# 2. FOUNDATIONS

Doubt grows with knowledge.

*(Johann Wolfgang von Goethe)*

Contents



In this chapter, terminology and practices from metrics, software components and variation management are covered. Moreover, concepts like metamodeling and paper prototyping are introduced. The content is not intended to be comprehensive. Hence, for introductory readings it is referred to [CN02], [PBvdL05], [LL10] and [Sny03].

## 2.1. Preliminaries to Metrics

Metrics form an integral part of the management and technical activities that are implemented in every *software organization*[1]. Metrics provide organizations with objective and reliable information, which helps to make sustainable decisions that will have positively repercussions throughout their business. The metrics and the information derived from them are treated by software organizations as a valuable asset. This information facilitates in all the organizational levels, from project stage up to management, the decision making.

### 2.1.1. Metrics in the Organization

Metric should not be confused with measure. A *metric* is a quantitative property of products or processes whose values are somehow a representation of certain type (like numbers) [Mey00]. A measure is the value of a metric for a certain product or process.

[1] As a remark, the terms of software organization and organization are used indistinctly overall the presentation of this work and they both refer to software organization.

The usage of metrics is an ongoing process, whose aim is to identify problems and to consider improvements within the software organization. In the past, the measurement process was treated as additional, non-value-added task to accomplish. Nowadays, measurement and analysis activities form a basic software engineering practice. Process improvement approaches, such as the Capability Maturity Model Integration (CMMI) [Sof10], assist organizations to institutionalize the applicability of a measurement process with focus on project management, always assuring the software quality.

For example, CMMI covers metrics on two of its five maturity levels and the Measure and Analysis (MA) process area has the purpose to develop and sustain a measurement capability that is used to support management information needs.

Organizations design their technical and management operations to take advantage of objective metric data. Analysis reports associated with metric data support short and long-term decision making. For example, a mature software organization uses metrics to guide process improvement decisions and investments. The information derived from the analysis of metrics is considered to be a strategic resource to track many issues within the organization. For instance, a project can monitor its current budget against planned. Metric data help to perform better planning and evaluation from a proposed project.

Since software has become a key component in every organization, keeping the pace in the rapidly changing world of information technology surrounded by an increasingly competitive environment is considered a major factor in business strategies and corporate investment. Given the fact of large investment in developing and maintaining critical information assets, it is imperative to objectively assess and manage software projects.

Landis et al. highlight eight key metrics, which contribute to the management of software development projects [LMW+90]. These metrics are:

1. **Source code growth rate** reflects requirements completeness and the software development process.

2. **Effort data** reflect the nature of the project environment and the type of problem being solved.

3. **System size estimates** reflect requirements stability and completeness within the environment.

4. **Computer usage**, which is directly related to the particular process being applied.

5. **Error rates** reflect the total number of errors vs. estimated errors.

6. **Reported/corrected software discrepancies** allow to gain insight into software reliability, progress in attaining test completion, staffing weaknesses and testing quality.

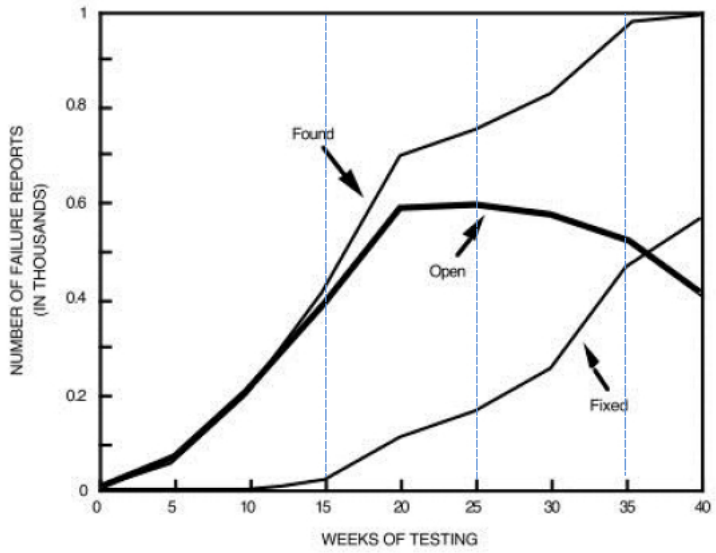


Figure 2.1.: Metric number of failure reports by status

7. **Software change rate** reflects the software development process and stability of project requirements.

8. **Development activity status data** provide insight into the progress of individual development activities.

An example of such metrics is depicted in Figure 2.1. The metric returns the number of failure reports during test with a specific status (found, open and fixed). The information that a manager can abstract from the visualization helps to identify the symptom, determine the cause, take some corrective action and expect the results.

First, the symptom was that early in testing, errors were not getting fixed (first 15 weeks). Since the errors were not found during system testing, software of lower quality was produced and some corrective actions were taken. More developers were assigned at week 20 to help addressing open errors. As a result the system attained stability at week 35, with errors being corrected faster than they were being reported [LMW+90].

Metrics are equally important at the organization level as into the project level. They can provide with key indicators of project achievement, adherence and quality. According to McGarry, metric data as management aid help to deal with the issues listed below [McG01].

**Effective Communication:** Metrics help the decision maker to manage business goals and associated tasks at all levels within the organization and communicate the health of the organization.

**Track Project Plan Goals:** Metrics help to describe the status of the project regarding its processes and products. Metrics also represent the progress of the activities being executed and the quality of their results.

**Risk Management:** Metrics assist with a proactive management strategy. For example, estimations can be analyzed and potential problems could be better evaluated and prioritized. It is known that the earlier a problem is discovered, the less cost it will represent for the organization and less problematic to solve it.

**Elaborate Key Trade-off Decisions:** The projects are subject to constraints and decisions being made in one area, which certainly would impact another different area. These impacts need to be assessed. The results from the assessment can be used to elaborate trade-offs meeting the project goals.

**Rationalize Decisions:** Decision makers, technical and project managers, must be able to defend their estimates and plans with historical data. Metrics provide a solid rationale for selecting the best available option.

It is important to notice, metric data by itself does not guarantee that a project will succeed. Nevertheless, it provides the decision makers with the sufficient information to deal with critical and non-critical issues inherent in projects and to follow a proactive approach. For this reason, metric data supports the projects and consequently the organization, to succeed.

The current enterprise surroundings of software are characterized by changing technology within a very competitive market. Customers demand more functionality and rapid implementation of any feature of their business for a low cost to satisfy their demands. Nowadays, it is getting to be more and more difficult to establish a software organization as a market leader of information technology and still more difficult to maintain it in the top, performing better than its competitors from technical and business perspectives.

In order to achieve such challenging objectives, a software organization must consider the following characteristics [McG01]:

**Objective information made available.** Valuable information is made available to all levels where decision-making needs to be done and its use is part of the organization culture. Issues are openly identified and addressed.

**Historical and estimation data from the projects is made available.** Therefore, it is taken into account to define project goals and performance expectations.

**Business processes are designed to be evolvable.** Informed decision-making and actions follow from up-to-date objective information.

The previous characteristics have one aspect in common: all of them deal with information, hence these characteristics are metric related.

Now, metrics provide a learning structure into the organization. They help an organi-

zation to understand the gaps between how is the current performance from the organization and what are the expected levels. Moreover, metrics allow the optimization of an organization according to its business and technical constraints.

Every project produces deliverables and follows processes. One key challenge in every software organization is to improve the performance in each of its projects. Thus, quality on products and processes is accomplished. The improvement in the performance implies that the time to effectuate processes and elaborate products remains under control. Each project is unique in terms of technical activities being executed, domain application, interdependencies with other systems and several constraints. Each project emerges with distinct characteristics on its process and product.

The organizational metric approach must be adaptable to effectively address the unique information needs and characteristics of each project. Even when the project stage is the adequate level for the implementation of project specific metrics, there are different information needs at organizational level. The information needs from the organization level have a broader scope. A scope that covers the information from all the projects. In order to analyze the overall organization, metrics across many projects can be combined using different techniques to satisfy specific information needs. For example, managers from different projects may be concerned with the number of requirements by status (e.g. open, closed and rejected) from their projects. However, one manager controlling a small project may be interested in monitoring this *numbers weekly*, while other manager in charge of a large project is interested in a *monthly* track of the activities. Nevertheless, the key aspect is the availability of data at the project level and the metric that provides this information only needs to be adapted accordingly.

Usage of metrics had followed many different approaches in the projects. Not all of them have been effective and for those who had passed the first test may not even outlived the project itself. In the worst case, some of the approaches have selected the *best* set of metrics equally applicable to all the projects within the organization. By the term best, it is meant to be repeatable on at least two projects. Another case is the usage of metric analysis tools either available in the market or developed by some vendor with poor knowledge of the organization or lack of understanding about business information needs. The last approach takes into consideration two key characteristics:

1. Metric data that relate directly to the information needs of every project decision maker.

2. A structured and repeatable measurement process that specifies project measurement activities and related information interfaces.

The metric data and the measurement process work together. These are aligned to each project life cycle, providing with different information as the needs of the decision makers change. Both are tailored to meet the specific characteristics of each project. Together they support successful project completion and improve the business and engineering performance.

A measurement process is used to establish, plan, perform and evaluate measurement

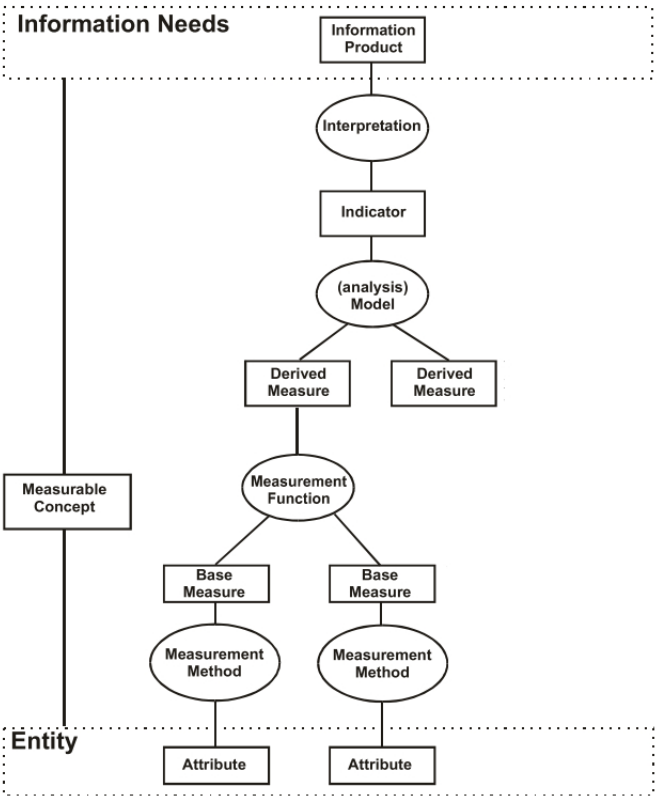


Figure 2.2.: Measurement Information Model

within a project or measurement structure from the entire organization. The measurement process should be flexible, tailorable and adaptable to the needs of different users.

The measurement process is described through a model. This model defines a series of required activities used to understand:

1. What information needs are required.
2. How metrics and analysis results are going to be applied.
3. How to determine whether the results from the analysis are valid.

The model is a required resource within the measurement process [Sta07]. It provides project specific metrics, organized within a structure and relates them to every single information need. It provides an analysis path that supports some sort of advice, which is product of the analysis of historical data. Addtionally, the model supports analysis tasks and measurement planning.

### 2.1.2. Measurement Information Model

The Measurement Information Model is defined in the Standard ISO/IEC 15939 [Sta07] as the *Software Measurement Process*, depicted in Figure 2.2. The standard describes how data is collected and organized to meet required information needs.

The **information needs**, located at the top of the model, are required to support project decision making. The needs are usually related to the project and organization, to the first in an operational sense related to planning and execution and to the second it may address strategic requirements.

The **entities**, placed at the bottom of the model, are conformed by many different process elements and product attributes helping to satisfy the specific information needs.

The **measurable concept**, in the middle of the model, is a way to specify how the data will be measured and put together to outcome with results satisfying the information needs. It is important to explicitly mention what an indicator means. An *indicator*, or interpretation aid, is obtained by following the corresponding analysis model regarding the information needs. The indicators are the basis for overall analysis and decision making within the organization. The goals of the organization define which information needs are applicable to which projects.

In Figure 2.3, an example of measurement construct addressing a common project information need is provided. The information need is related with coding progress. The stakeholder needs to evaluate whether the writing code pace will allow the project to achieve the completion on time [McG01].

The idea of the measurable concept is that coding progress is related to the amount of work planned and work completed. Therefore, the entities of concern are the planned work items and completed work items. The data for the base metric must be provided, whereas data from the derived metric is computed. On this example a simple numerical threshold is used as decision criteria.

Every project defines many information needs, which are used during any point in time for analysis. These information needs are said to be variable, therefore they will be changing during the lifecycle of the project according to certain assumptions, constraints and project goals.

The *Measurement Information Model* is a structure binding information needs to relevant processes, projects, products and/or resources and attributes of concern. A set of relevant attributes are transformed into indicators that provide a basis for decision making.

Once that the Measurement Information Model has been defined, the next step is to ask where to store the metrics. Harrison had intensively studied an approach for maintaining the metrics data into a universal metrics repository [Har04]. This approach suggests treating the metrics and their relationships as data, which represents measurement data of products and processes.

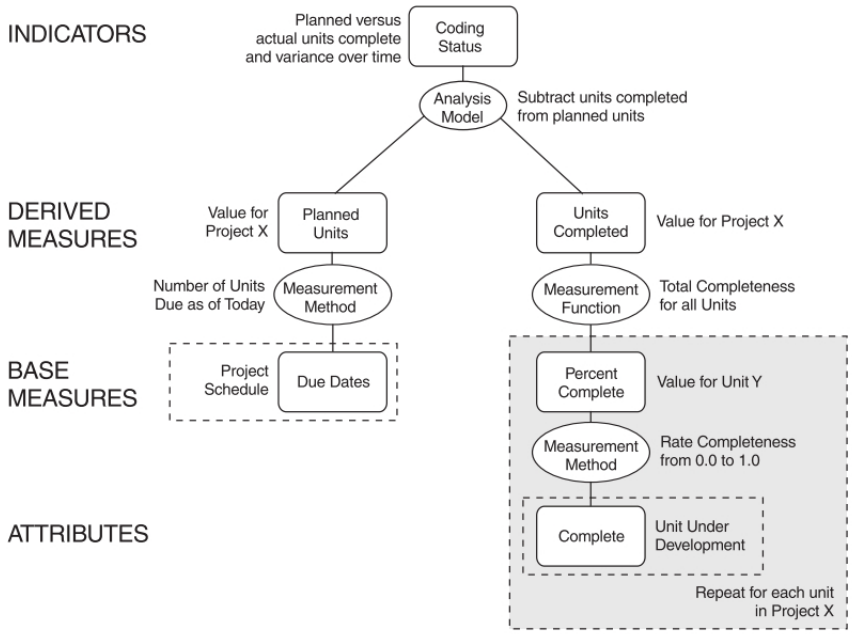


Figure 2.3.: Measurament construct of coding progress

The universal metrics repository using data for representation of measurement data prevents to reach the following characteristics:

**Obsolescence.** One clear example is the metrics that are built into database schemas. This means, by the time that the schema was created, only a set of metrics complying at the time of the creation with the information needs from the organization were being collected. As the information needs are changing, the metrics change as well. With a schema directly attached to the metrics, the creation or removal from metrics leads to incompatibility issues between the schema and the metrics. The modification requires additional effort.

This limitation could be prevented treating the metrics themselves and their relationships as data. This characteristic is called meta-data (see Section 2.2).

**Ambiguity.** It may be often that users from the repository don't have a clear idea about what represent the metric. Documentation from repositories could alleviate this problem. Nevertheless, it is important to mention that documentation needs to be updated every time that changes are performed in the metric repository. Meta-data as characteristic description of the metric imposes the maintenance together and can provide with a useful or meaningful description.

### 2.1.3. Representation of Metrics

The usefulness of a metric representation is given by a combination of at least two elements: its content and its form. The *content* aspect refers to what the metric does or is good for; e.g., the metric *Requirements by number and status* performed quarterly in a small project is unlikely to be highly useful. Similarly, the metric *Change requests by number and status* performed weekly would be of little utility to a big project with duration of one year or more. That is, metrics must be fit for purpose; adapting the contents of the metrics to the needs of its users. Since the needs of the users vary from project to project, it is impossible to *fix* the content of the metrics.

There is a second aspect to utility: the *form* of the metric. This means how the metric is expressed, regardless of what the metric does. A linguistic analogy from the expressiveness can be that the content of the metric is like telling something. What it is said at different times cannot be predicted. The form of a metric is like the language used to say something. It's possible to switch languages, but it is also possible to fix a particular language and say almost anything using it. Similarly, it is possible to choose what language wants to be used to express metrics and express any necessary metrics using it. This has the advantage that all the stakeholders can know how to interpret what it is being said. The selected language has to offer the ability to express with richness of detail any foreseeable meaning. Of course some languages are better than others at expressing very specific things. For this reason, selecting the language to express the metrics is not a trivial task.

The most immediate language used to express metrics is natural language, e.g. German or English. For example, *Earned value* (See Appendix A) has been expressed using English as a metric to measure project progress in an objective manner [Ear60]. Natural language is readable and understandable by humans but hardly intelligible by software tools. In any case, natural language is used to define the metric *earned value*, which can be easily assimilated by a stakeholder. The metric can be used by some tool only after tedious translation work.

A metric is composed of numerous concepts that refer to each other in complicated ways. A metric can be visualized as a net of entities interconnected in between. It is possible to write the metric onto paper and describe it in natural language, but all the references must be maintained by hand.

As an alternative to natural language, a modeling language can be used. Even when the idea might appear odd, the goal is the systematic representation of the metric and its content, e.g. the modeling of the metric *Earned value*. On this scenario, the metric describes how much of the resources have been consumed from the team members at a certain point of time. This can be translated into a measurement model by meanings of a measurement concept, entities constituting the concept and information needs. From this point of view, a metric is a model of the achievements that are conceivably possible, which are sustained by specific information needs.

The link between metrics and information needs just like the Measurement Information

Model states and therefore a suitable modeling language rather than a natural language can be used to express it. This modeling in fact allows accuracy, reduces ambiguity and gains the possibility of expressing the structure of information. The ideal modeling language used to represent metrics must be generic enough so that any conceivable metric can be expressed but, at the same time, concrete enough so that all the metric concepts can be treated with specific semantics. Also, the language must allow the modeling from systematic adaptation of metrics in a very changing environment and at the same way must keep most of the commonalities unmodified.

For example, *Unified Modeling Language (UML)* [Obj10] can be used as a language to express metrics. UML is a standardized general-purpose modeling language in the field of Software Engineering described by a number of class diagrams. By the very nature of object orientation, UML is multi-purpose and at the same time limited to the set of concepts its authors choose to include. This means that the representation of metrics is possible and the usage of UML would facilitate model driven development from a tool that could be supporting the tailoring of metrics.

Another alternative, besides UML, is a domain-specific language (DSL) [Dun94]. This specification language can be used to fit the representation of metrics as well. However for the sake of applicability, which can be limited in comparison to learn the language, the UML is selected as the language included in this work for representation purposes.

## 2.2. Metamodeling

A way to avoid misunderstanding regarding the Measurement Information Model is to establish a shared, agreed-upon set of concepts and terms and use them systematically for every communication. It is worth emphasizing that not only metric concepts and terms must be standardized across the organization, but also the mapping between them. This means that there is small value in establishing the terms to be used and the concepts in play if the relationships between them are not equally defined.

The term *metric* is probably the best example, as it is commonly misused to describe many different measurement concepts. Without concise and consistent terminology, effective communication among stakeholders is impossible. Because decision makers need to fulfill their measurement information needs, consistent measurement terminology is mandatory. This is achieved by the use of *meta-models*, where meta- indicates something *further* or *beyond*. Metamodeling, its characterization and needs are introduced in the next section.

### 2.2.1. Meta-models

The term *meta-model* is a qualified variant of *model*. This recommends that metamodeling is a specific kind of modeling. In fact, metamodeling is the analysis, construction and development of meta-models, which are a qualified variant of models. The differ-