- Orthogonal Array Testing is a method that may be utilised to cut down on the amount of possible permutations and achieve maximum coverage while conducting fewer tests overall.

Example

• Take into consideration the send option available in the fax programme.

• The send function receives four parameters, which are referred to as P1, P2, P3, and P4. Each one might be any one of three distinct values.

• P1 assumes the following values:

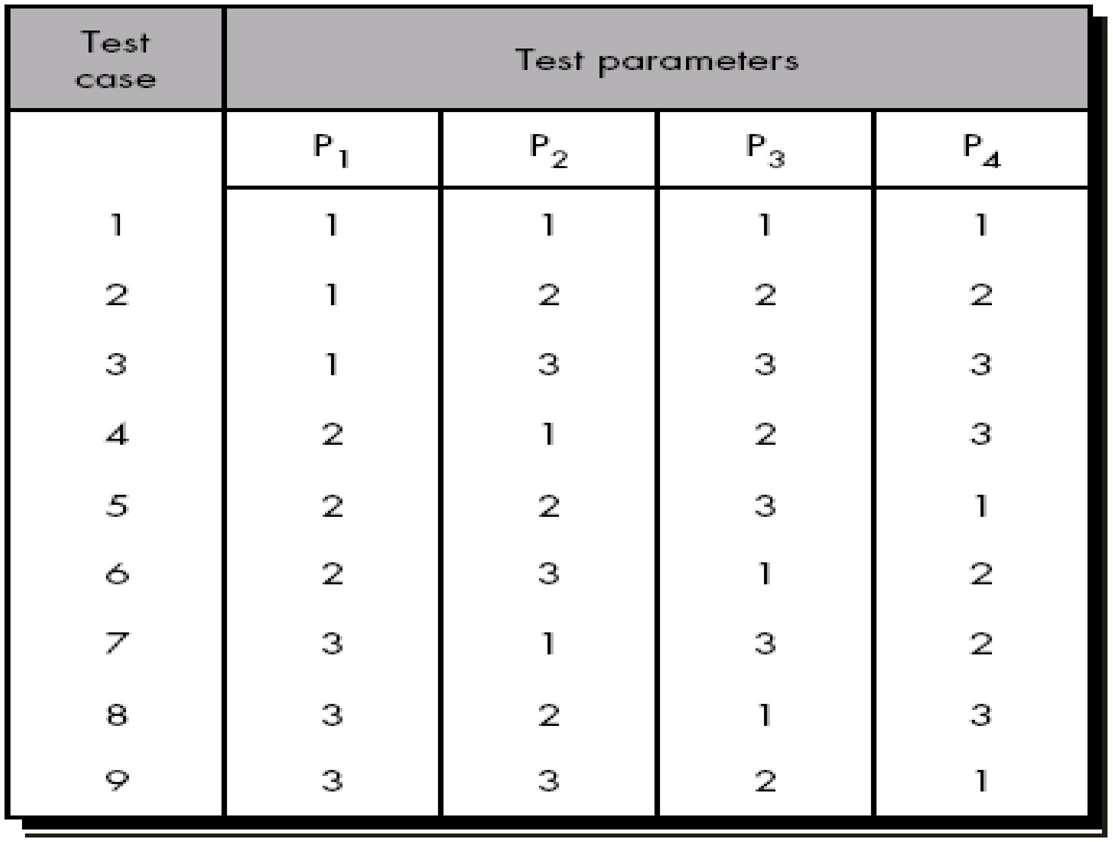
Send it now if P1 equals 1, send it an hour from now if P1 equals 2, send it after midnight if P1 equals 3.

• Other send functions would be indicated by P2, P3, and P4 taking on the values 1, 2, and 3 respectively.

• The OAT is an array of values, and each column in the array represents a parameter, which is a value that can be one of a set of predetermined options called levels.

• Each entry in this table represents a different test case.

• Parameters are mixed in pairs rather than expressing all of the potential combinations of levels and parameters; this is done in order to improve performance.



Test case	Test parameters			
	P_1	P_2	P_3	P_4
1	1	1	1	1
2	1	2	2	2
3	1	3	3	3
4	2	1	2	3
5	2	2	3	1
6	2	3	1	2
7	3	1	3	2
8	3	2	1	3
9	3	3	2	1

MBU MOHAN BABU UNIVERSITY
Sree Sainath Nagar, Tirupati 517 102

Module 5 RISK, QUALITY MANAGEMENT AND REENGINEERING

Risk and Quality Management: Reactive and Proactive risk strategies, Software risks, Risk Mitigation Monitoring and Management (RMMM), RMMM plan, Software quality factors, Defect Amplification Model, Formal Technical Reviews (FTR), Software Quality Assurance (SQA)-Tasks, Goals and Metrics; Software reliability.

Reengineering: Introduction, Business Process Reengineering (BPR), Software Reengineering, Restructuring, Reverse engineering, Forward engineering.

RISK MANAGEMENT

Introduction:

* A software development team can learn to deal with uncertainty through a process known as risk analysis and management.

* A risk is a prospective problem; it may or may not materialize.

* However, it is a good idea to identify risks, evaluate their likelihood of occurring, and calculate their potential impact.

Reactive VS Proactive Risk strategies:

(i) Reactive risk Strategies:

- Until something goes wrong, the software team does nothing about potential dangers.
- When an issue arises, the team springs into action to try to find a solution as soon as possible.
- In this state, one is in "Fire - Fighting Mode," often known as the "Indiana Jones School of risk management" because of the famous quote attributed to the hero, "Don't worry I'll think of something!" when presented with a seemingly insurmountable problem.

(ii) Proactive Risk Strategies:

* This method gets underway a very long time before any technical work is done. Here, we take a look at the various dangers that could arise.

After determining their likelihood and potential for harm, as well as the order in which they should be addressed, the software team devises a strategy for mitigating the risks they face.

* The primary goal is to prevent risk, but because no risk can be eliminated entirely,

* The team is working to establish a contingency plan that will enable it to react in a controlled and efficient manner in the event that a risk is encountered.

<u>Software Risks: </u>Risk Characteristics:

(1) The possibility that the risk will not materialize; in other words, there are no dangers that are one hundred percent likely to occur.

(2) Unwanted Consequences Or Losses: If a risk were to materialize, unintended consequences or losses would follow.

* When doing an analysis of risks, it is essential to quantify both the degree of loss that is connected with each risk as well as the level of uncertainty that is associated with that risk.

<u>Categories of Danger:</u>

(i) Dangers to the Project:

* It has an impact on the project's overall plan

* If a project's risks are realised, it is likely that the following will occur:

=> The timetable for the project will be delayed.

=> There will be an increase in cost.

* The identification of project dangers

=> Possibilities Regarding the Budget

=> Set a timetable

=> Staffing and organization in terms of personnel

=> Useful Reference The term "Stakeholder"

==> Problems With Requirements..

(ii) Risks Involving Technology:

* It poses a risk to both the overall quality and the schedule adherence of the software that is to be produced
* If a technological danger turns out to be a reality, implementation can become more challenging.
* Identification of Potential Technical Risks
=> The Possibile Construction
=> The Putting Into Action
=> Point of Contact
=> The Act of Checking
=> Issues with upkeep and maintenance
* There is a potential for technical concerns because the problem is more difficult to address than we had anticipated it would be.

(iii) Dangers to the Business:
* It has an impact on the possibility of the programme being constructed.
* The following are the top five dangers to a business:
(i) Constructing a wonderful product (Or System) that virtually no one desires at all [Risks in the Market]
(ii) Developing a product that is no longer compatible with the overarching strategy of the company [Risks to the company's strategic position]
(iii) Constructing a product that the sales team is unable to sell because they do not comprehend how to do so [Risks Regarding Sales]
(iv) The loss of support from top management as a result of a shift in the organization's focus (Or) a transformation in individuals [Risks Involved in Management]
(v) Failing to invest sufficient financial resources or personnel [Risks to the Budget]

(iv) The Known Dangers:

* These risks can be identified when the project strategy, as well as the commercial and technological environment in which the project is being produced, have undergone thorough evolution.
* Additional credible information sources include the following:
=>Date of delivery that is impossible to meet

=> The lack of requirements for documents
=> An unfavourable setting for development

(v) Risks That Can Be Anticipated:

* These dangers were derived from analysis of completed projects in the past.

As an example
:=> Turnover in the workforce
=> Lack of effective communication with the client
=> A reduction in the amount of work put in by staff as continuous maintenance requests are attended to

(vi) Dangers That Cannot Be Anticipated:
* These dangers are like a wild card in a deck of cards because they can and do materialise. However, it is quite challenging to predict them in advance.

Identifying Potential Dangers:
* It is an organised effort to identify potential dangers to the project plan, such as timetables, resource loads, and cost estimations.
* A known and predictable risk is identified, which is the first step that a project manager takes towards avoiding it when possible and regulating it when it is important to do so.

There are two main categories of dangers, which are as follows:

(i) Generic Risks - Every software development project faces the possibility of these risks.
(ii) Product-special Risks: These can be identified only by individuals who have a clear grasp of the technology, the people, and the environment that is special to the software that is going to be produced.
These risks are unique to the software that is going to be built.

* The creation of a risk item check list is one approach of identifying risks that can be used.

* The checklist can be used to identify potential risks, with a particular emphasis on a subset of known and foreseeable dangers falling into one or more of the following generic subcategories:
Size of the Product = Danger Associated with the Overall Size of the Software to be constructed (Or) remodeled.
The risk that is connected with the limitations placed on the business as a result of either administration or the market place
Customer Characteristics = Risk Associated with the Degree of Complexity of the the ability of both the customer and the developers to interact with the client in a manner that is with in a timely manner.

Risks linked with the extent to which the process is followed = Process Definition
The software development method has been outlined and is being carried out by the Organisation for Development and Cooperation Risks related to the quantitative abilities linked with the development environment as well as the quality of the tools that will be utilised to construct the item being soldRisk linked with the complexity of the system equals the technology that must be built.
As well as the "Newness" of the technology to be constructed, It is contained within a package by the System.
Risks associated with the overall technical and project experience of the software engineers who will be doing the work are proportional to the size and level of expertise of the staff.
Assessing the Overall Dangers of the Project:
* The questions that are presented here are based on risk information received from seasoned software project managers.
* The order of the questions reflects their relative significance to the completion of the project.
Possible Answers:
(1) Do the highest-level software and customer managers have an official commitment to back the project?

(2) Do the people who will be using the finished result have a passionate commitment to the project and the system or product that will be built?

(3) Is there a mutual understanding between the software engineering team and its customers regarding the requirements?

(4) To what extent have customers been involved in the process of defining the requirements?

(5) Do end-users have expectations that are grounded in reality?

(6) Is there no change to the project's scope?

(7) Does the team working on the software engineering have the appropriate variety of skill sets?

(8) Can you guarantee that the project requirements won't change?

(9) Does the team working on the project have previous experience working with the technology that will be implemented?

(10) Does the project team have a sufficient number of members to complete the task at hand?

(11) Do all of the customer and user constituencies have the same opinion regarding the significance of the project as well as the needs for the system or product that will be constructed?

* The proportion of unfavorable replies to these questions is directly linked to the degree to which the project is at risk.

Risk Factors and Influencing Factors:

* The following criteria are used to define the risk factors:

(1) Performance Risks - The degree to which it is unknown that the product will fulfill its specifications and be suitable for the purpose for which it was designed.

(2) Financial Risks = The degree to which it is uncertain that the project's allotted funds will be sufficient

(3) Support Risks = The degree to which it is uncertain that the finished programme will be simple to amend, modify, and improve.

(4) Schedule Risks => The degree to which it is unknown if the project schedule will be kept and that the product will be delivered on time

* The influence of each risk driver on the various risk components can be broken down into the following four categories:

=> Insignificance

=> On the margins

=> Very important

=> A cataclysmic event

Projections of Danger:

* Another name for this concept is risk assessment.

* It tries to assign a score to each risk in two different ways:

(i) the chance or probability that the risk is indeed there in the situation

(ii) the repercussions of the problem connected with the risk in the event that it occurs take place.

* The project planner and the technical personnel will complete the following four processes of risk projection:

(1) Construct a rating system that takes into account how likely it is that a risk may occur.

(2) Explain in detail the possible outcomes of the risk.

(3) Determine an estimate of how the risk will affect the project and the product.

(4) Make sure that the overall precision of the risk projection is taken into consideration, so that there is no room for misinterpretation.

7.12 <u>Risk Table:</u>

*A risk table provides a project manager with a simple technique for risk projection

Risks	**Category**	**Probability**	**Impact**	**RMMM**
Size estimate may be significantly low	PS	60%	2	
Large number of users than planned	PS	30%	3	
Less reuse than planned	PS	70%	2	

Delivery deadline will be tightened	BU	50%	2	
Customer will change requirements	PS	80%	2	
Staff Inexperienced	ST	30%	2	

Values of Influence

1. Extremely bad
2. Critical Level
3. Minimum:
4. Maximum:

* All risks identified by the project team are included in the first column of the table.

PS = Project Sizing Danger

BU == Business Risk

* Third column == Impact

* Fourth column == Probability of Occurrence

* Fifth column == After the first four columns are filled in, the risk table is complete; it is sorted according to likelihood and impact.

* High probability and high impact risks move to the top of the table

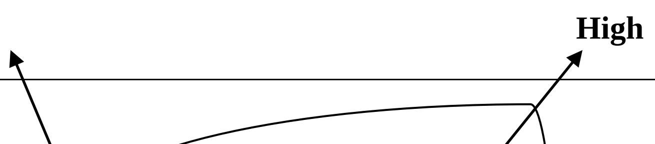
* Low probability risks move to the bottom of the table

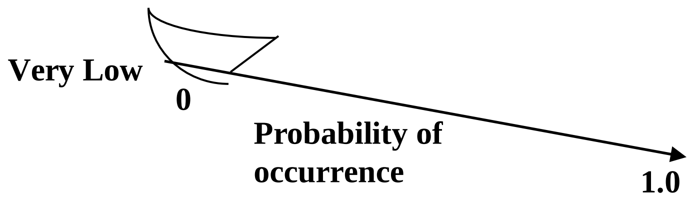
* A "Cut Off Line" is established when the project manager analyses the sorted table.

*All risks above the Cut Off Line must be managed, as indicated by the horizontal line drawn at some point on the table, which suggests that only those risks above the line will receive further attention.

* Risk mitigation, monitoring, and management information can be found at the link given in the fifth column labelled RMMM.

<u>Example:</u>





* A risk factor like the one in the above image that has a big impact but a low chance of happening shouldn't receive a lot of management attention.
* Both low impact risks with a high probability and high impact risks with a moderate to high likelihood should be carried over into the subsequent risk analysis steps.

Risk Refinement:

* As more time goes by and more information is gathered about the project and the risk, it is feasible that the risk can be refined into a group of hazards that are more specific.
* One method for performing refinement is to express the risk in the format of Condition, Transition, and Consequence [CTC]. * The potential harm is described using the following format:

Given that <condition> then there is concern
That (possibly) <Consequence>

Risk Mitigation, Monitoring and management [RMMM]:

* If a software team takes a preventative, proactive approach to risk management, they will always come out on top.
* The following are the preventative measures that could be taken:

Mitigation Strategy:

=> In order to discover the factors that contribute to employee turnover, you need have a conversation with the current personnel.

Example: Unsafe working conditions, low pay, and a highly competitive employment market

=> Avoid those factors that are within our control prior to the beginning of the project

=> Assume that people will quit the project at some point after it has begun, and work on developing strategies to maintain continuity in their absence.

Organise the project team in such a way that knowledge regarding each aspect of the development process is broadly distributed. Define the documentation standards, and then set up a procedure to make sure that the papers are developed in a timely manner.

=> Carry out peer reviews on every piece of work [so that more than one person is aware of what's going on]

=> Ensure that every important technologist has a backup staff member assigned to them. When initiatives are already in progress, risk monitoring operations start

=> The following are some of the factors that can be monitored:

Monitoring Strategy:

=> The general attitudes of the members of the team in response to the demands of the project

=> The degree to which the team's members have developed strong interpersonal relationships with one another

=> The cohesiveness of the team as a whole

=> Problems that could arise with regard to salary and benefits

=> Differences in the quality of work available both inside and outside the organisation

Risk Management and Contingency Planning:

* The project is well on its way to completion, and a number of people have announced that they would be quitting.

* Supposing that the risk reduction approach was implemented correctly, this would mean that

=> Backup is availableInformation is recorded for future reference.

=> The crew as a whole has been made aware of the information.

In addition, the manager could examine the project schedule by itself for a little period of time.

* It should be noted that additional project costs will result from RMMM stages.

RMMM – Plan:

* The RMMM plan, which records every task completed for risk analysis

* The project manager incorporates it into the overall project plan.

* Some software teams use Risk Information Sheets (RIS) to document each risk separately rather than creating a formal RMMM document.

RISK INFORMATION SHEET			
Risk Id: CSE -06	**Date:** 11/12/2006	**Probability:** 80%	**Impact:** High
Description: In actuality, only 70% of the software components slated for reuse will be incorporated into the program.			
Refinement / Context: **Sub Condition 1:** Without awareness of internal design guidelines, a third party built certain reusable components. **Sub Condition 2:** There is a lack of solidification in the design elements for component interfaces. **Sub Condition 3:** A few reusable parts have been implemented using a language that the target environment does not support.			
Mitigation / Monitoring: (1) To ascertain whether design issues are being followed, get in touch with a third party. (2) Initiate the interface standard completion process.			
Management / Contingency Plan / Trigger: 30, 300 is the new computation. Include his payment in the project's contingency budget. Create a revised timetable and assign staff appropriately. **Trigger:** Effectiveness of mitigation measures as of 10/3/06			
Current Status: 15/3/2007 : Mitigation steps Initiated			
Originator: Arivu		**Assigned:** Selvan	

1. Quality

- Quality as “a characteristic or attribute of something.”
- There are two possible types of quality: -
    - *A product's design quality improves if it is manufactured in accordance with requirements;*
    - *The degree to which the design standards are adhered to during production is known as quality of conformance.*
- When it comes to software development,
    - requirements, specifications, and system design are all included in quality of design.
    - The main focus of quality of conformity is implementation.

User satisfaction = compliant product + good quality + delivery within budget and schedule

- *Quality control refers to the various tests, evaluations, and inspections that are conducted during the software development process.*
- *The procedure has a feedback loop as part of quality control.*
- *The idea that every work product has measurable, established requirements to which we can compare the results of every operation is fundamental to quality control.*
- *The feedback loop is necessary to reduce the amount of flaws that are created.*

3. Quality Assurance

- *Management's auditing and reporting duties comprise quality assurance.*
- *Should the data obtained from quality assurance reveal flaws, management must address the issues and deploy the required resources to rectify quality concerns.*

4. Cost of Quality

- *All expenses incurred in pursuing quality or carrying out quality-related tasks are included in the cost of quality.*
- *Quality costs* may be divided 3 mode of cost:
    - Prevention
    - Appraisal
    - Failure.

Software Quality Assurance (SQA)

Conformance to openly stated functional and performance objectives, explicitly defined development standards, and implicit features that are anticipated of all software that has been produced professionally is the definition of software quality.
•The purpose of definition is to emphasise the following three crucial points:

The requirements for the software serve as the basis for determining its overall quality. A lack of quality can be defined as non-compliance with the requirements.

- A set of development criteria is defined by the standards that have been specified. If the criteria are not followed, it will almost certainly result in a lack of quality.

A group of unstated requirements is frequently not taken into consideration. The quality of software is questionable if it satisfies its formal requirements but falls short of meeting its implicit requirements even if it does so.

SQA group activities

The Software Quality Assurance (SQA) group is comprised of software engineers, project managers, customers, salespeople, and individual members.
• There are two distinct groups involved in the process of ensuring the quality of software.
- Computer programmers who are involved in technical tasks
- Planning, directing, documenting, analyzing, and reporting on quality assurance activities are the responsibilities of the SQA group.

• Software engineers address quality activities through the use of technical protocols and metrics, formal technical reviews, and carefully planned software testing.
• The mission of the SQA group is to provide support to the software development team so that they may produce a product of high quality.

Role of an SQA group

1. Prepares an SQA plan for a project.

- The plan is developed during project planning and is reviewed by all stakeholders.

• The plan identifies the following:

- Audits and reviews that are to be undertaken
- Evaluations that are to be performed
- Specifications for the project that relate to the appropriate standards
- Protocols for the recording and investigation of errors
- Documents that are going to be created by the SQA group
- The total number of comments that were submitted to the software development team

2. Participates in the development of a project's software process description. • The SQA group reviews the process description to ensure that it conforms to externally enforced standards (like ISO-9001), organizational policy, internal software standards, and other project plan components.

3. Reviews software engineering-related activities to make sure the defined software process is being followed. The SQA group is in charge of finding, recording, and monitoring process deviations as well as verifying that any required modifications have been made.

4. Performs audits on certain software work products to check for conformity with standards that have been established as part of the software development process.

• The SQA group examines certain work items, locates, documents, and keeps track of deviations; ensures that repairs have been made; and provides the project manager with periodic updates on the results of its work.

5. Ensures that any variations in the software work and work products are properly documented and dealt with in accordance with a procedure that has been documented.

• There is a possibility that the project plan, the process description, the applicable standards, or the technical work products will contain deviations.

6. **Records any noncompliance and reports to senior management.**

- Items that are not in compliance are tracked until they are fixed.

FORMAL TECHNICAL REVIEWS

- It is a software quality assurance activity performed by software engineers

Objectives of the FTR are

- To find mistakes in the logic, function, or implementation of any software representation;
- To confirm that the software under review satisfies its requirements;

- To make sure the software has been represented in accordance with established standards;
- To produce software that is developed uniformly;
- To simplify projects.

- Walkthroughs, inspections, round-robin reviews, and other small-group technical evaluations of software are really included in the FTR class of reviews.

FTR- Review meeting

• The normal number of attendees for a review meeting should be between three and five people; • The number of participants in the review should fall anywhere between three and five.

• There should be some level of advance preparation, but the amount of effort required from each individual should not exceed two hours.

• The time allotted for the review meeting must be significantly shorter than two hours.

• FTR concentrates on a particular (and relatively insignificant) portion of the software as a whole. • For instance, rather than attempting to examine the complete design, FTRs are carried out for each component or small group of components.

• Each and every review meeting ought to be constrained by the following guidelines:

• Typically, the evaluation should include participation from between three and five different people.

• There should be some level of advance preparation, but the amount of effort required from each individual should not exceed two hours.

• The time allotted for the review meeting must be significantly shorter than two hours.

• FTR concentrates on a particular (and relatively insignificant) portion of the software as a whole. • For instance, rather than attempting to examine the complete design, FTRs are carried out for each component or small group of components.

Review Reporting and Record Keeping

• During the FTR, a reviewer acts as the recorder and takes active notes on all of the concerns that have been brought up. • At the conclusion of the review meeting, these notes are summarised, and a review issues list is produced.

•A review summary report provides responses to the following three questions:

1. What exactly was looked over?
2. Who was the reviewer?
3. What were the results of the study and what were the findings?

• The review summary report is incorporated into the historical record of the project and may be provided to the project leader as well as any other parties that have an interest in the work.

•There are two functions served by the review issues list:

1. To pinpoint sections of the product that are causing issues
2. to act as a checkpoint for action items that acts as a guidance for the producer while he or she makes corrections.

• It is important to design a follow-up method to guarantee that items on the issues list have been correctly remedied. • A issues list is typically attached to the summary report. • A summary report normally includes an issues list. If this is not done, it is possible that the issues that have been brought up would "fall between the cracks."

• One strategy involves handing the obligation for follow-up to the person in charge of the review.

SOFTWARE RELIABILITY

• The statistical definition of software reliability is "the probability of failure-free operation of a computer programme in a specified environment for a specified amount of time."

•What exactly does it mean to fail at something?

–Failure is defined as nonconformance to software requirements in the context of any conversation pertaining to the quality and dependability of software.

It's possible that fixing one mistake will lead to the creation of new ones, which will then lead to new mistakes, which will then lead to new failures.

• The reliability of software can be monitored, guided, and estimated by using historical data in conjunction with development data.

Measures of Reliability and Availability

• Mean-time-between-failure (MTBF), where MTBF = MTTF + MTTR, is a straightforward method for determining how reliable something is.

Mean-time-to-failure (MTTF) and mean-time-to-repair (MTTR) are two different acronyms that refer to the same concept.

• The mean time between failures, or MTBF, is a far more meaningful measurement than defects per KLOC or problems per FP.

To put it another way, an end-user is only concerned with the number of failures, not the total number of errors. The total error count is not a particularly reliable indicator of the dependability of a system because the failure rate of each individual error found within a programme is not the same.

•We need to establish a measure of availability in addition to the dependability metric that we already have.

• Software availability is the likelihood that a programme is working according to the requirements at a given point in time. The formula for calculating availability is as follows: Availability = [MTTF/(MTTR + MTTF)] 100%.

• The sensitivity of the MTBF reliability measure to the MTTF and MTTR is the same. • The availability measure is somewhat more sensitive to the MTTR, which is an indirect indication of the maintainability of software.

Reengineering

Introduction

- Regardless of application size, complexity and domain environment modification occurs.

1. Due to new feature demanded by customer.
2. Due to errors.
3. Due to new technology

- We have to maintain when it is necessary and we have to re-engineer right.
- What is it?
- Who does it?
- Why it is important?
- <u>What are the steps?</u>

❖ Maintenance correct the defects, adopts new functionality as per user needs and changing env

❖ At strategic level BPR identifies and evaluates the existing business process and create revised BP that better meet the current goals

- <u>What is the work product?</u>

❖ Variety of maintenance and re-engineering work products are produced
eg: usecases, analysis and design model, test procedures

❖ The final output is upgraded software

- <u>How do you ensure that you have done right?</u>

❖ Use SQA practices that are applied to every SE process

✓ Technical reviews assess the analysis and design models

✓ Specialized review consider the business applicability and compatibility

✓ Testing is applied to uncover the errors in content functionality etc.

<u>**Re-Engineering Advantages**</u>

✓ Reduced risk

✓ There is a high risk in new software development. There may be development problems, staffing problems and specification problems.

✓ Reduced cost

✓ The cost of re-engineering is often significantly less than the costs of developing new software.

Business Process Re-Engineering

- BPR extends for beyond scope of IT and SE
- Concerned with re-designing business processes to make them more responsive and more efficient.

Business Process:

- BP is a set of logically related tasks performed to achieve a defined business outcome.
- Within the BP people, equipment, material resources and business procedures are combined to produce s specified results.
- The overall business can be segmented as fallow
  Business ->Business System->Business Process->Business Sub-process.
- BPR can be applied to at any level of hierarchy but as the scope of BPR broadens, risk associated with BPR grows dramatically.

BPR Model:

- BPR is iterative model
- BG and process that achieve them must be adapted to changing env.
- For this reason there is no start and end to BPR it is evolutionary process.
- The BPR Model consisting of 6 activities

1. Business Definition:

- Business goals are identified within the context of 4 key drivers
- ❖ Cost reduction
- ❖ Time reduction
- ❖ Quality improvement
- ❖ Personal development and empowerment
- Goals can be identified at the business level or a for a specific component of a business.

2. Process Identification:

- Process to achieve the business goals can be identified.
- After identifying process this can be ranked by importance, by need of change, or in any other way that is appropriate for RE activity

3. Process Evaluation:

- The existing process is thoroughly analyzed and measured.
- Process tasks are identified.

- The cost and time consumed by process tasks can be noted and quality/performance problems are isolated.

4. Process Specification and design:

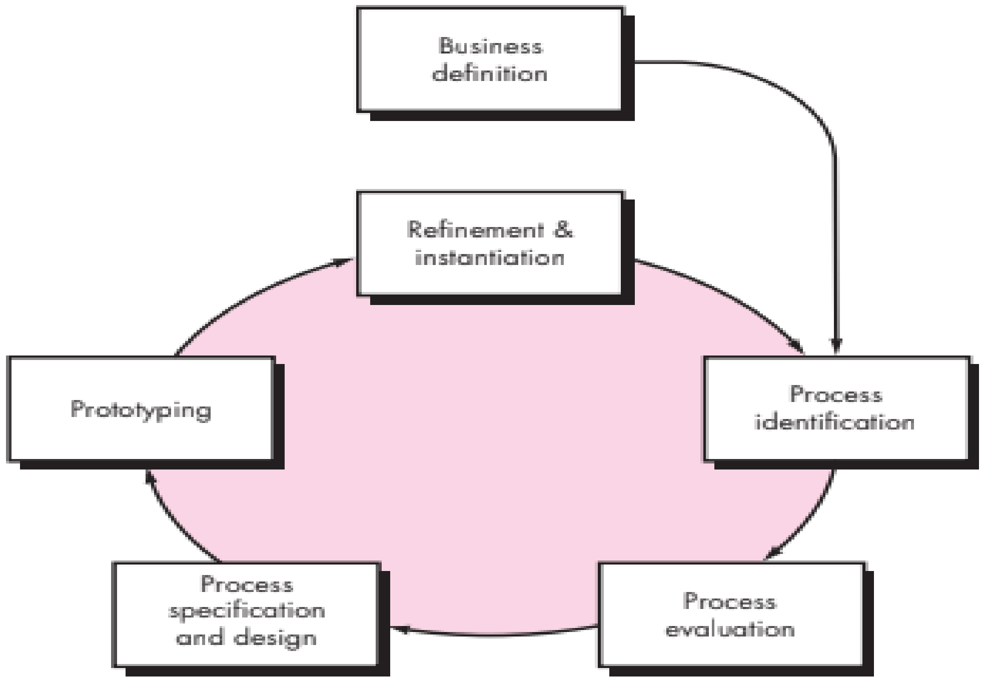
- Based on info obtain by the first three BPR activities use-cases are prepared for each process that is to redesigned.
- Within the BPR use-case identify a scenario that detects some outcome to a customer.
- Within the use-case as the specification of the process new set of tasks are designed for the process

5. Prototyping:

- A redesigned BP must be prototype before it fully integrated into the business.
- This activity test the process that refinement can be made.

6. Refinement and Instantiation:

- Based on the prototype BP is refined and then instantiated with a business system.



- ❑ BPR activities are sometimes used in conjunction with workflow analysis tools.
- ❑ The intent of this tool is to build model of existing workflow in an effort to better analyze existing process.

- The scenario is all to common
- Reorganizing and modifying existing software systems to make them more maintainable.

Objectives:

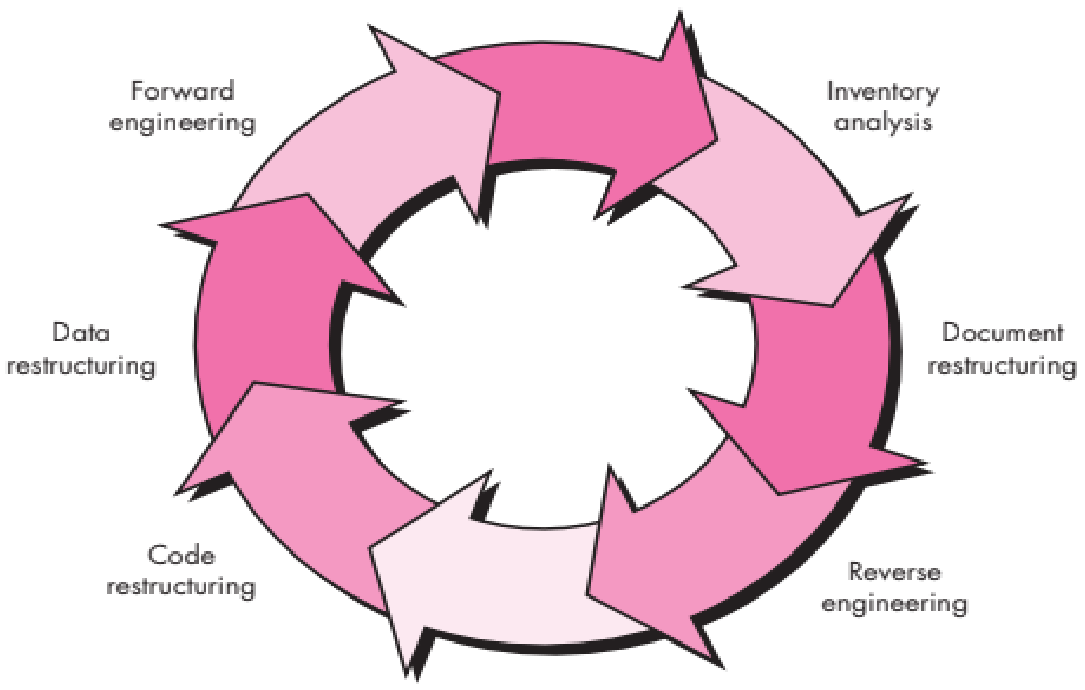
- To explain why software re-engineering is a cost effective option for system evolution
- To describe the activities involved in the software re-engineering process
- To distinguish between software and data reengineering and to explain the problems of data re-engineering

Software Re-Engineering Process Model:

- Re-engineering takes time, it costs significant amounts of money, and it absorbs resource.
- For all these reason re-engineering is not accomplished in a few months or even a few years.
- Re-engineering of information system is an activity that will absorb IT resources for many years. That's why every organization needs a pragmatic strategy for software re-engineering.
- A workable strategy is encompassed in re-engineering process model.
- Re-engineering is a rebuilding activity.

Eg: Rebuilding of a house

- ❖ Before you can start rebuilding, it would seem reasonable to inspect the house.
- ❖ Before you tear down and rebuild the entire house, be sure that structure is weak.
- ❖ Before you start rebuilding be sure you understand how the original was built.
- ❖ If you begin to rebuild, use only the most modern, long lasting materials.
- ❖ If you decide to rebuild be disciplined about it.



Software Re-engineering Activities:

- The scenario is to common to all: An application is served the business need of a company for 10 or 15 years.
- During that type it has been corrected, adapted and enhance many times.
- Re-Engineering activities paradigm is a cyclical model that means each activities presented as a part of paradigm can be re visited.
- Totally we are having 6 software re-engineering activities.

❖ **<u>Inventory analysis:</u>**

- Every software org should have a inventory of all application.
- Inventory is nothing but spread sheet type of document.
- By sorting the info according to business critically, longevity etc..
- Resources can then allocated to candidate application for re-engineering work.
- This inventory should be revised on a regular cycle as a status of application change.

❖ **<u>Document Restructuring:</u>**

- Weak document is trademark for many legacy software. What you can do about it? And what are your opt?

➢ Creating document is far too time consuming:

▪ If a system is working you just go for live with what you have.

- In some cases this is correct approach why because it is not possible to re-create document for hundreds of computer program.
- If the program is static document will end otherwise it wont.

➢ Document must be updated, but your org have limited resources:

- Here we use a “documented when touched” approach it will not necessary to fully document an application.
- Rather than that those portions of the system that are currently undergoing change are fully documented.

➢ The system is business critical and must be fully re documented.

- Even in this case an intelligent approach to pare documentation to as essential minimum.

❖ **<u>Reverse engineering:</u>**

- The term RE has its origin in the hardware world.
- A company disassemble a competitive hardware product in effort to understand its competitor design and many of secrets.
- These secrets can be easily understood whenever the specifications obtained.
- But these documents are proprietary and not available for re engineering.
- Because of this SE derives one or more design and manufacturing specification for a product by examining the actual outputs.
- Sometime RE is done company’s own work to understand the specification.
- Therefore RE is the process of design recovery.
- RE tools extracts data, architecture and procedural design info from an existing system.

❖ **<u>Code Restructuring:</u>**

- The most common type of re-engineering code restructuring.
- Legacy system have solid program architecture but individual modules coded in way that makes them difficult to understand, test, maintain.
- In this case code within the suspect module can be restructured.
- To accomplish this task source code can be analyzed by using restructuring tools.
- Violations in structure programming are noted and then code is restructured(this can be done automatically) or re written in modern language.
- Resultant code can be reviewed and tested to ensure that no anomalies have been introduced.