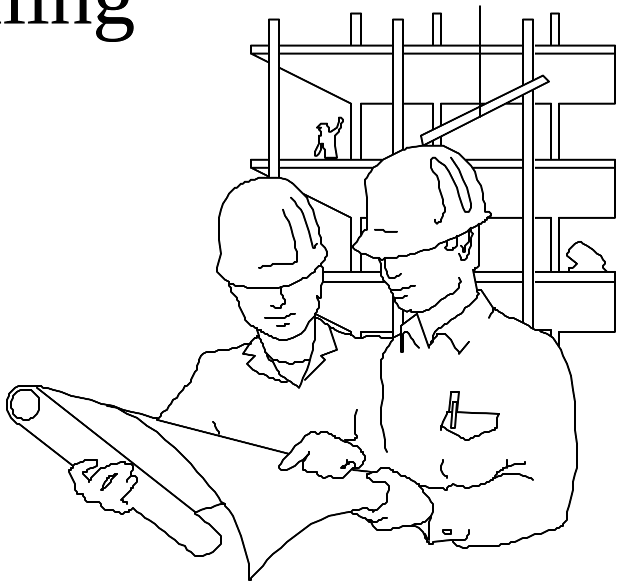
- Reverse Engineering
  - Program slicer
  - Static analyzer
  - Dynamic analyzer
  - Data flow analyzer
  - Cross-referencer
  - Dependency analyzer
  - Transformation tool
- Testing
  - Simulator
  - Test case generator
  - Test path generators
- Other
  - CM
  - Documentation
  - Complexity assessment

Project Management

- General Knowledge, Conduct, and Ethics
- Software Quality Management
- Software Engineering Processes
- **Program and Project Management**
- Software Metrics, Measurement, and Analytical Methods
- Software Verification and Validation (V&V)
- Software Configuration Management

Program and Project Management

- Planning
- Tracking and Controlling
- Risk Management



Program and Project Management

- Planning
  - Project planning elements
  - Goal-setting and deployment
  - Project planning tools
  - Cost and value data

Planning

- Project Plan document
- Project planning factors
- Project planning methods and tools
- Goal setting and deployment factors
- Maintenance types
- Software maintenance and adaptability program planning
- Supplier management methodologies

Contract Review

- Scope
- Standards
- Deliverables
- Risks and Contingencies
- Proprietary information
- Resources identified
- Subcontractor requirements

Contract Review (cont)

- Customer interfaces
  - Design
  - Test
  - Acceptance
- Security
- Acceptance criteria
- Change process
- Training

Project Plan (SMP)

- Objectives and scope
- Contracts and standards
- Technical constraints
- Issues and alternatives
- WBS
- Estimates

Project Planning Elements

- Project Definition
- Scope Management
- Estimation - Forecasting
- Definition Phase
- Customer Goals
- Contract Review
- Software Management Plan

Project Initiation

- Contract
- Request for quote/proposal
- Feasibility study
- R & D
- Customer request
- Business need
- Strategic need

Project Factors

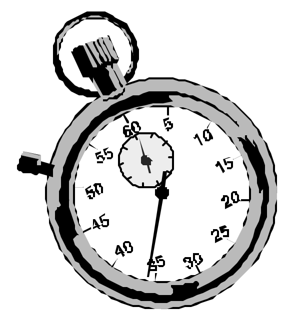
- Work to be done
- Risks
- Required resources
- Tasks
- Milestones
- Effort required
- Schedule

Goal-setting and Deployment

- Elements with goals
  - Size, cost, schedule, task duration
  - Critical project resources
  - Product quality
  - Software technical activities
  - Measures and metrics
- Goal Setting
- Goal Deployment

Estimation Elements

- Effort
- Resources
- Scheduled deliveries



Estimation Means

- Analogy
- Engineering assessment
- Subject Matter Experts (SME)
- Parametric Modeling

Parametric Modeling Tools

- COCOMO
- Before You Leap (BYL)
- Wang Institute Cost Model (WICOMO)
- DEC Plan
- SLIM
- Checkpoint

Scheduling Projects

- Critical path scheduling
- Resource availability
- Task cost estimates
- Task crashing times
- Task crashing costs

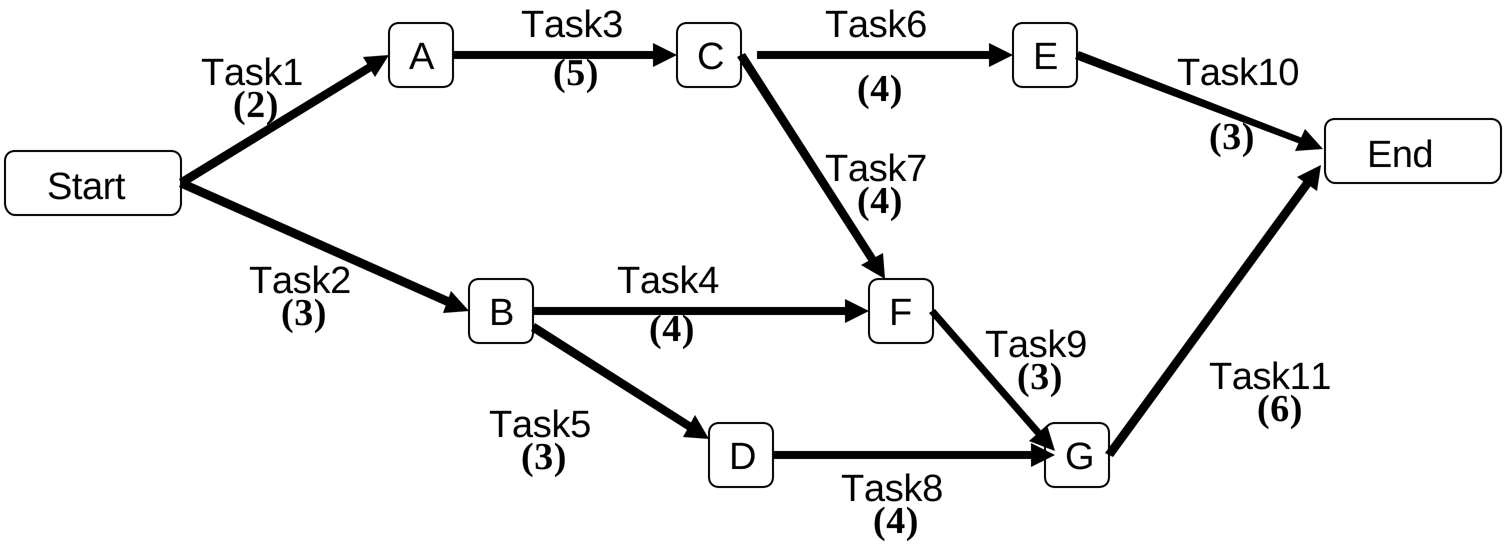
Software Project Management

- Work Breakdown Structure
- Phase Based Software Activities
- Planned Roles and Activities

Milestones

- Events
- Entry and exit criteria
- Correspond with phase transitions
- Key for process control

PERT Example

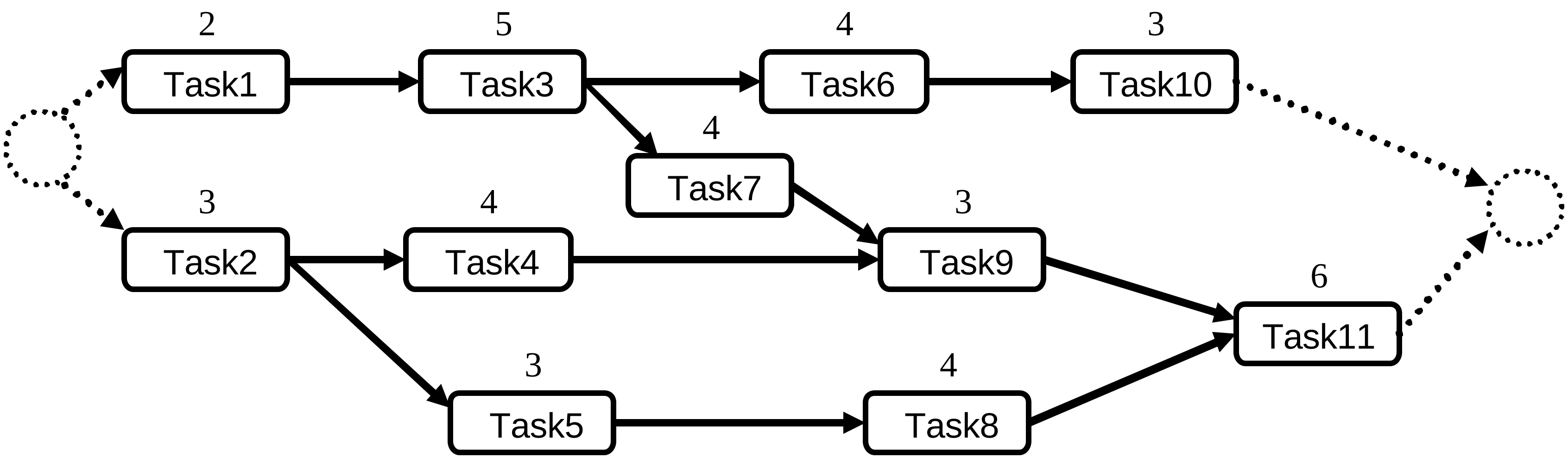


Program Evaluation and Review Technique

PERT / CPM Example

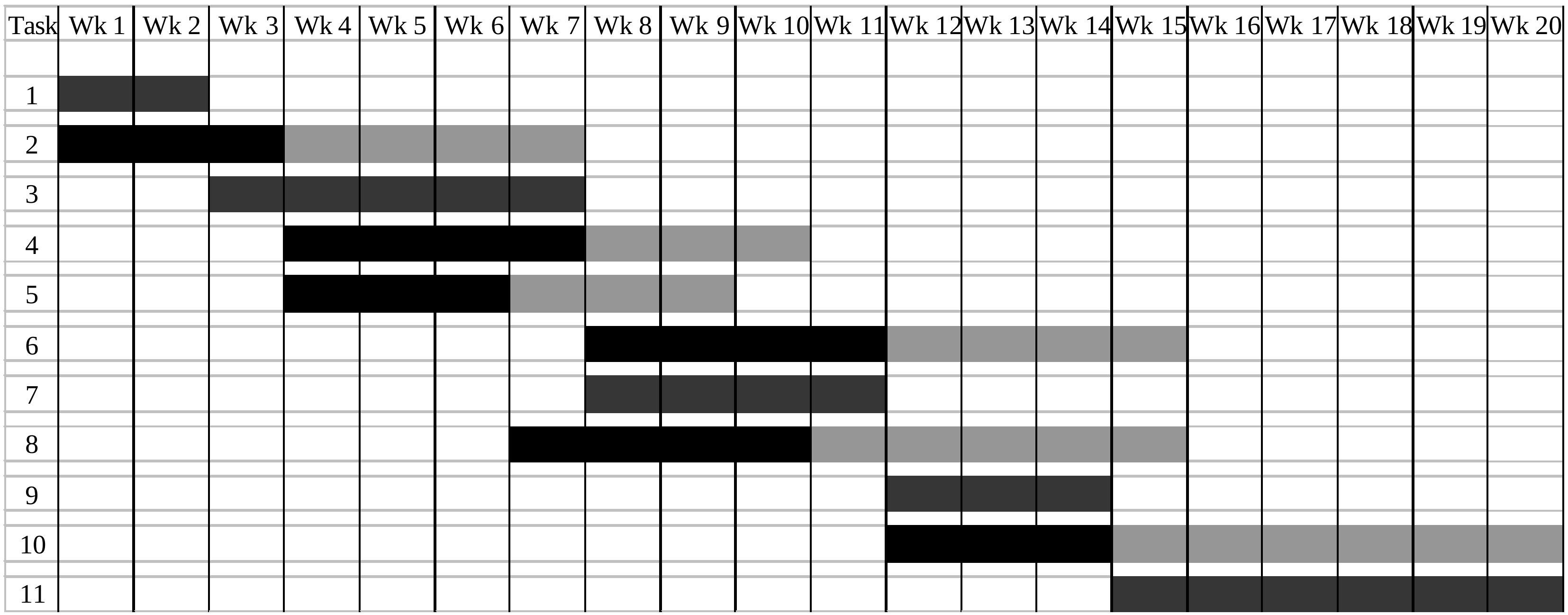
Task	Time	Early	Late
Start	-	-	-
1	2	0	0
2	3	0	4
3	5	2	2
4	4	3	7
5	3	3	7
6	4	7	13
7	4	7	7
8	4	6	10
9	3	11	11
10	3	11	17
11	6	14	14
End	-	20	20

PERT/CPM



Critical Path Method

Gantt Chart Example



Work Breakdown Structure

- Estimate size
- Decompose tasks
- Identify dependencies
- Build vs. Buy

Project Sizing Using Function Points

- Counting ‘Externals’
  - Inputs (X4)
  - Outputs (X5)
  - Inquires (X4)
  - Interfaces (X7)
- Add count of internal [logical] files (X10)
- Summed values adjusted by experience
- All values weighted by complexity

Cost and Value Data

- Resource planning
- Cost estimation
- Cost budgeting
- Cost Control

Resource Planning

- Input data:
  - Task duration from WBS
  - Project scope statement
  - Activity duration estimates
- Output data:
  - Resource requirements
    - People
    - Platforms
    - Equipment
    - Materials

Cost Estimation

- Input data:
  - WBS
  - Resource requirements
  - Historical data and information
  - Resource rates
  - Risks considered for cost estimates
- Output data:
  - Cost estimates (as cost and schedule)

Cost Budgeting

- Input data:
  - Cost estimates
  - WBS
  - Project schedule
  - Risk management plan

- Output data:
  - Cost baseline

Cost Control Process

- Manage to the budget
- Procedures for changing cost baseline
  - Cost tracking system
  - Approvals for authorizing changes
- Performance measurement techniques
  - Assess the magnitude of cost variations
  - Earned value measures

Earned Value Management

- Planned value
  - Physical work schedule to be performed
  - Estimated value of the work
- Earned value
  - Physical work accomplished
  - Estimated value of the work
- Actual costs
  - Amount of resources consumed

Project Progress

- Phase transitioning control techniques
- Collecting ‘Cost of Quality’ data
- ‘Cost of Quality’ categories
- Cost, progress, and deliverable tracking

Tracking and Controlling

- Phase transition control techniques
- Reporting and interpreting Cost of Quality (COQ) data
- Tracking elements and methods
- Project reviews

Tracking and Monitoring

- Project Plan
- Management approval for external commitments
- Internal and external change management
- Product quality
- Reviews of status, plans, performance and issues
- Formal reviews at milestones
- Tracing and control of
  - Size, cost, schedule
  - Computer resource
  - Technical activities
  - Software technical, cost, resource, schedule, and quality risks
  - Measures, metrics and replanning data
- Post-release lessons learned review

Phase Transition Controls

- PERT
- CPM
- WBS
- Schedule

Scheduling Factors

- Development environment
- Programmer skill levels
- Language and compiler stability
- Adjunct resources (e.g., test beds)
- Schedule commitments
- SDLC
- Corporate culture

Budgeting

- BCWP
- ACWP
- BCWS
- Earned Value Analysis

Earned Value Analysis Computations

	Scheduled Completion	Actual Completion	Budgeted Cost	Actual Cost
Task 1				
Task 2				
Task 3				
Task 4				
…				
Task *n*				
	<count to date>	<count to date>	<cost to date>	<cost to date>

Earned Values

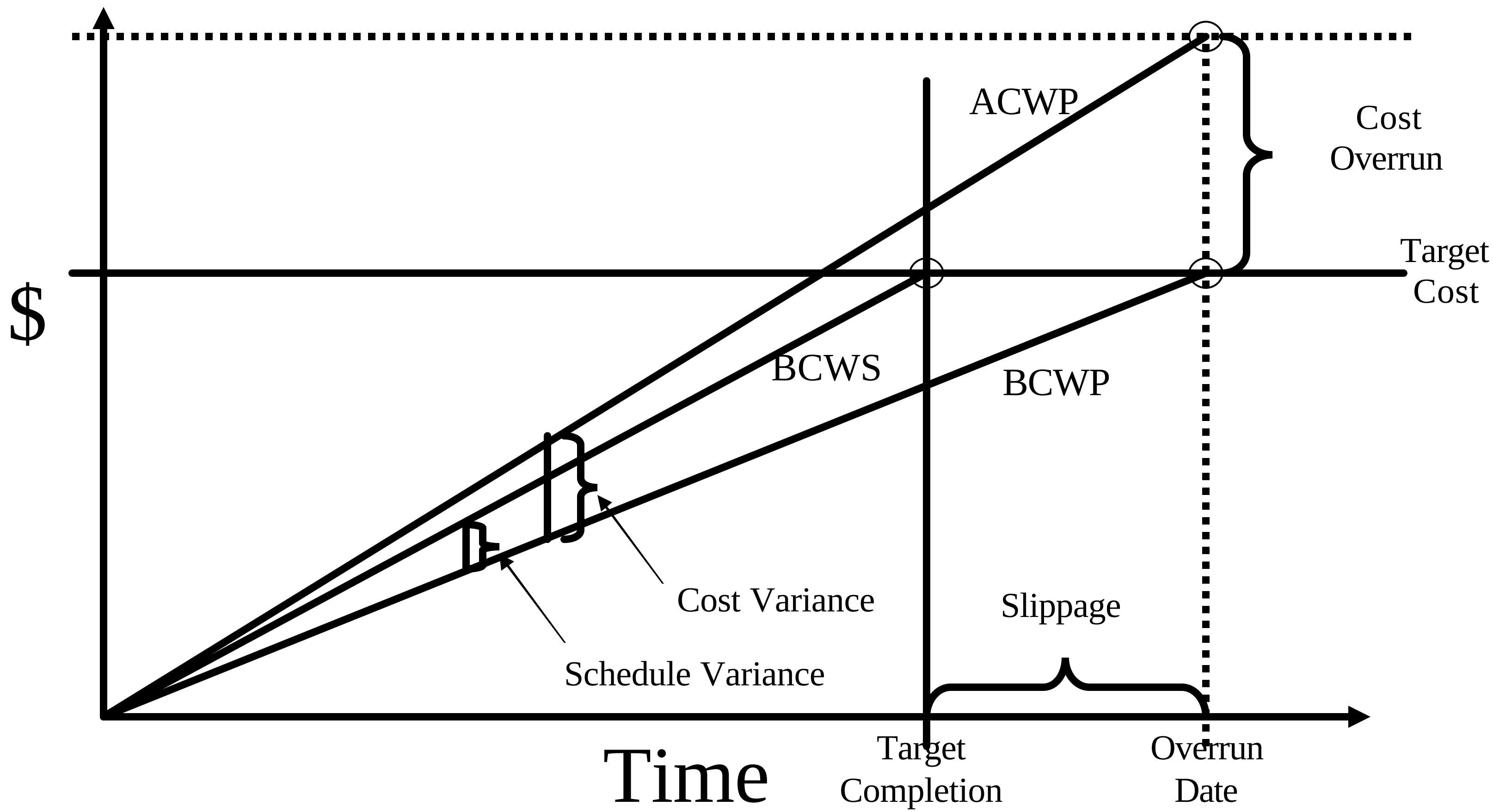
Cost Variance = BCWP – ACWP

Schedule Variance = BCWP – BCWS

Cost Performance Index = BCWP/ACWP

Earned Value = BCWP

Earned Value Analysis



Other Cost Metrics

- Planned Value (PV)
- Earned Value (EV)
- Actual Cost (AC)
- Budgeted [cost] at Completion (BC)
- Latest Revised Estimate (LRE)

- Schedule Variance = EV -PV
- Cost Variance = EV - AC
- Schedule Performance Indicator = EV/PV
- Cost Performance Indicator = EV/AC

Quality Cost Responsibilities

- QA
- Accounting
- Management
- General Manager

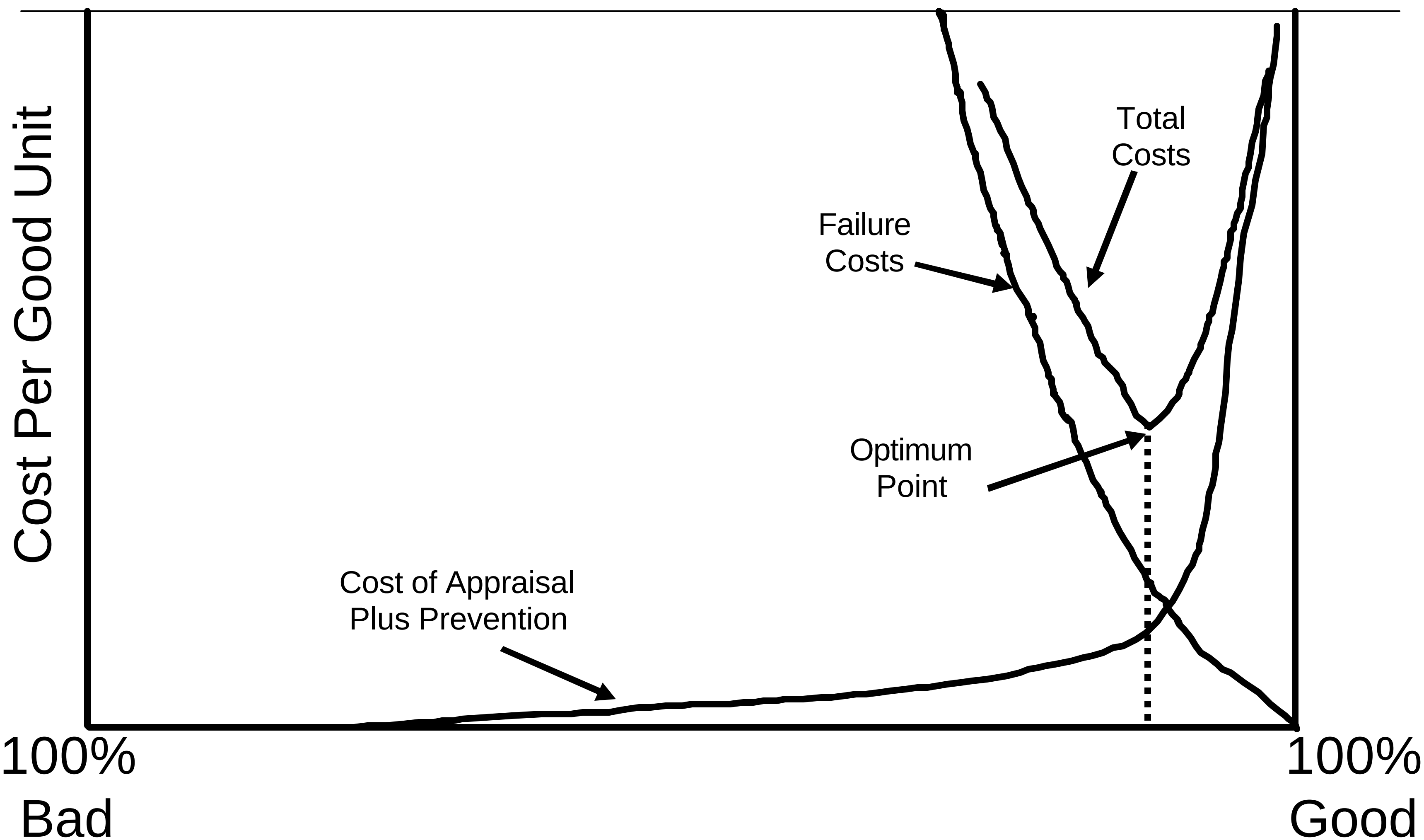
Quality Cost Improvement

- Identify categories
- Collect costs
- Oversee accuracy
- Analyze data

Quality Cost Program

- Budgeting
- Cost Categories
- Advantages/Limitations
- Optimum Costs

Optimum Quality Costs



Project Reviews

- Senior Management Review
- Project Status Review
- Closed Loop Methodologies
  - Project Plan updates
  - Corrective action plans
  - Lessons learned

Implementation

- Project management tools
- Reporting ‘Cost of Quality’ data
- Trade-offs in release decisions

Risk Management

- Risk management planning methods
- Risk probability
- Product release decisions
- Software security, safety, and hazard analysis issues

Risk Management Planning

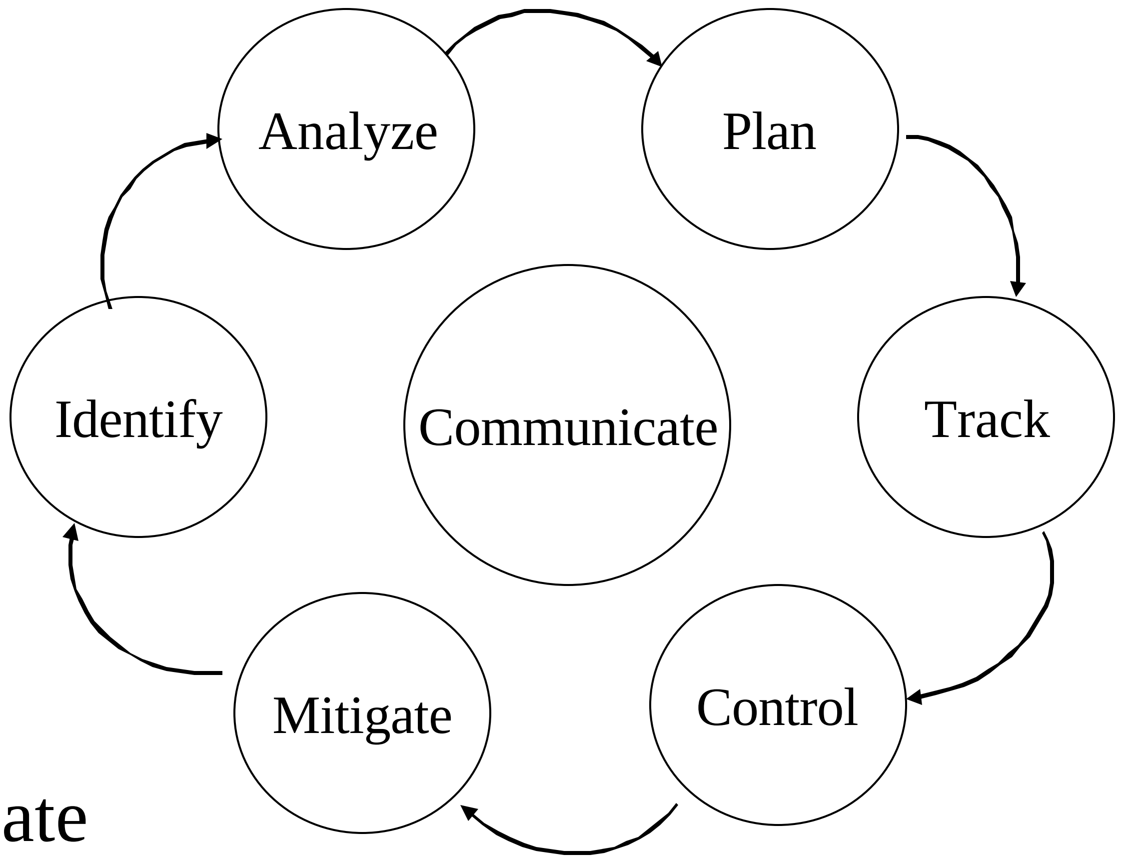
- Project risks
  - Product performance
  - Actual costs higher than budgeted
  - Late product delivery
- Risk areas
  - Technical (performance)
  - Supportability (performance)
  - Programmatic (performance)
  - Cost
  - Schedule

Risk Management

- Features - Costs - Schedule
- Uncertainties
  - Requirements
  - People
  - Metrics Data
  - Subjective values
  - Model correctness
- Risk management process

Continuous Risk Management

- Identify
- Analyze
- Plan
- Track
- Control
- Mitigate
- Communicate



Uncertainty in Quantification

- Product requirements
- Variability in personal performance
- Inaccuracies in measurements
- Judgement variations

Risk Probabilities

- Components of risk
  - An [unwanted] event
  - A likelihood of occurrence
  - The impact of the event
  - Ability to notice the event
- Risk identification
- Qualitative risk analysis

Risk Response - Contingency Plans

- Avoidance: action to eliminate the risk
- Transference: shift the consequence elsewhere
- Mitigation: reduce the likelihood or consequence
- Acceptance: decision not to change

Product Release Decisions

- Decision attributes
  - Content (functionality)
  - Schedule
  - Quality
- Management team decision
- Release Criteria
  - Identified deliverables
  - Documents reviewed
  - Testing complete with acceptable pass rate
  - Regression testing complete
  - Prioritized defects resolved
  - Planned release criteria all met
  - Risks identified and mitigated
  - Exceptions to plans are resolved
  - Customer notified
  - Product Support plan in place
  - Quality criteria have been met

Project Corrective Actions

- Factors requiring trade-off
  - Schedule
  - Cost
  - Quality
  - Customer requirements
  - Functionality
- Schedule compression most common issue
- Typical decisions
  - Outsourcing software development (contractors)
  - Phased deliveries of increasing functionality
  - Phased delivery of customer requirements
  - Delivery with known non-critical defects
  - Re-negotiating cost or schedule

Security, Safety, and Hazards

- Faults
- Authentication
- Safety-Critical software (HACCP)
- Real-time logic
- Petri net models
- Fault tree analysis

Common Security Faults

- Programmer errors
- Input Overflows
- Fault prevention
- Syntax checking flaws
- Making a function too general
- Documentation
  - Lapses
  - Wrong
  - Missing
  - Confusing
  - Faulty
- Lack of security awareness
- Easter Eggs
- Miss-configured security functions
- Default passwords

Metrics and Measurement

- General Knowledge, Conduct, and Ethics
- Software Quality Management
- Software Engineering Processes
- Program and Project Management
- **Software Metrics, Measurement, and Analytical Methods**
- Software Verification and Validation (V&V)
- Software Configuration Management

Background for Software Metrics

- Definitions
- Process and product measurement
- Analytical techniques

Software Metrics Questions

- How much do we need to know before measuring
- How do we know what we’ve measured
- What do the measures really mean
- What arithmetic makes sense

Measurement Theory Definitions

- Entity (object or event)
- Attribute (property of an entity)
- Measurement (assign a value)
- Primitive (measurable entity)
- Measure (value assigned)
- Reliability (repeatability)
- Validity (freedom from bias)
- Measurement Error (systematic and random)
- Representational Condition (model)

Definitions of Metrics and Measures

- Software Metrics
- Defect
- Error
- Failure
- Fault
- KCSI
- KLOC
- KSLOC
- Dependent/Independent Variables

Measurement Theory

- Conditions for measurement
  - Sets
  - Relations
  - Axioms
  - Functions

Types of Measures

<table>
<tr><th>Scale</th><th>Description</th><th>Example</th><th>Center</th><th>Dispersion</th><th>Tests</th></tr>
<tr><td>Nominal</td><td>Names or categories</td><td>Colors</td><td>Mode</td><td>Information only</td><td>Chi-square</td></tr>
<tr><td>Ordinal</td><td>Ordered</td><td>Defect classifications</td><td>Median</td><td>Percentages</td><td>Sign or run test</td></tr>
<tr><td>Interval</td><td>Differences can be measured</td><td>Temperature (F)</td><td>Arithmetic Mean</td><td>Standard or average deviation</td><td rowspan="2">t-test
F test
correlation analysis</td></tr>
<tr><td>Ratio</td><td>Inherent zero start value</td><td>Money</td><td>Geometric or harmonic mean</td><td>Percent variation</td></tr>
</table>

Central Limit Theorem

Population with

- mean μ
- variance σ_x^2
- large number of samples (*n* of them)
- distribution of sample means is normal
- with population mean and variance (μ, σ_x^2)

Measures of Central Tendency

- Mean (X-bar)

  arithmetic average

- Mode

  most frequent

- Median

  middle value

  (when ordered)

Measures of Dispersion

- Range

  largest - smallest

- Variance (σ^2, S^2)

  $\sigma^2 = \sum(X - \mu)^2/N$

  $S^2 = \sum(X - X\text{-bar})^2/(n-1)$

- Standard Deviation

  $\sigma = SQRT\ [\ \sum(X - \mu)^2/N\ ]$

  $S = SQRT\ [\ \sum(X - X\text{-bar})^2/(n-1)\ ]$

- Coefficient of Variation

  COV =(S/X-bar) * 100

Psychology of Metrics

- Potential for problems
- Results based
- Clear objectives
- Process and product focus
- Avoid misuse
- Hawthorne effect

GQM Metric Selection Paradigm *

- Goal

  What you want to learn

- Question

  What answer you need to know

- Metric

  How to measure/ compute it

* Basili and Rombach (1984)

Process and product measurement

- Process, product, and resource metrics
- Commonly used metrics
- Software quality attributes
- Defect detection effectiveness measures
- Program performance and process effectiveness

Designing Measures

- Types of software measures
  - Product
  - Process
  - Resources
- Purpose of metrics
  - Assessment
  - Prediction
- Model is required

Examples of Software Metrics

Product	Process	Resource
Spec	Develop spec	Personnel
Design	Detailed design	Software
Code	Testing	Hardware
Test Data		Work area

Designing Measures

- IEEE Standard 1061

  Software Metrics Methodology

- IEEE Standard 982.1

  Standard Dictionary of Measures

- IEEE Standard 982.2

  Guide to using IEEE 982.1

IEEE 1061 Methodology

- Establish Software Quality Requirements
- Identify Software Quality Metrics
- Implement the Metrics
- Analyze the Metrics Results
- Validate the Metrics

Metrics Examples

- Halstead - Software Science
- Boehm - COCOMO
- Albrecht - Function Points
- McCabe - Cyclomatic Complexity
- De Marco - Bang Metrics
- LOC Measures

Software Quality Attributes

Attribute	Metrics
Correctness	Faults/LOC Faults/Requirements Faults/Standards
Efficiency	Actual or Allocated Utilization
Flexibility	Average labor days to change
Integrity	Faults/Lines Faults relative to security
Inter-operability	Effort to couple or develop
Maintainability	Average labor days to fix
Portability	Effort to transport or to develop
Reliability	Faults/LOC

Tables 7.7 and 7.8 (pp. VII-26 & 27)

Attribute	**Metrics**
Verifiability	Number of implementable requirements
Usability	Average number of errors made by users in a given time
Reusability	Value of resources saved
Testability	Number of requirements with test criteria; number of requirements planned
Expandability or Extendability	Amount of spare capacity available
Performance	Number of transactions per second
Robustness	Time to restart after system failure
Traceability	Number of requirements included in, assigned to, or tested

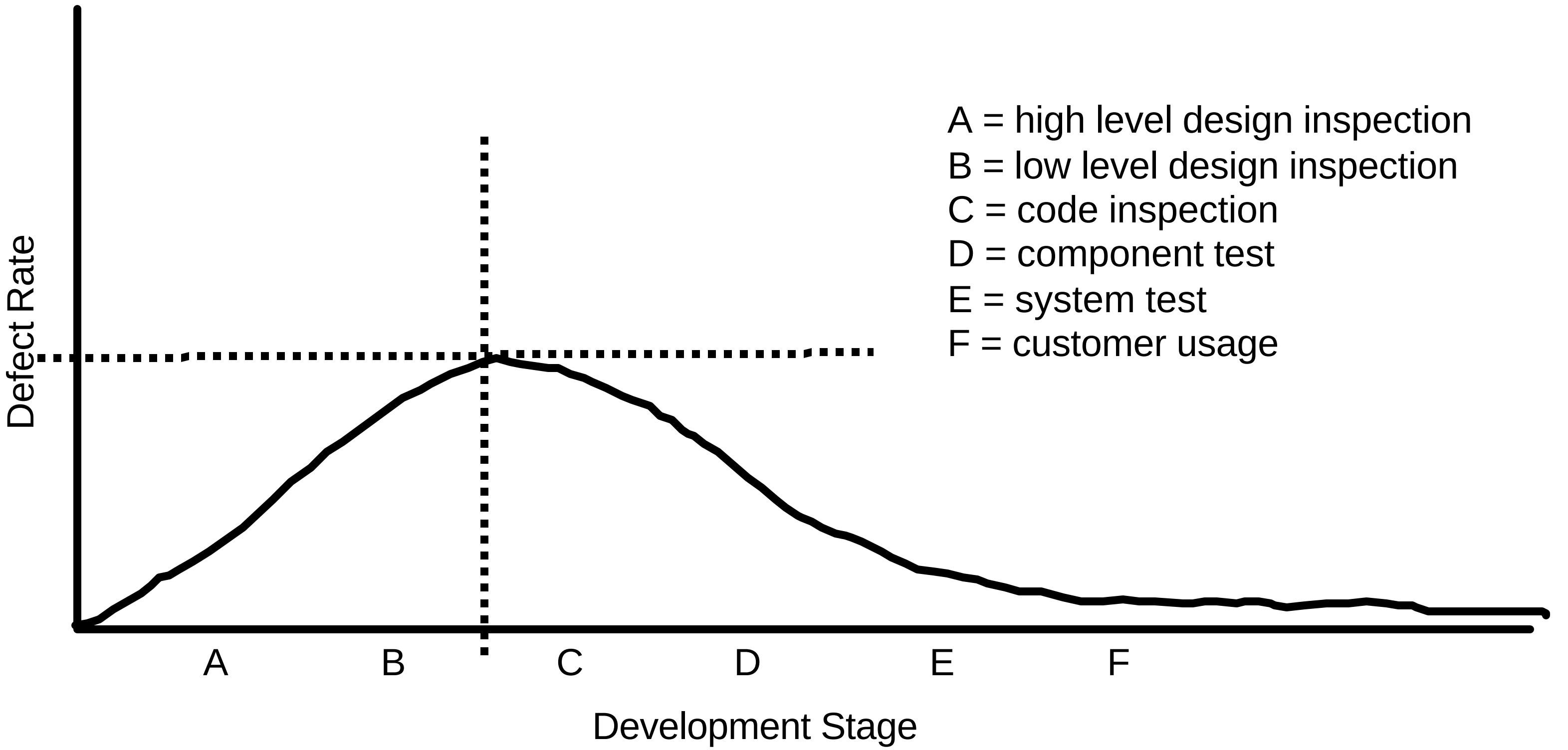
(IEEE 610.12)

General Software Factors*

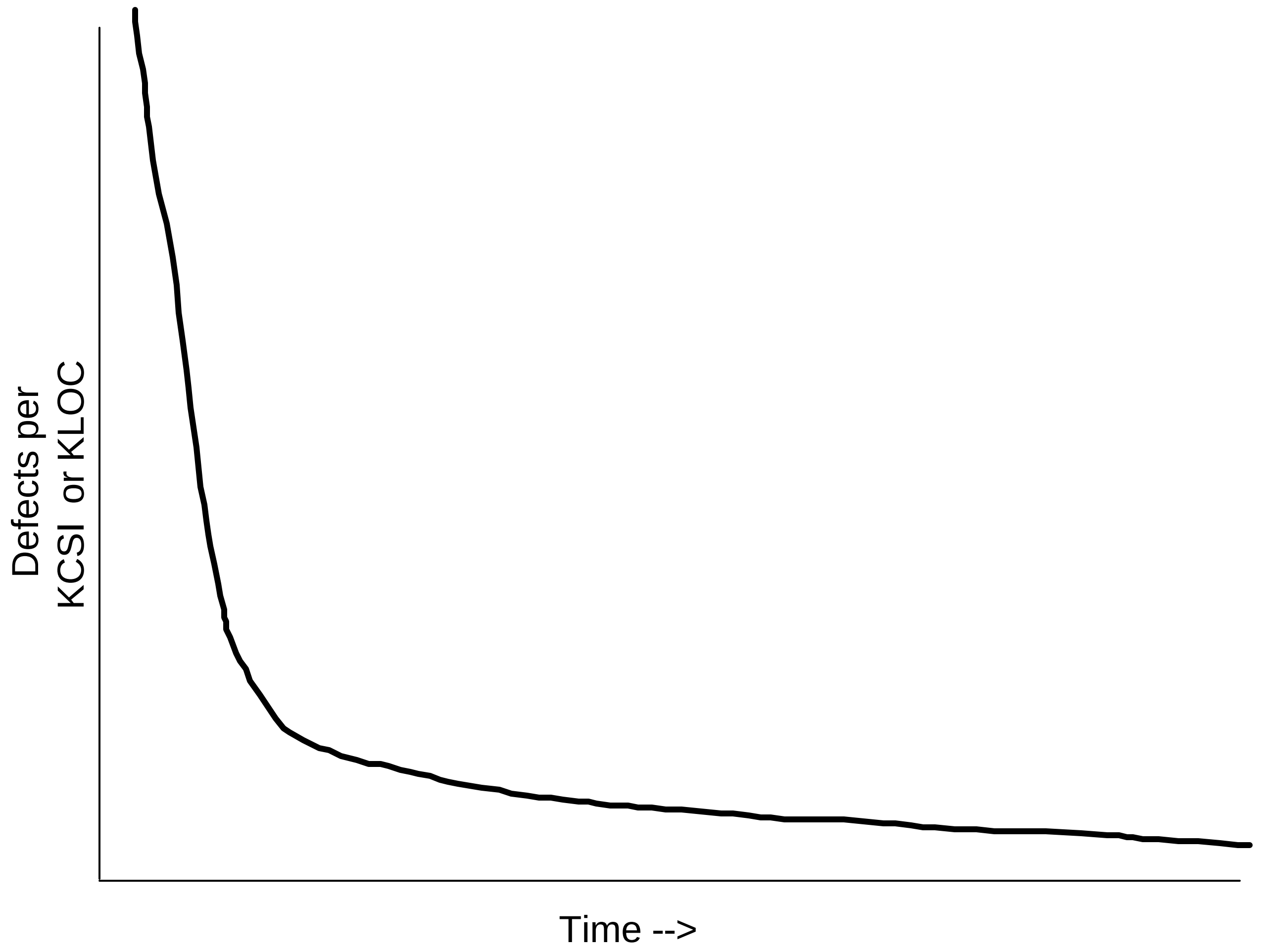
- Correctness
- Reliability
- Efficiency
- Integrity
- Security
- Usability
- Maintainability
- Testability
- Flexibility
- Portability
- Reusability
- Inter-operability
- Understandability

* Capers Jones, 1997

Rayleigh Model



Exponential Distribution



Software Models

- Time Between Failures (TBF)
- Fault Counts (FC)
- Examples
  - Jelinski-Moranda (J-M) (TBF)
  - Littlewood (LW) (TBF)
  - Goel-Okumoto (G-O) (FC)
  - Goel-Okumoto (NHPP) (FC)
  - Musa-Okumoto (FC)
  - Delayed S and Inflection S (FC)

Defect Removal Effectiveness

$$\text{Error detection effectiveness} = \frac{Errors\_found\_by\_inspection}{Total\_errors\_before\_inspection} * 100$$

(Michael Fagan, 1976)

$$\text{Defect detection effectiveness} = \frac{Defects\_found\_by\_operation}{Defects\_present\_at\_operation} * 100$$

$$\text{Defect removal effectiveness} = \frac{Defects\_found}{Defects\_found\_+\_Defects\_not\_found} * 100$$

(Capers Jones, 1991)

Phase Containment

Development Phase	Defect Cause	Defect Detection
Requirements	Requirements gathering Requirements specification	Requirements analysis Requirements review
High level design	Design work	Inspection
Low level design	Design work	Inspection
Coding	Coding	Inspection
Unit test	Bad defect fixes	Unit testing
Integration	Bad defect fixes	Integration testing
System test	Bad defect fixes	System testing

Process Measures

- Steps *
  - Actual vs. Plan
  - Extrapolate
  - Assess impact of problems
  - ID and evaluate alternatives
  - Make decisions
  - Monitor for changes
- Trends
- Thresholds & Targets

* McGarry, 2002

Process Effectiveness

- ISO 9000-2000
  - Procedures followed
  - Measured results
  - Process ownership
  - Management support
  - Incentive alignment
  - Proper training
  - Feedback encouraged
  - Improvement results
  - Uses technology
- CMM/CMMI Level
- Measurements
  - Performance
  - Reliability
  - Usability
  - Cost of process
  - Cost of Quality
  - Responsiveness
  - Adaptability
  - Dependability
  - Applicability
  - Accuracy
  - Quality of products

Metrics Data Integrity

- Data correctness
  - Collection
  - Analysis
  - Comparisons
- Data errors
- Data management
- Data quality
  - Bias
  - Rounding
  - Sequence
  - Timeliness
  - Population statistics
  - Entry errors
  - Hunches
  - Environment

Software Data Error Types*

- Calculation error
- Blank field
- Transfer data (prior project)
- Entry error
- Transfer data (this project)
- Impossible values
- Process sequence not followed

* Disney, 1998

Seven Quality Analysis Tools*

- Check Sheet
- Pareto Diagram
- Histogram
- Scatter Diagram
- Graph
- Control Chart
- Cause and Effect Diagram

* Ishikawa, 1991

Check Sheet (Checklist)

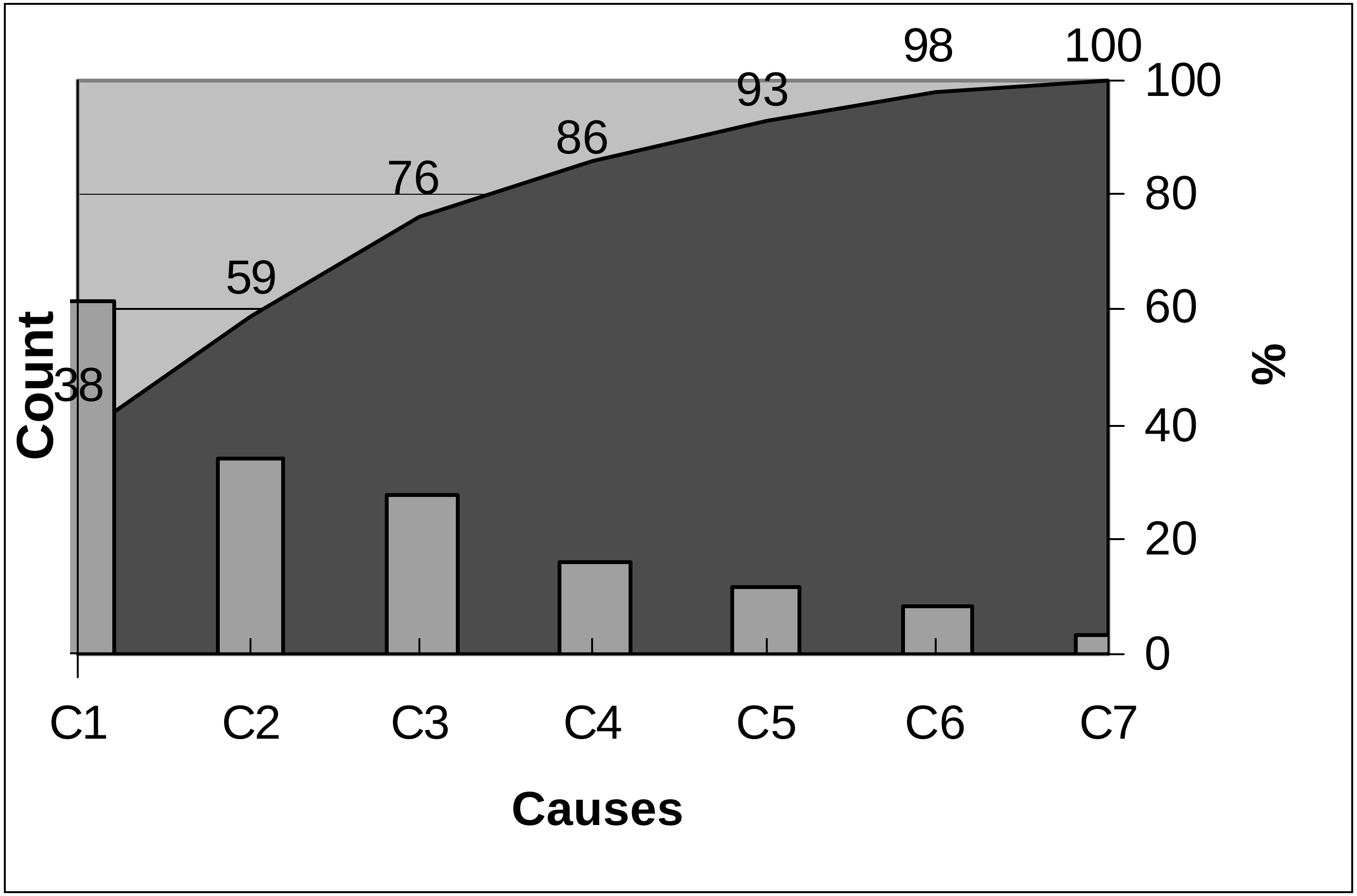
TEST CASE DOCUMENTATION

- Run instructions complete, concise, and conforms to standards
- Evaluation instructions complete, concise, and conforms to standards
- All instructions provided in an on-line file
- Installation instructions complete, concise, and conforms to standards
- Special instructions complete, concise, and conforms to standards
- Configuration Management instructions complete, concise, and conforms to standards
- Tests are traceable to a test design specification.

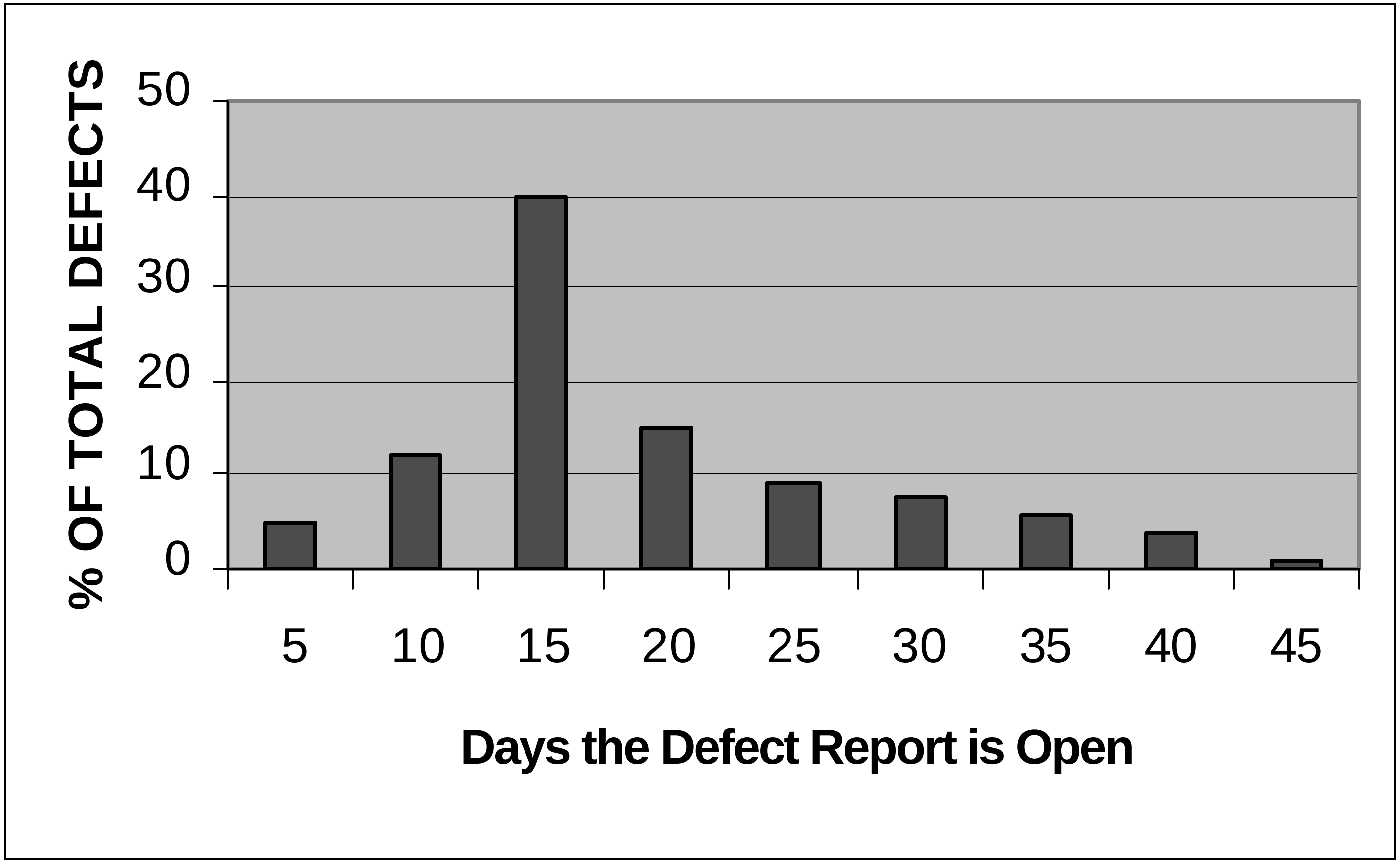
11/1/00

Keith Stobie - Testware Evaluation

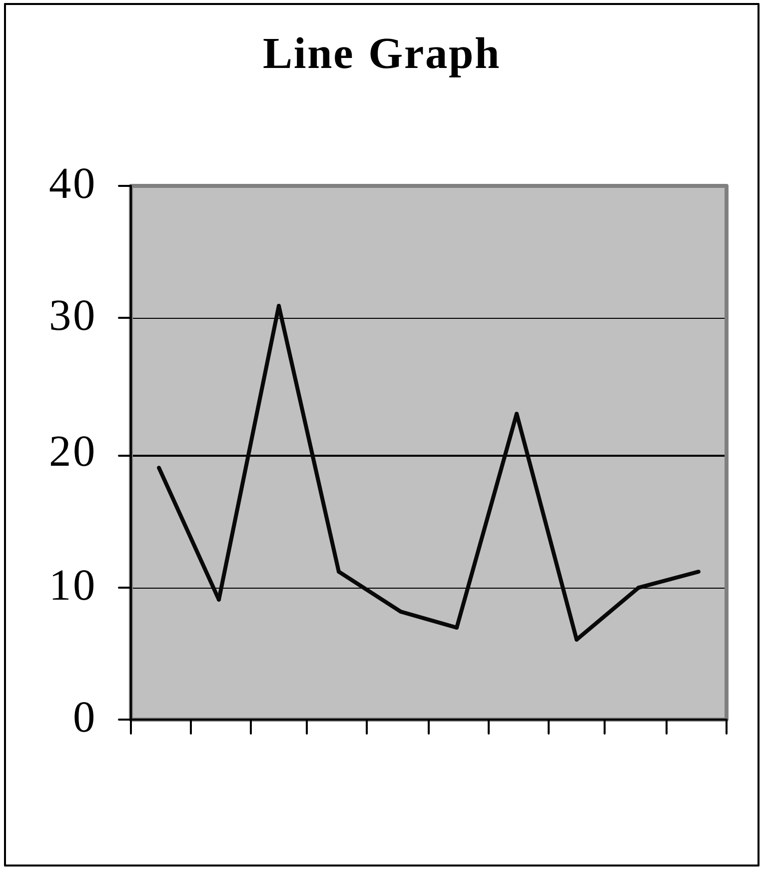
Pareto Diagram

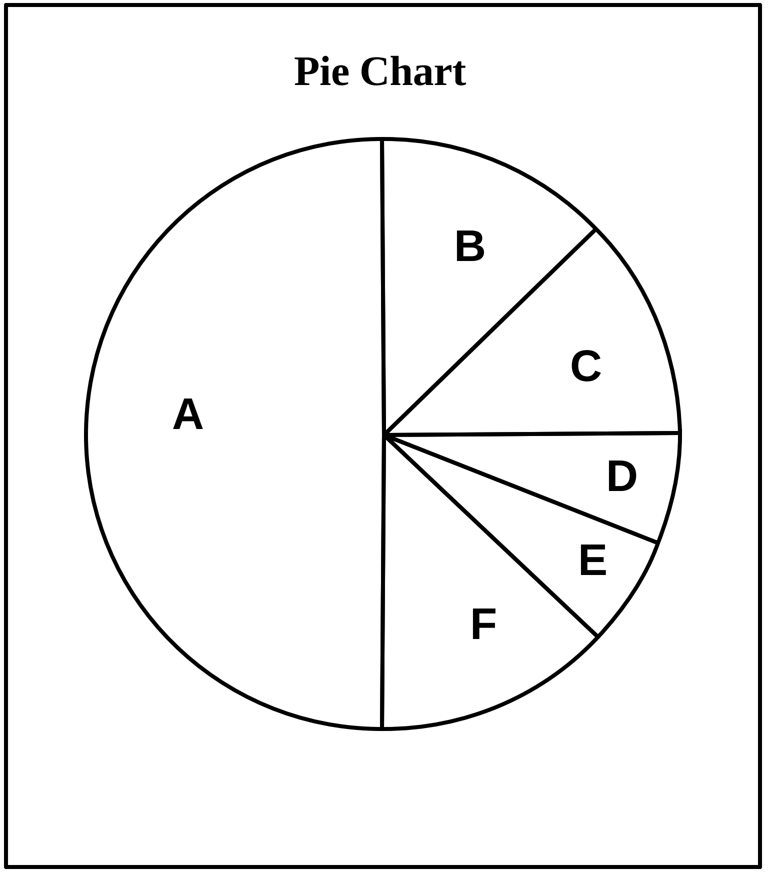


Histogram

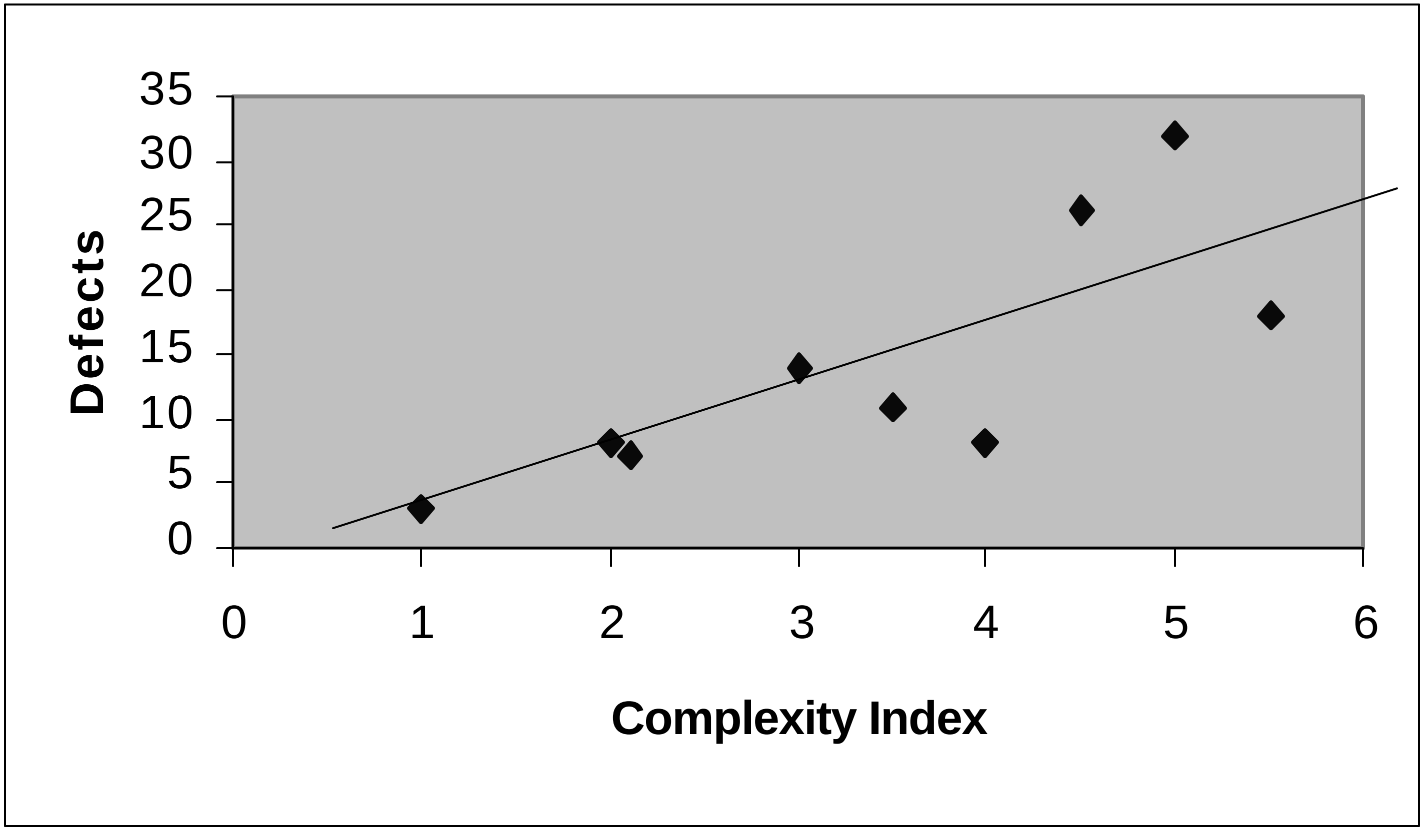

% OF TOTAL DEFECTS
50
40
30
20
10
0
5
10
15
20
25
30
35
40
45
Days the Defect Report is Open

Graphs

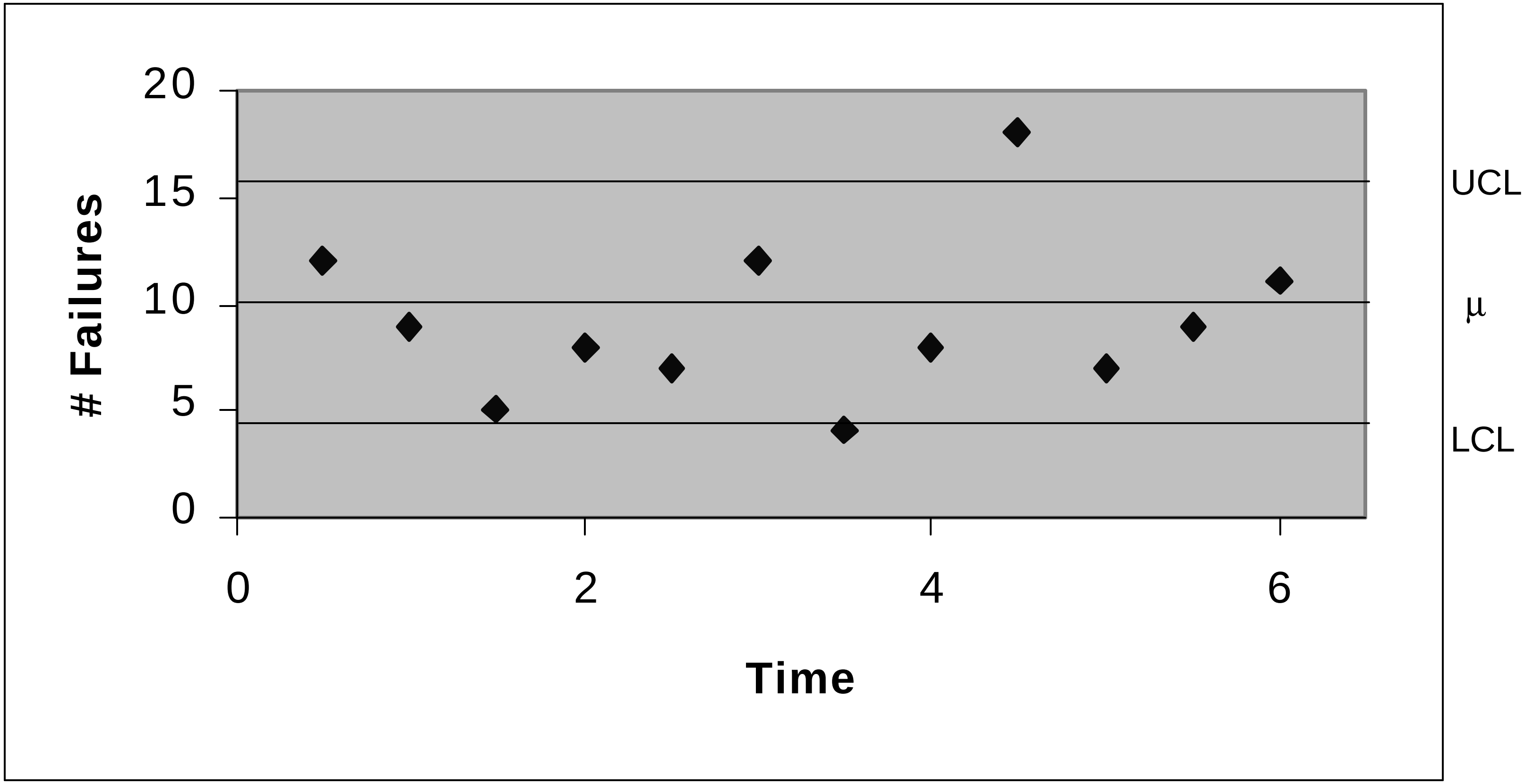




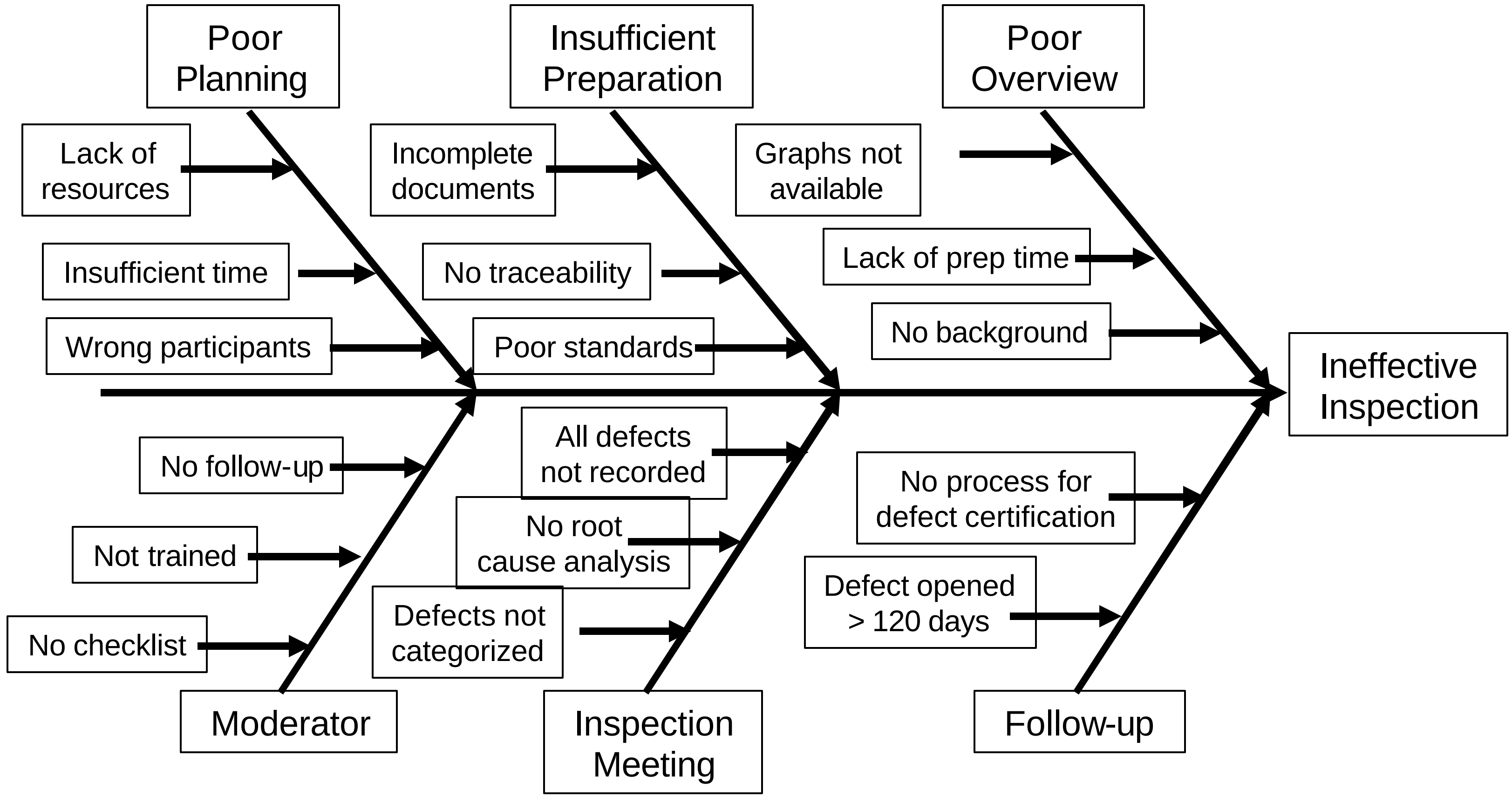
Scatter Diagram



Control Chart



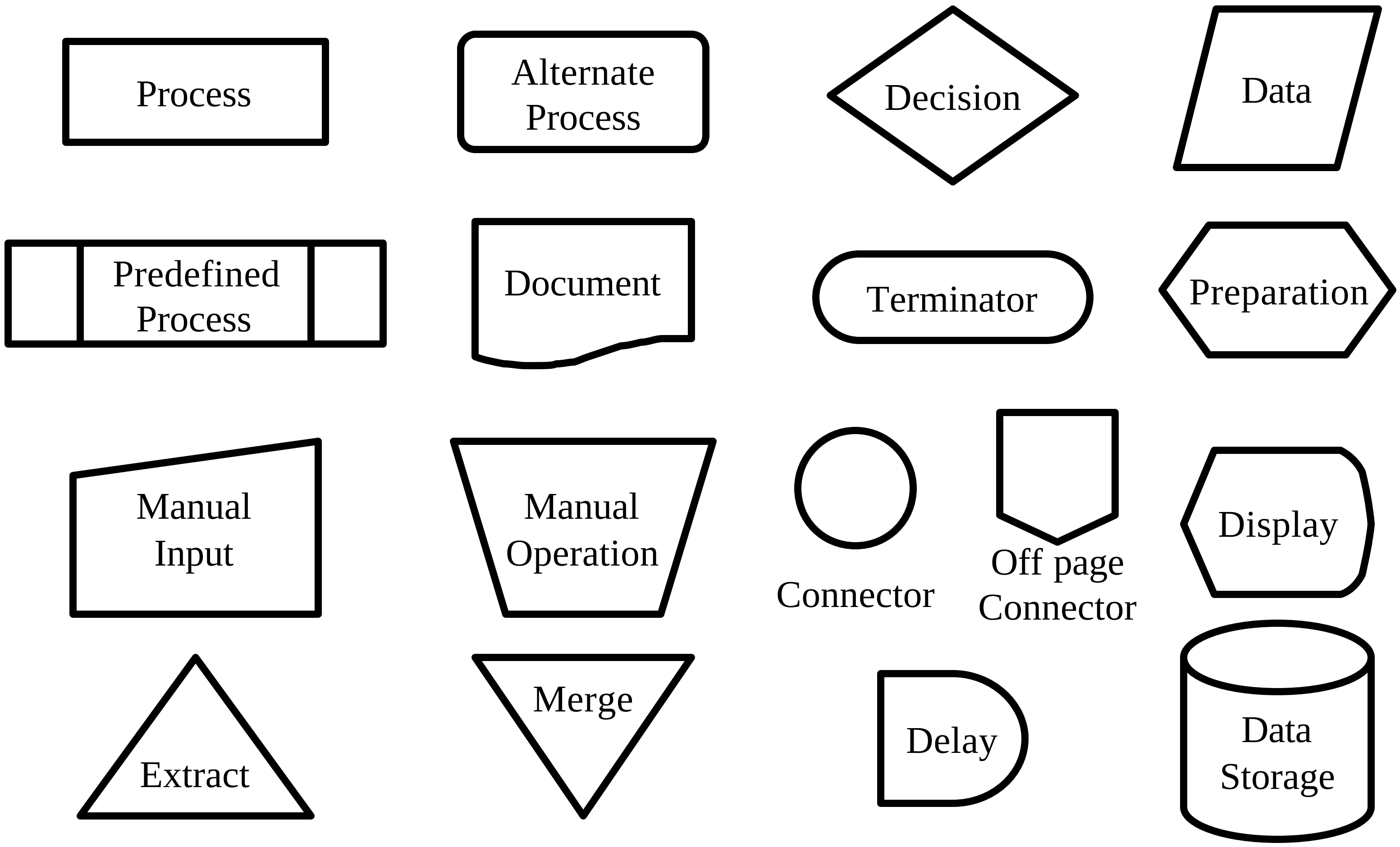
Cause and Effect Diagram



Other Quality Tools

- Flow Charts
- Problem solving
- Root cause analysis
- Plan-Do-Check-Act (PDCA)
- 7 new tools

Example Flow Chart Symbols



Problem Solving Process

- Principles
  - Clearly defined problem
  - Participation w/out criticism
  - Address root causes (not symptoms)
  - Data-driven decisions
- Process
  - State problem from customer view
  - Generate a list of root causes
  - Solutions to address largest causes
  - Prioritize solutions

Root Cause Analysis

- Identify Initial Cause
- Repeated Questioning
  - When was the problem found?
  - Where could the problem have been found?
  - Where should the problem have been found?

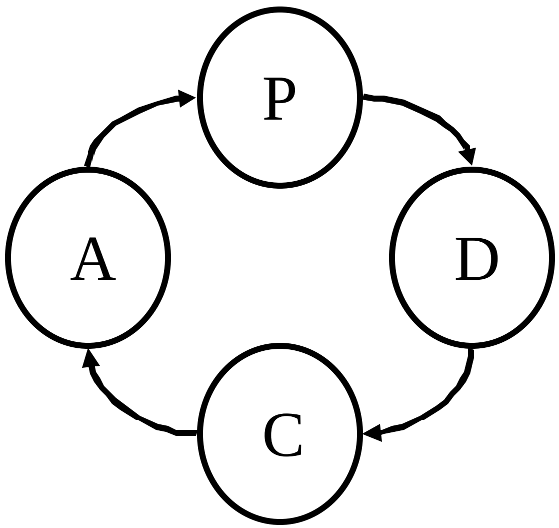
Deming's PDCA cycle (Walter Shewhart Cycle)

Plan:

What to accomplish? What changes? What data? New observations? New test? How to use planned observations

Do:

Carry out the change or test



Check:

Observe the effects of the change or test

Act:

Study the results. What lessons? What predictions? What improvements or changes (if any)?

Seven New Tools

Japanese names	American counterpart
1. Relations Diagram	2. Affinity diagram
2. Affinity diagram	3. Tree diagram
3. Systematic diagram	6. Process decision program chart (PDPC)
4. Matrix diagram	5. Matrix diagram
5. Matrix analysis	1. Interrelationship digraph (I.D.)
6. Process decision program chart (PDPC)	4. Prioritization matrices
7. Arrow diagram	7. Activity network diagram

Affinity Diagrams

Exam Prep Example

Resources

- CSQE Primer
- Literature
- Textbooks
- ASQ BOK

Learn

- Watch video
- CSQE class
- SQE Classes
- Group study
- Tutor
- Self study
- Mentor

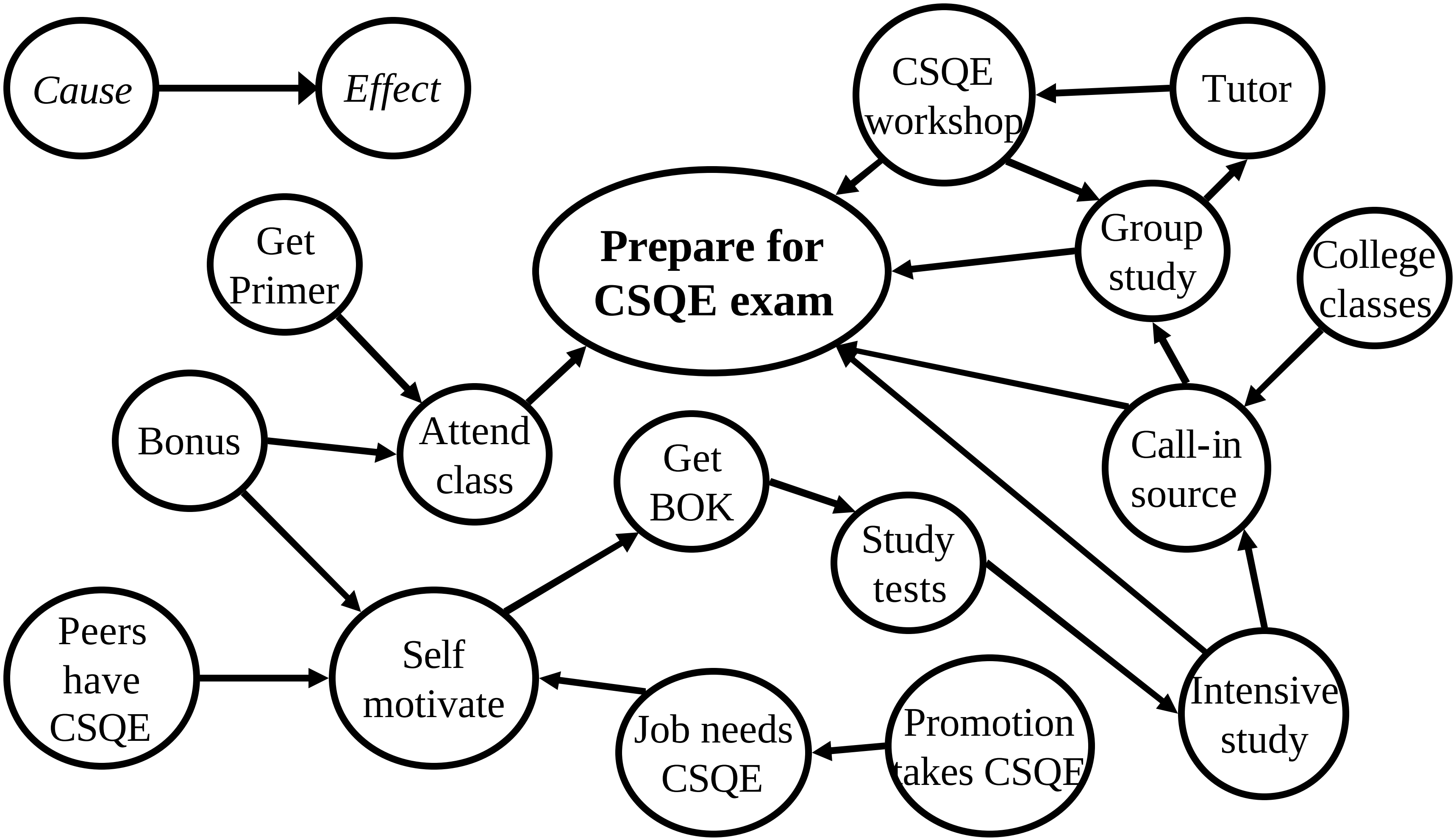
Preparation

- Teach BOK
- Experience
- Study intensively
- Start early
- Study subjects separately
- Practice exams

Motivation

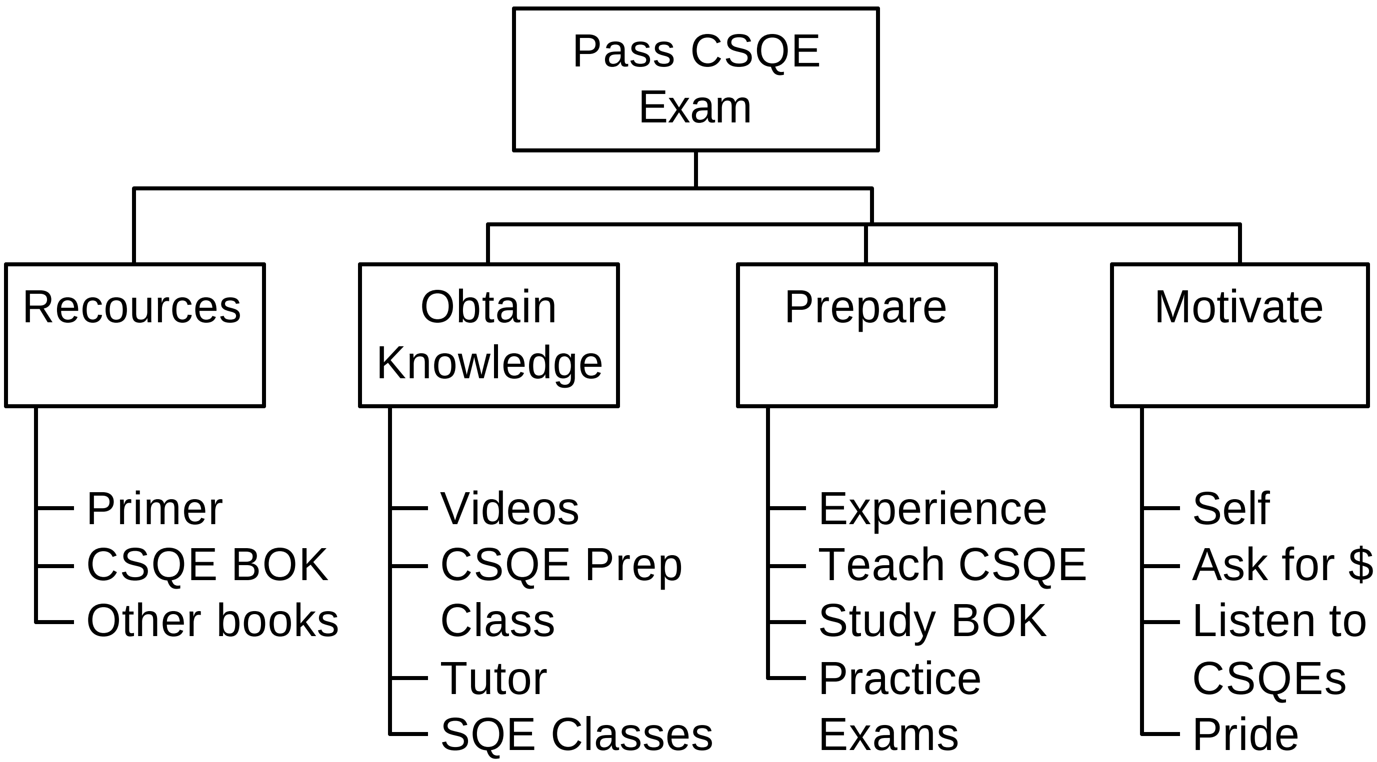
- Bonus $
- Motivate self
- Listen to CSQEs
- Develop pride
- Improve work

Interrelationship Digraph Example



Tree Diagrams

CSQE Test Prep



Criteria Weighting Matrix Example

<u>Weighted Criteria</u>

- List Criteria
- Each person assigns percentages
- Sum to Totals
- Totals are composite weights

Criteria	Perosn 1 Ranking	Perosn 2 Ranking	Perosn 3 Ranking	Perosn 4 Ranking	Total
Work Experience	0.05	0.10	0.10	0.20	0.45
Have a Tutor	0.10	0.20	0.30	0.10	0.70
Study in Group	0.15	0.10	0.05	0.20	0.50
Attend CSQE Refresher	0.25	0.20	0.20	0.30	0.95
Study Old Tests	0.15	0.15	0.25	0.10	0.65
High Motivation	0.30	0.25	0.10	0.10	0.75
Total	1.00	1.00	1.00	1.00	4.00

Criteria/Factor Ranking Matrix

<u>Criteria/Factor Ranking</u>

- List Factors (facets) being weighed
- Consensus rank (order) Factors for each Criteria

Factors \ Criteria	Work Experience	Have a Tutor	Study in Group	Attend CSQE Refresher	Study Old Tests	High Motivation
Knowledge and Ethics	1	2	3	2	1	3
Quality Management	2	3	1	3	3	4
Audits	4	1	2	1	2	2
Engineering Processes	3	4	4	4	5	5
Program Management	7	7	8	7	6	8
Software Metrics	6	8	7	8	8	7
Verification & Validation	5	6	5	6	7	6
Configuration Management	8	5	6	5	4	1

Prioritization Matrix Example

Prioritize

- Multiply Weighted Criteria by Factor Ranking
- Compute Total for Each Factor

Weight	0.45	0.70	0.50	0.95	0.65	0.75	
Criteria / Factors	Work Experience	Have a Tutor	Study in Group	Attend CSQE Refresher	Study Old Tests	High Motivation	Total
Knowledge and Ethics	0.45	1.40	1.50	1.90	0.65	2.25	8.2
Quality Management	0.90	2.10	0.50	2.85	1.95	3.00	11.3
Audits	1.80	0.70	1.00	0.95	1.30	1.50	7.3
Engineering Processes	1.35	2.80	2.00	3.80	3.25	3.75	17.0
Program Management	3.15	4.90	4.00	6.65	3.90	6.00	28.6
Software Metrics	2.70	5.60	3.50	7.60	5.20	5.25	29.9
Verification & Validation	2.25	4.20	2.50	5.70	4.55	4.50	23.7
Configuration Management	3.60	3.50	3.00	4.75	2.60	0.75	18.2

Matrix Diagrams

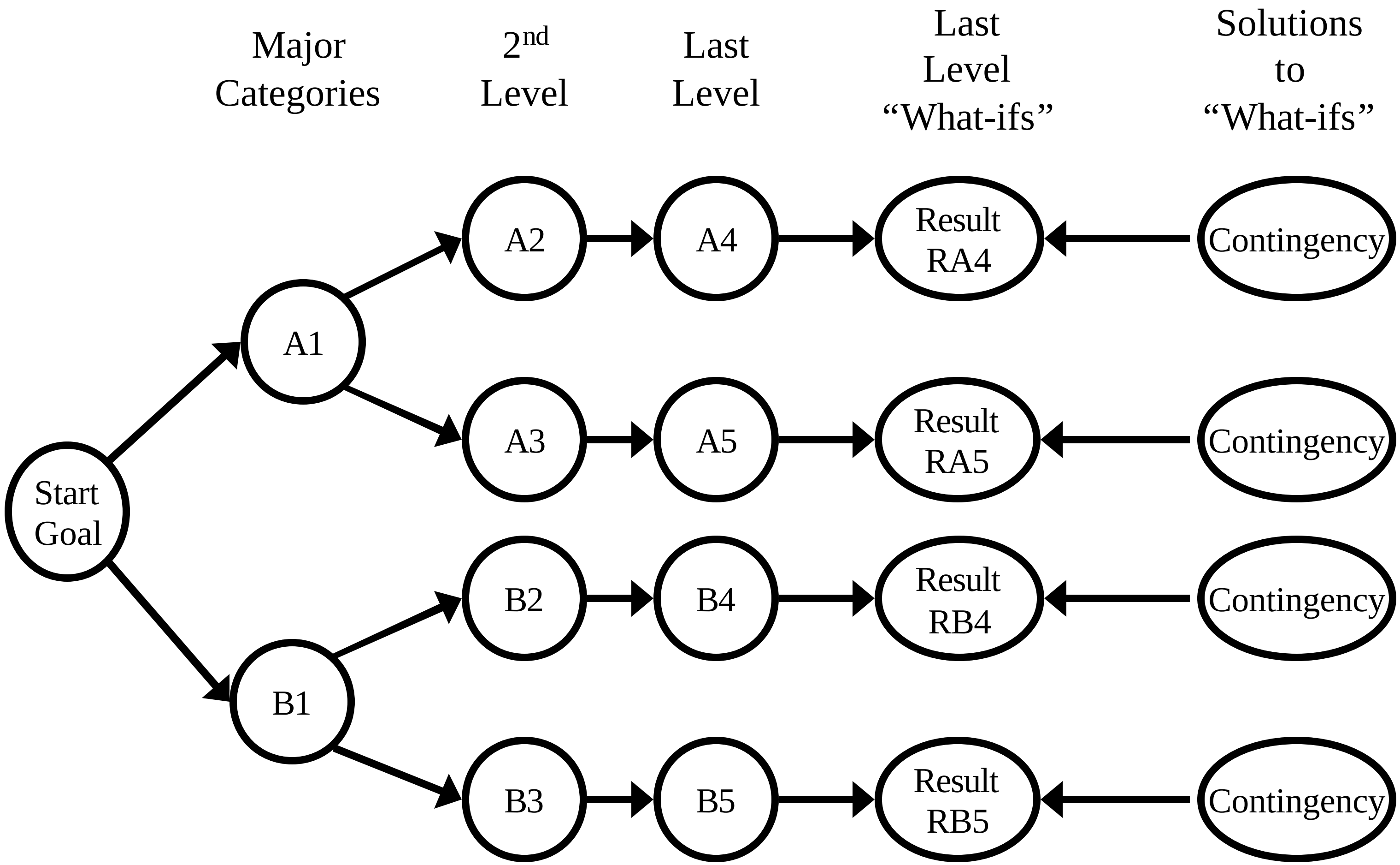
- L-type
  Standard, two factor matrix
- T-type
  Two factors on Y-axis, one on X-axis
- X-type
  Two factors on each Y-axis and X-axis
- Y-type
  Two L-type matrices with common Y-axis
  Standard, three dimensional matrix
- C-type
  Two L-type matrices with common Y-axis
  Three dimensional matrix with unique values

X-type Matrix Example

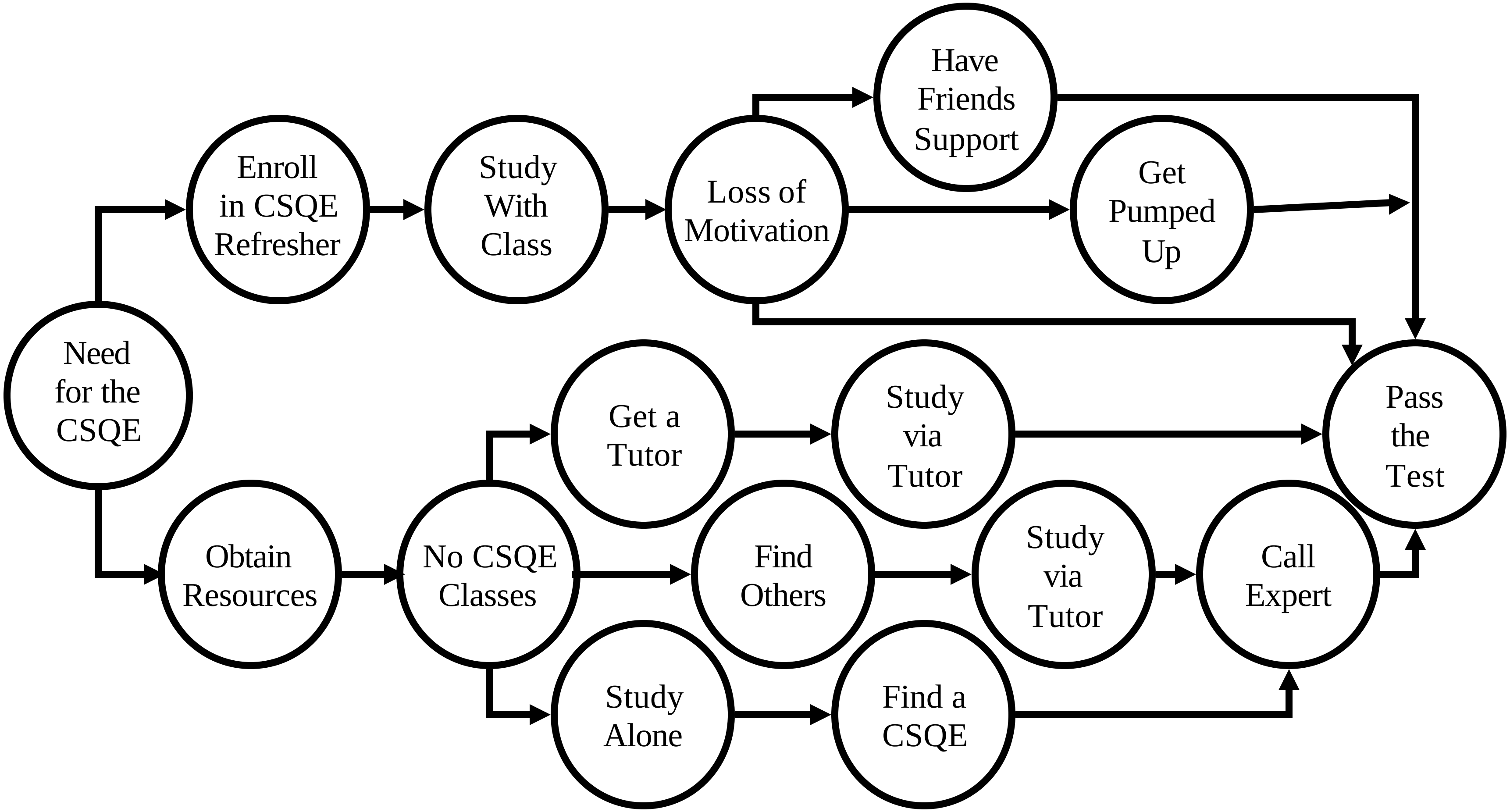
O			Cause 1		**?**			**X**
	X	**?**	Cause 2				**?**	
			Cause 3			**O**		**O**
			Cause 4					
Team 1	Team 2	Team 3		Dept 1	Dept 2	Dept 3	Dept 4	Dept 5
			Problem 1					
	O		Problem 2					**O**
			Problem 3		**?**		**?**	
X	**?**	**O**	Problem 4			**X**		
			Problem 5					
			Problem 6		**X**			

X - Presence documented
O - Presence likely
? - Presence possible

Process Decision Program Charts (PDPC) Example 1



Process Decision Program Chart Example 2



Activity Network Diagram Link

Sampling Theory and Techniques

- Sampling is data collection
- Consumer's risk (β)
- Producer's risk (α)
- Acceptance sampling
  - Attributes plan
  - Variables plan
- Auditing sampling
- Random sampling
- Sequential sampling
- Stratified sampling
- Required sample size

Sampling Errors

		Actual Quality	
		Good	Bad
The Decision Made	Called Good	$1 - \alpha$ producer's confidence	β Type II Error
	Called Bad	α Type I Error	$1 - \beta$ consumer's confidence

Verification and Validation

- General Knowledge, Conduct, and Ethics
- Software Quality Management
- Software Engineering Processes
- Program and Project Management
- Software Metrics, Measurement, and Analytical Methods
- **Software Verification and Validation**
- Software Configuration Management

V&V Theory

- V&V planning procedures and tasks
- V&V program
- Evaluating software products and processes
- Interfaces

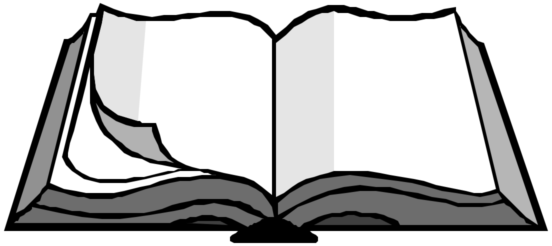
V & V Quality Attributes*

- Accuracy
- Completeness
- Consistency
- Efficiency
- Expandability
- Flexibility
- Interoperability
- Maintainability
- Manageability
- Portability
- Readability
- Reusability
- Reliability
- Safety
- Security
- Survivability
- Testability
- Usability

*** IEEE 1059**

Vocabulary

- Inspection
- Walkthrough
- Review
- Test
- V & V
- Regression



V & V Planning

- Scope for V & V
- Establish objectives
- Analyze project inputs
- Select techniques and tools
- Develop the plan

V & V Methods

- Evaluating products and processes
- Requirements traceability
- Evaluating requirements
- Evaluating interfaces
- Evaluating test plans
- Evaluating the severity of anomalies
- Assessing proposed modifications
- Which V & V tasks should be iterated

Reviewing a V & V Program

- Evaluation criteria
- Requirements traceability
- Evaluate documents
- Evaluate test plans
- Evaluate test results
- Audit results and corrective actions

V & V Interfaces

- Requirements
- Design
- Source code

Inspection Types

- Desk check (walkthrough)
- Walkthrough (evaluate)
- Inspection (debug)
- Management review
- Technical review

Measures for Reviewing Inspection Efforts

- Inspection Rate
- Defect Rate
- Effectiveness Ratio
- Trends
- Technical Accomplishments
- Resource Utilization

Review and Inspection Items

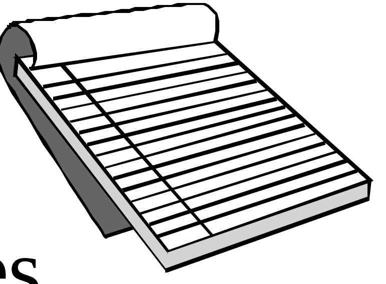
- Project items
- Project change items
- Software tool items
- Software process items

Review and Inspection Process

- Objectives
- Entry criteria
- Exit criteria
- Techniques and methods
- Participant roles

Inspection Data Collection

- Defect type
- Defect class
- Defect severity
- Inspection measures
- Phase containment
- Data management
- Reporting



V&V Test planning and design

- Types of tests
- Test tools
- Test strategies
- Test design
- Test coverage of specifications
- Test environments
- Supplier components and products
- Test plans

Types of Tests

- Functional
- Performance
- Regression
- Load
- Worst case
- Perfective
- Exploratory
- Random-Input
- Certification
- Stress
- Usability
- Real Time
- Life*
- Collision*
- Security*
- Installation*
- Recovery*

* Not included in Primer

Test Levels

- Unit
- Component
- Integration
- System
- Field

Test Strategies

- Top down
- Bottom Up
- Black box
- White box
- Simulation
- I/O first
- Alpha/Beta testing
- Fault insertion
- Fault-Error handling
- Equivalence class partitioning
- Boundary value analysis
- Cause effect graphing
- Error guessing
- Customer defects

Test Design

- Excellent test cases
- Equivalence classes
- Scenarios
- Customer defects
- Test coverage
- Fault insertion

An Excellent Test Case

- Reasonable probability of catching an error*
- Not redundant with other tests*
- Best of breed*
- Exercise to stress the area of interest
- Minimal use of other areas
- Neither too simple nor too complex*
- Makes failures obvious*
- Allows isolation and identification of errors

* Kaner (1993)

Test Coverage of Specifications

- Functions
- States
- Data
- Time
- Localization
- Internationalization

Test Environments

- Test Libraries
- Drivers and Stubs
- Harnesses
- Compatibility Labs
- Simulators
- Other tools

Supplier Quality

- Testing supplier components
  - COTS
  - MOTS
  - Fully Developed
- Customer deliverables specifications
  - User documentation
  - Marketing materials
  - Training materials

Test Planning

- Types of plans
  - System
  - Acceptance
  - Validation
- Implementation
  - Scheduling
  - Freezing
  - Dependencies
  - V-model
  - Error repair models
  - Acceptance testing
  - Resources
  - Analysis of test results

Test Documentation

- Defect recording
- Defect tracking
- Test Plan
- Test Log
- Test Design
- Test Case
- Test Procedure

Test Reports

- Defect report
- Test result
- Test report
- Traceability matrix
- Test completion metrics
- Trouble reports

Test Management

- Scheduling
- Freezing
- Resource Management
- Dependencies
- Analysis of Results

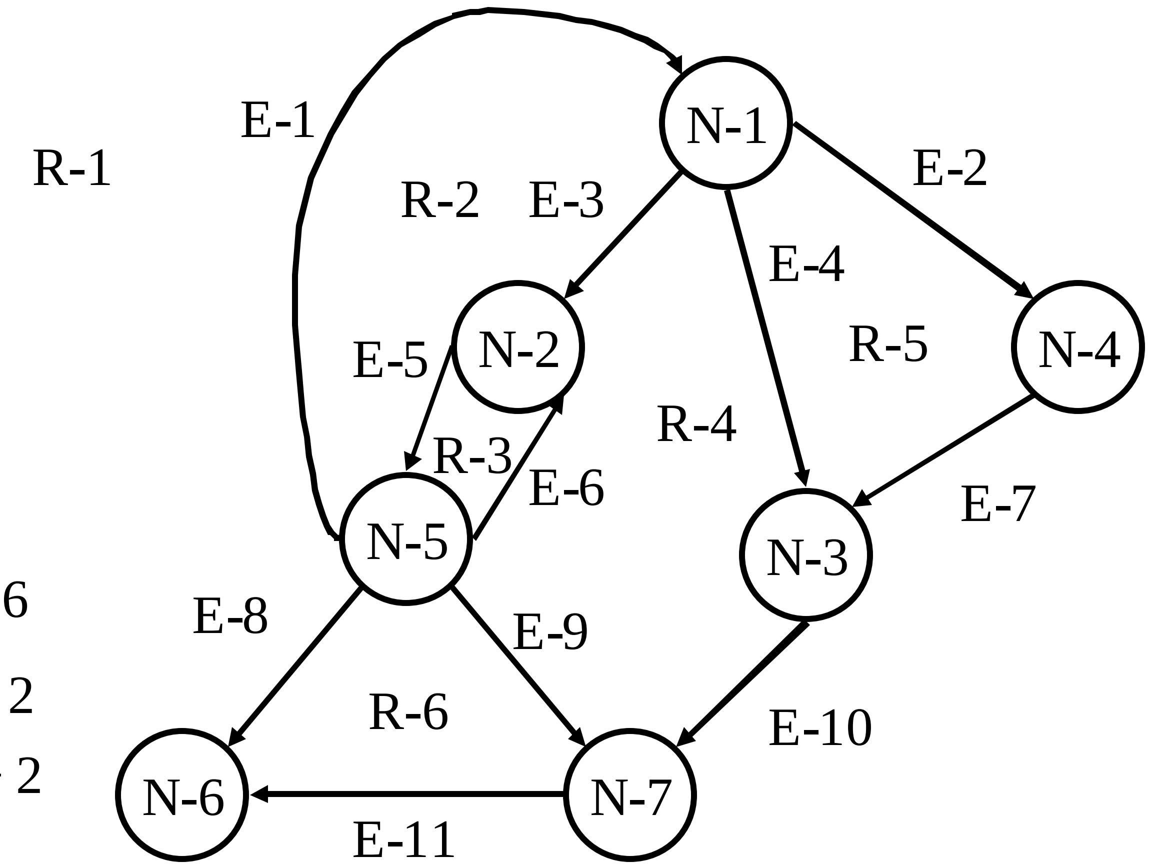
Reviewing Testing Efforts

- Tasks
  - Technical results
  - Planning
  - Risk management
  - Resource use
- Methods
  - Management review
  - Technical review
  - Inspection
  - Walkthrough
  - Test
  - Audit

Test Coverage of Code

- Branch
- Condition
- Domain
- Cyclomatic complexity
- Boundary
- Path
- Individual predicates
- Data
- Functions

Cyclomatic Complexity



R(G) = 6

N(G) = 7

E(G) = 11

V(G) = R(G) = 6

V(G) = E - N + 2

= 11 - 7 + 2

= **6**

V(G) = 6

Severity of Anomalies*

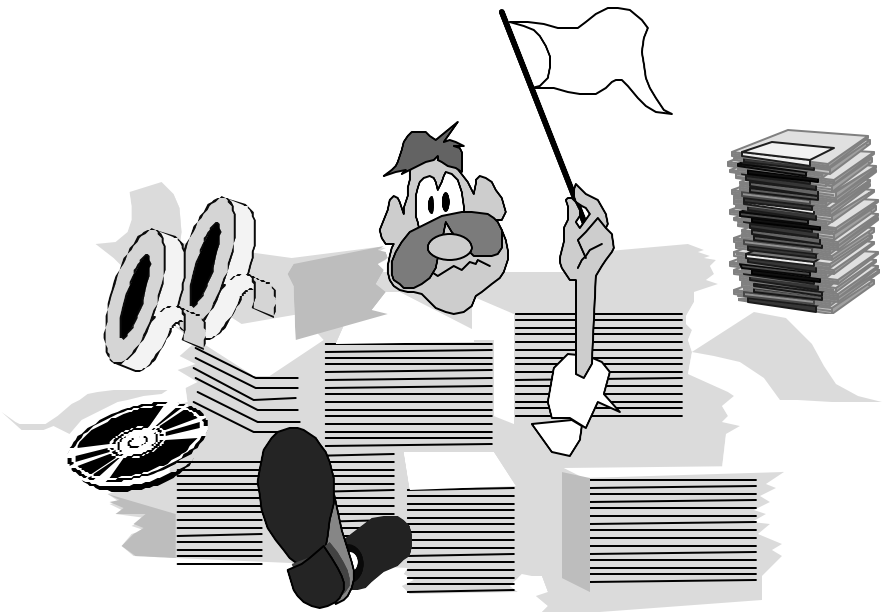
Criticality	Description	Level
High	Selected function affects critical performance of the system.	4
Major	Selected function affects important system performance.	3
Moderate	Selected function affects critical system performance, but workaround strategies can be implemented to compensate for loss of performance.	2
Low	Selected function has a noticeable effect on system performance, but only creates inconvenience to the user if the function does not perform in accordance with requirements.	1

*** Example from IEEE 1012-1998**

Configuration Management

- General Knowledge, Conduct, and Ethics
- Software Quality Management
- Software Engineering Processes
- Program and Project Management
- Software Metrics, Measurement, and Analytical Methods
- Software Verification and Validation (V&V)
- **Software Configuration Management**

Configuration Management



Benefits from CM

- Product attributes defined
- Documented product configuration
- Labeled and correlated
- Change evaluation
- Change management
- Configuration captured
- Configuration verification

Rationale for CM

- Change inevitable
- Complexity increases
- Simultaneous updates
- Double maintenance
- Shared code
- Common code
- Versions evolve

CM Terms$_1$

- Baseline: A uniquely identified consistent group of related SCIs captured as a milestone. A specification or product that has been reviewed and agreed upon as a basis for future development, which can be changed only through procedures
- CASE: Computer Aided Software Engineering; Use of software tools to design, requirements tracing, code production, testing, doc generation, and other software activities
- Configuration: Functional and physical characteristics of hardware or software as set forth in technical docs or achieved in a product
- Configuration Control: Element of CM; Evaluation, coordination, approval (or dis-), and implementation of changes to CItems after establishment of C ID.
- Configuration Control Board (CCB): Group responsible for evaluating and approving (or dis-) proposed changes to C Items and ensuring implementation of approved changes

CM Terms$_2$

- Configuration Identification: selecting the C Items for a system and recording their functional and physical characteristics in technical docs
- Configuration Item (CI): Hardware or Software (or both) designated for CM and treated as a single entity in the CM process
- Configuration Management (CM): Discipline applying technical and admin direction and surveillance to ID and doc the functional and physical characteristics of a C Item, control changes, record and report change processing and implementation status, and verify compliance with requirements
- Configuration Status Accounting: Element of CM: recording and reporting info to effectively manage a config; includes config ID, status of proposed changes, and implementation status of approved changes
- Patch: 1) Modification to object program by replacing code; 2) Modification to object program without recompiling from source
- Release: Formal notification and distribution of approved version

CM Terms$_3$

- Software Configuration: programs, documents, data structures; all items produced as part of software engineering process
- Software Configuration Item (SCI): Information produced, such as source code, design documents, test suite, or program modules. The software subset of CI. “That which is managed together.” (SCI often includes tools so versions can be reproduced)
- Software Development Life Cycle (SDLC): Period of time begins at concept and ends when software is no longer available for use
- Software Engineering environment: Hardware, software, and firmware used for SE effort; includes equipment, compilers, assemblers, OS, debuggers, simulators, emulators, test tools, documentation tools, and DBMS
- Software Library: Controlled collection of software and related documents to aid Software development, use, or maintenance. Types: master, production, library, software development library, software repository, and system library
- Versions: named SCI set with a defined set of functional capabilities

Configuration Infrastructure

- Configuration management
- Library/repository processes
- Defect tracking and library tools

SCM Responsibilities

- Configuration identification
- Configuration control
- Status accounting
- Co-chair audits and reviews
- Co-chair CCB
- Maintain baselines
- Implement SCM plan

SCM Tools

- SEI's 15 concepts
- SEI's four SCM models
  - Check out/in
  - Composition
  - Long transactions
  - Change sets

UNIX SCM Utilities

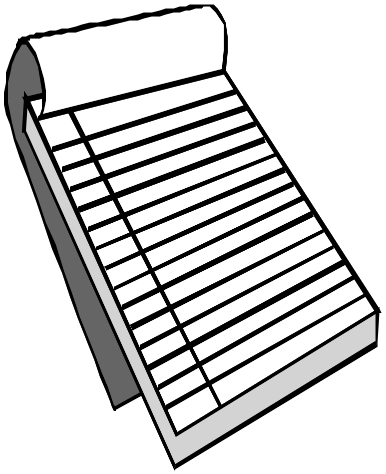
- SCCS and RCS
- CVS
- Make

SCM tool features

- Version Control
- Configurations
- Processes
- Change control
- Team support
- Library/Repository
- Security/Protection
- Reporting/Query
- Tool integration
- Builds
- Releases
- Customization
- GUI

Configuration Identification

- Configuration items
- Structure
- Identification of items
- Baselines
- Accessibility
- Traceability
- Software builds
- Methods



Software Product Partitioning

- Managerial factors
- Technical factors
  - Hierarchy
  - External objects
  - Internal objects
  - Environment and Tools

Configuration Control

- Version control
- Naming
- Configuration ID
- Library control
- Release process
- Patching

Configuration Control

- Item and baseline control
- Proposed modifications
- Review and configuration control boards (CCBs)
- Concurrent development
- Traceability
- Version control
- Configuration item interfaces

Configuration Status Accounting

- Status reporting
- Changes to configuration items and baselines
- Documentation control
- Configuration Audits
  - Functional configuration audit
  - Physical configuration audit

Change Control

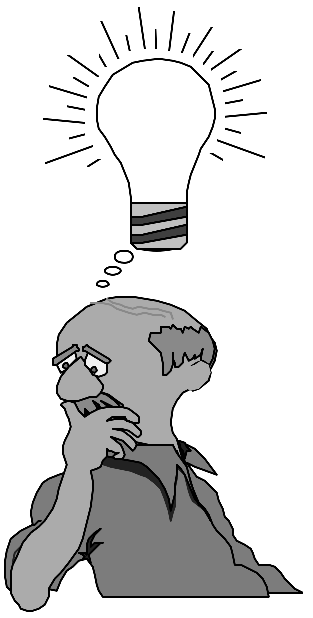
- Change process
- Change review
- Change approval
- Baselines
- Impact analysis
- Change management
- Verification

CM Issues

- Definition of configuration
- Status of configuration
- Change control
- Communication of changes
- Release and distribution issues
  - Product release process issues
  - Packaging, production, and distribution

Exam Preparation

- Get good rest
- Be prepared to focus
- Bring indexed library
- Clear your head
- Relax and stay cool
- Soft pencils, calculator, munchies



About The Exam

- Know the rules
  - Score is number right
  - Time is main enemy
- Take care of the answer sheet
  - Verify name and exam information
  - Check the numbering layout
  - How to mark/clear answers
- Write notes in the test book
  - Eliminate wrong answers
  - Mark probably right
  - Where will you find the answer?

Taking The Exam

- Answer easy questions first
- Make two + passes
- Re-read every question
- Parse hard questions
- "Batch" research
- Answer all questions at the end
  - Leave it until the last second
  - Guess smart - use your notes