Software Project Management

Marwadi University

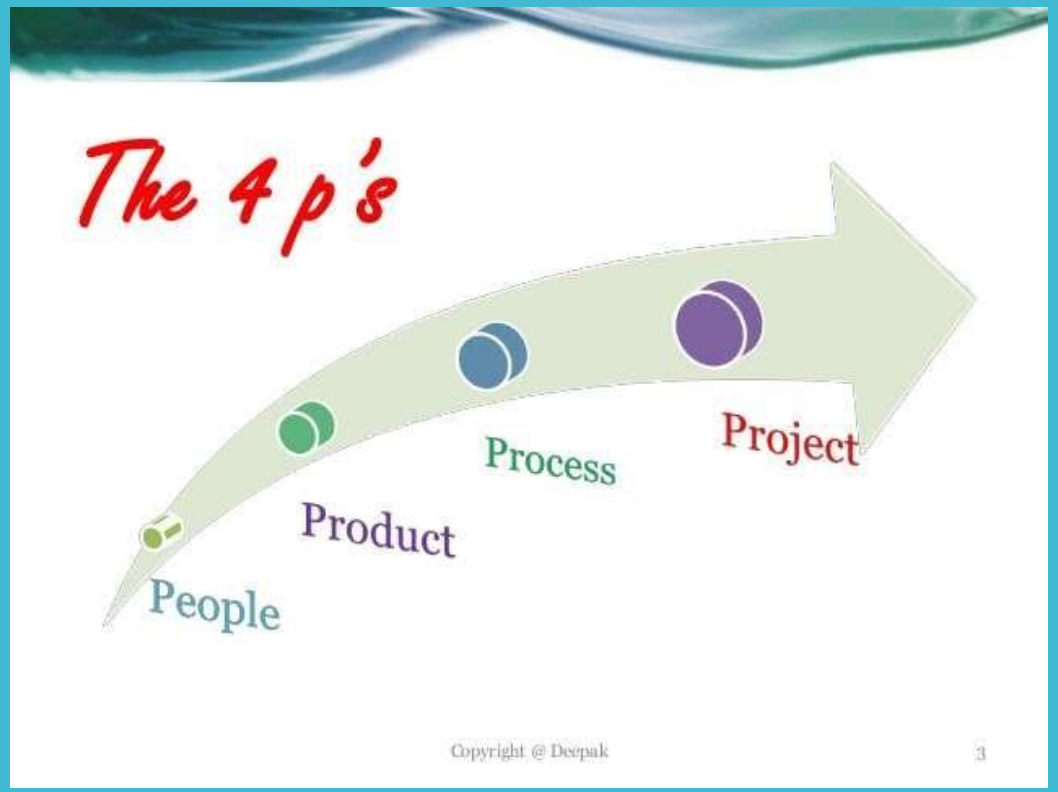
Computer Engineering Diploma

Unit 2:-
Software Project Management

Software Engineering (09CE2402)

Management Spectrum

- The management spectrum describes the management of a software project or how to make a project successful.
- It focuses on the four P's; **people**, **product**, process and project.
- Here, the manager of the project has to control all these P's to have a smooth flow in the project progress and to reach the goal.



(1) The People

- People of a project includes from **manager to developer**, from **customer to end user**.
- But mainly people of a project highlight the **developers.**
- It is so important to have highly skilled and motivated developers that the Software Engineering Institute has developed a People Management Capability Maturity Model (PM-CMM).

The People

- People Management Capability Maturity Model (PM-CMM).

  -- recruiting and selection

  -- managing performance

  -- training and career development

  -- compensation and reward

(2) The Product

- Product is any software that has to be developed.
- To develop successfully, product objectives and scope should be established, alternative solutions should be considered, and technical and management constraints should be identified.
- Without this information, it is impossible to define reasonable and accurate estimates of the cost, an effective assessment of risk, a manageable project schedule that provides a meaningful indication of progress.

(3) The Process

- A software process provides the framework from which a comprehensive plan for software development can be established.
- A number of different tasks sets— tasks, milestones, work products, and quality assurance points— enable the framework activities to be adapted to the characteristics of the software project and the requirements of the project team.
- Finally, umbrella activities overlay the process model.
- Umbrella activities are independent of any one framework activity and occur throughout the process.

Framework Activities

- Communication
- Planning
- Modeling
- Analysis of requirements
- Design
- Construction
- Code generation
- Testing
- Deployment

Umbrella Activities

- Software Project Management
- Formal Technical Reviews
- Software Quality Assurance
- Reusability Management
- Risk Management

(4) The Project

- Here, the manager has to do some job.
- The project includes all and everything of the total development process and to avoid project failure the manager has to take some steps, has to be concerned about some common warnings etc.
- In order to manage a successful software project, we must understand what can go wrong (so that problems can be Avoided)and how to do it right.
- A project is a series of steps where we need to make accurate decision so as to make a successful project.

W5HH Principle

W5HH Principle

- In an excellent paper on software process and projects, Barry Boehm states:

“you need an organizing principle that scales down to provide simple plans for simple projects.”

- Boehm suggests an approach that addresses project objectives, milestones and schedules, responsibilities, management and technical approaches and required resources.
- He calls it the WWWWWHH principle.

W5HH Principle

- Why is the system being developed?
- What will be done?
- When will it be done?
- Who is responsible for a function?
- Where are they organizationally located?
- How will the job be done technically & managerially?
- How much of each resource is needed?

W5HH Principle

#_Why_ is the system being developed?

❖The answer to this question enables all parties to assess the validity of business reasons for the software work. Stated in another way, does the business purpose justify the expenditure of people, time, and money?

#_What_ will be done?

❖This is the guiding principle in determining the tasks that need to be completed.

W5HH Principle

#*When* will be done ?

❖This includes important milestones and the timeline for the project.

#*Who* is responsible for a function?

❖This is where you determine which team member takes on which responsibilities.

#*Where* are they organizationally located?

❖This step gives you time to determine what other stakeholders, customers have a role in the project and where they are found.

W5HH Principle

#How will the job be done technically and managerially?

❖Once product scope is established, a management and technical strategy for the project must be defined.

#How much of each resource is needed?

❖The goal of this step is to figure out the amount of resources necessary to complete the project.

W5HH Principle

- Boehm's W5HH principle is applicable regardless of the size or complexity of a software project.
- The questions noted provide an excellent planning outline for the project manager and the software team.

Requirement Engineering

Requirement Engineering

- **It is the process of establishing the services that the customer requires from a system and the constraint under which it operates and is developed.**
- This includes set of tasks that lead to an understanding of

1. - what the business impact of software will be,
2. - what does the customer want, and
3. - how end-users will interact with the software

Level of Requirement

- **User Requirement:** collection of statements in natural language and description of services the system provides and its operational constraints.

Written for customer.

- **System Requirement:** structured document that gives the detailed description of the system services.

It is a written contract between customer and contractor.

- **Software Specification:** detailed description that can serve as a basis for design and implementation.

Written for software developers.

Functional and Non- Functional Requirements

Requirements

- Requirements analysis is very critical process that enables the success of a system or software project to be assessed.
- Requirements are generally split into two types: **Functional** and **Non-functional requirements.**

Requirements

Functional requirements

- Statements of services that the system should provide, how the system should react to particular inputs and how the system should behave in particular situations.

Non-functional requirements

- Constraints on the services or functions offered by the system such as timing constraints, constraints on the development process, standards, etc.

Functional Requirements

- These are the requirements that the end user specifically demands as basic facilities that the system should offer.
- All these functionalities need to be necessarily incorporated into the system as a part of the contract.
- These are represented or stated in the form of input to be given to the system, the operation performed and the output expected.
- They are basically the requirements stated by the user which one can see directly in the final product, unlike the non-functional requirements.

Non-Functional Requirements

- These are basically the quality constraints that the system must satisfy according to the project contract.
- The priority or extent to which these factors are implemented varies from one project to other.
- They are also called non-behavioral requirements.
- They basically deal with issues like:

1) Portability
2) Security
3) Maintainability
4) Reliability
5) Scalability
6) Performance
7) Reusability
8) Flexibility

Non-functional Requirements

Product requirements

- Requirements which specify that the delivered product must behave in a particular way, *e.g.* execution speed, reliability *etc.*

Organizational requirements

- Requirements which are a consequence of organizational policies and procedures, *e.g.* process standards used, implementation requirements *etc.*

Non-functional Requirements

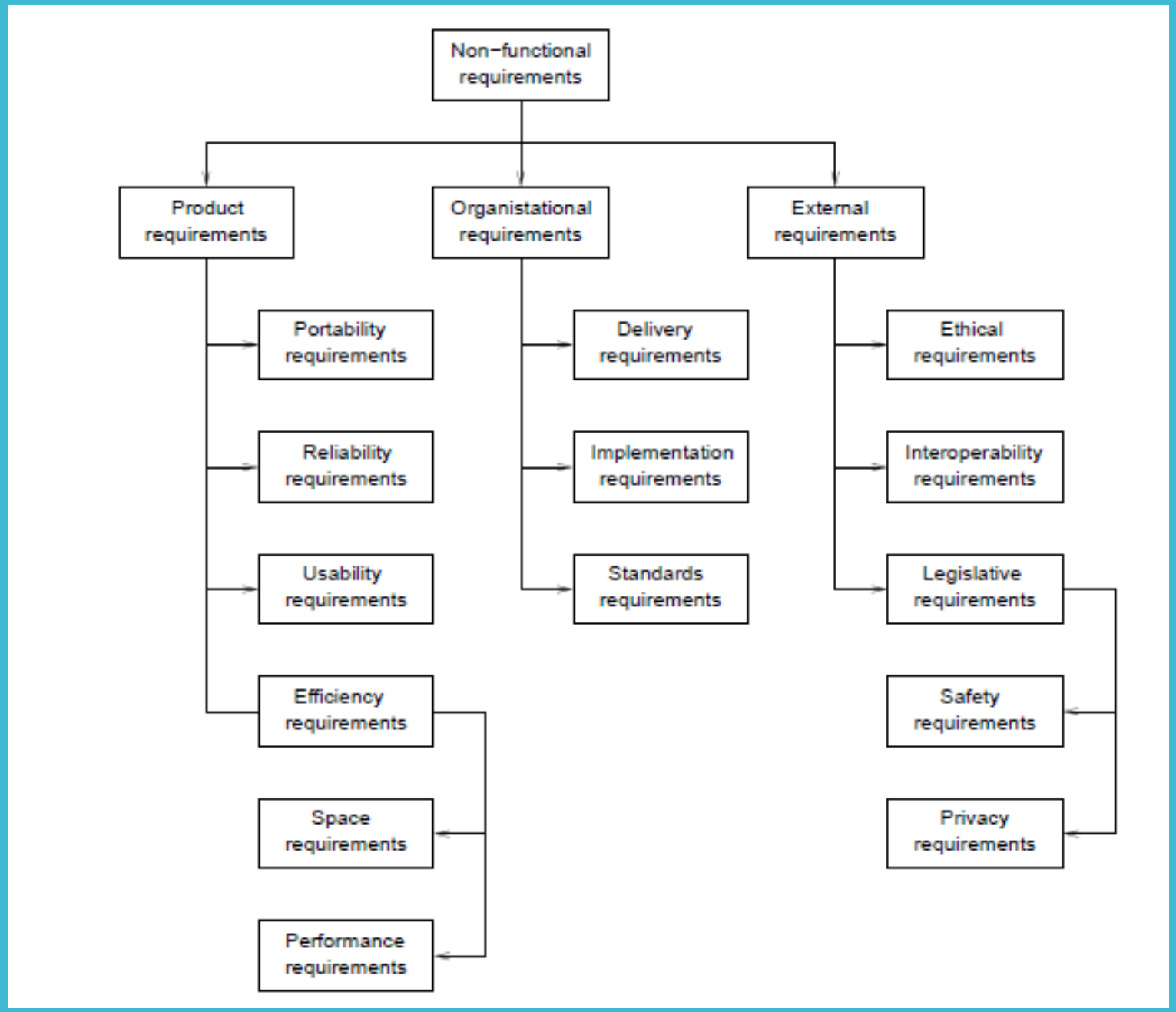
External requirements

- Requirements which arise from factors which are external to the system and its development process, *e.g.* interoperability requirements, legislative requirements *etc.*

Non-Functional Requirements

- They basically deal with issues like:

1) Portability
2) Security
3) Maintainability
4) Reliability
5) Scalability
6) Performance
7) Reusability
8) Flexibility


Non−functional requirements
Product requirements
Organistational requirements
External requirements
Portability requirements
Reliability requirements
Usability requirements
Efficiency requirements
Space requirements
Performance requirements
Delivery requirements
Implementation requirements
Standards requirements
Ethical requirements
Interoperability requirements
Legislative requirements
Safety requirements
Privacy requirements

Difference between Functional and Non-functional Requirements

Sr. No.	Functional Requirements	Non Functional Requirements
1	A functional requirement defines a system or its component.	A non-functional requirement defines the quality attribute of a software system.
2	It specifies “What should the software system do?”	It places constraints on “How should the software system fulfill the functional requirements?”
3	Functional requirement is specified by User.	Non-functional requirement is specified by technical peoples e.g. Architect, Technical leaders and software developers.
4	It is mandatory.	It is not mandatory.
5	It is captured in use case.	It is captured as a quality attribute.
6	Helps you verify the functionality of the software.	Helps you to verify the performance of the software.
7	Functional Testing like System, Integration, End to End, API testing, etc. are done.	Non-Functional Testing like Performance, Stress, Usability, Security testing, etc are done.

Difference between Functional and Non-functional Requirements

Sr. No.	Functional Requirements	Non Functional Requirements
8	Usually easy to define.	Usually more difficult to define.
9	**Example**	**Example**
	1) Authentication of user whenever he/she logs into the system	**1)** Emails should be sent with a latency of no greater than 12 hours from such an activity.
	2) System shutdown in case of a cyber attack.	**2)** The processing of each request should be done within 10 seconds.
	3) A Verification email is sent to user whenever he/she registers for the first time on some software system.	**3)** The site should load in 3 seconds when the number of simultaneous users are > 10000

Functional vs. Non-Functional Requirements

Functional Requirements	Non-functional Requirements
• **Products** The system **shall display** a list of all products offered by the shop. *MustHave* The system **shall organise** the list of products by product category. *MustHave* The system **shall display** detailed product descriptions consisting of name, photograph, price and text of description on demand. *MustHave* The system shall allow the items in the catalogue to be **searched**. *ShouldHave*. The system **shall display** the number of items currently in the shopping basket on each page of the catalogue. *CouldHave* • **Payment** The system **shall accept** all major credit cards. *MustHave* The system **shall validate** payment with the credit card processing company. *MustHave*	• **Capacity** The system shall support 1000 transactions per day. *ShouldHave* The system shall support a peak transaction rate of 10 transactions per second. *ShouldHave* The system shall support 5000 concurrent sessions. *MustHave* • **Availability** The system shall be available 24 hours per day, 360 days per year. *MustHave* The system shall not lose any transaction data. *MustHave* The system shall accept payment and raise an order within 5 seconds in 95% of the cases. *ShouldHave* The system shall log in a customer within 5 seconds. *ShouldHave*

Space

Performance

Availability

Reliability

Thank You!

Metrics in the Process and Project Domains

Software Metrics

- A software metric is a measure of software characteristics which are measurable or countable.
- Software metrics are valuable for many reasons, including measuring software performance, planning work items, measuring productivity, and many other uses.
- Within the software development process, many metrics are that are all connected.
- Software metrics are similar to the four functions of management: Planning, Organization, Control, or Improvement.

Classification of software Metrics

- **Software metrics can be classified into two types as follows:**

1. Product Metrics:

These are the measures of various characteristics of the software product. The two important software characteristics are:

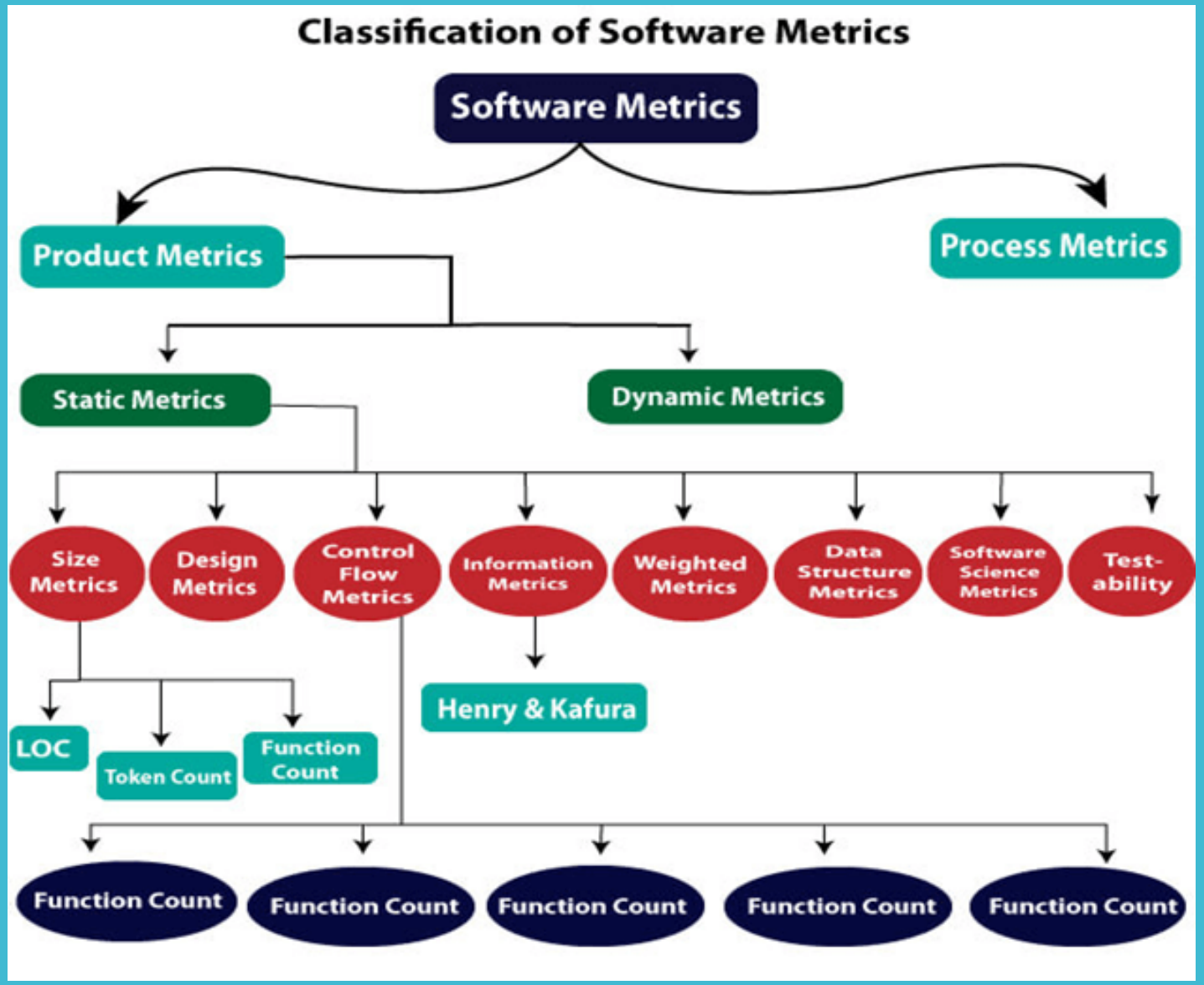
- Size and complexity of software.
- Quality and reliability of software.

- These metrics can be computed for different stages of SDLC.

2. Process Metrics:

- These are the measures of various characteristics of the software development process.
- For example, the efficiency of fault detection.
- They are used to measure the characteristics of methods, techniques, and tools that are used for developing software.

Classification of software Metrics



TYPES OF METRICS

(1) Internal metrics:

- Internal metrics are the metrics used for measuring properties that are viewed to be of greater importance to a software developer.
- For example, Lines of Code (LOC) measure.

(2) External metrics:

- External metrics are the metrics used for measuring properties that are viewed to be of greater importance to the user,
- For example, portability, reliability, functionality, usability, etc.

(3) Hybrid metrics:

- Hybrid metrics are the metrics that combine product, process, and resource metrics.
- For example, cost per FP where FP stands for Function Point Metric.

TYPES OF METRICS

(4) Project metrics:

- Project metrics are the metrics used by the project manager to check the project's progress.
- Data from the past projects are used to collect various metrics, like time and cost; these estimates are used as a base of new software.
- Note that as the project proceeds, the project manager will check its progress from time-to-time and will compare the effort, cost, and time with the original effort, cost and time.
- Also understand that these metrics are used to decrease the development costs, time efforts and risks.
- The project quality can also be improved.
- As quality improves, the number of errors and time, as well as cost required, is also reduced.

Advantages of Software Metrics

1. It makes the better control, planning and clear visibility.
2. It helps to increase the production and quality.
3. With the help of this we can measure the size of the software.
4. We can find out the cost of developed software.
5. With the help of software metrics we can find out the errors which creates problem on the first level of development life cycle.
6. With the help of this we can better control and examine this process of development life cycle.

Disadvantages of Software Metrics

1) The application of software metrics is not always easy, and in some cases, it is difficult and costly.
2) The verification and justification of software metrics are based on historical/empirical data whose validity is difficult to verify.
3) These are useful for managing software products but not for evaluating the performance of the technical staff.
4) The definition and derivation of Software metrics are usually based on assuming which are not standardized and may depend upon tools available and working environment.
5) Most of the predictive models rely on estimates of certain variables which are often not known precisely.

Thank You!!!!