3. RELATED WORK

The more we learn, the more we know about what other data we need and how better to collect it.

(Victor Basili)

Contents

The work starts with the premise that software metrics help project managers and software engineering groups during the software development life cycle to gain insight into the software development processes and products. All levels of project management can use metrics for planning and controlling their own projects. In addition, metrics allow project managers to justify decisions and to discuss the status of the projects with other stakeholders. In order to improve the quality of the processes and the products, many organizations incorporate metrics as part of their software engineering activities. Not only the incorporation of metrics is important but also the understanding of their structure.

3.1. Organizational Metric Program

Build software products on time, within budget and in accordance with quality goals is the objective of every software organization. For that reason, the creation of a metric program is of great importance. Metric programs serve to make sure that the plan is followed, that every project is meeting its schedule, cost and quality targets. It helps to detect problems early, when there is still time to do something about them. Tom DeMarco said once: *"You cannot control what you cannot measure"*. If a project isn't being controlled, it is out of control. There is no viable way to monitor risks or know whether its plans are being carried out. Effective project tracking is hard.

The overall goal is to bring the use of metrics from concept to standard operating procedure throughout the organization. Organizations need tools and techniques to collect and analyze metrics. However, it is clear that metrics are collected only if they meet a specific need or answer an important question. The successful implementation

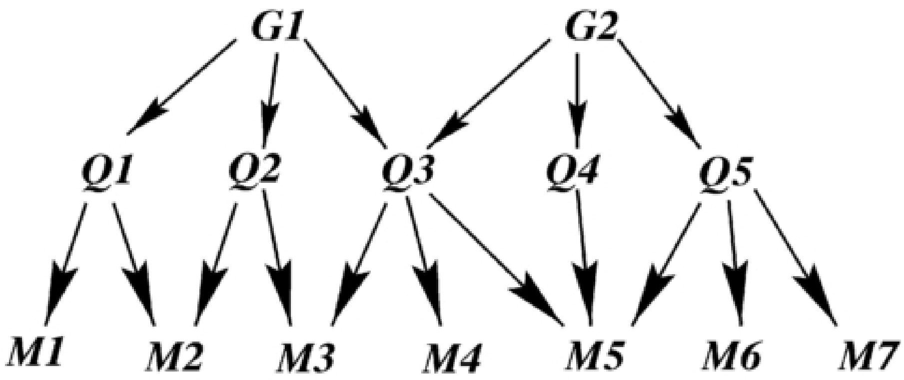

Figure 3.1.: An abstract GQM structure

of any metric program basically relies on the systematically use of certain paradigms and approaches from the theory.

Basili proposed a methodology for collecting valid software engineering data that it is known as the Goal-Question-Metric (GQM) paradigm [BCR94]. Since the metrics from an organization are supposed to provide certain information in a goal-oriented manner, one of the primary questions is the choice of key quality *indicators* (see Section 2.1.2), which will allow focusing on the stated goals for the concrete project.

The GQM paradigm represents a hierarchical model of three levels. It mapps a set of metrics to specific goals. Addtionally control questions are used in order to facilitate the process of evaluating whether the goal is achieved. Figure 3.1 illustrates these three-layer structure. At the conceptual level, the goals of the metric program are established by determining what process or product characteristic requires to be controlled. The operational level is defined as a set of questions refining the quality issue and breaking it down to measurable components. The final step is to define a metrics suite on a quantitative level to measure the data in order to provide interpretable results for the quality estimation.

Plfeeger and Offen et al. followed a two-step approach to establish their metric program [Pfl93] [OJ97]. The first step was to categorize each of the projects by reviewing their environment and operation process followed by an assessment, which finally conducted to the tool selection.

Assessment from the projects. The objective from this step was to categorize and then assess each project from the organization into five maturity levels. The categorization was required because of the different ways in which the projects were operating. Some projects used modern methods and procedures, tool suites and reviews for quality and adherence to standards, while others were poorly documented and barely under control. A standard set of metrics would imply to re-structure the composition of many projects.

Once that project categorization was accomplished, an informal process assessment (using the maturity framework) was applied on each project. The organization did not require a formal maturity framework, like CMMI [Sof10] or SPICE [Joi00]. However, the developed maturity framework was derived from a combination of the CMMI and the GQM paradigm.

Figure 3.2a shows an example of a project with maturity level 2; the process is still hard to identify but the project's requirements are defined and structured. A typical level 3 project, in Figure 3.2b, the process activities are visible. When a project reaches level 4, as shown in Figure 3.2c, it has a central point of control and process measures with feedback to the manager are appropriate. Finally, the most mature projects, shown in Figure 3.2d, can use processes measures and feedback to change the process dynamically as development progresses.

Tool Selection. Tools supporting collection of metrics were provided to the projects. An evaluation of commercially available tools as well as tools provided free to interested users was performed. The result of this evaluation was documented and informed to the managers from each of the projects, which based on their needs and according to their maturity level selected the tool and made them part of their tool suite.

The metrics were stored in individual project databases and in a corporate database. As an example of this starter tool kit, project management measures were recorded in a customized spreadsheet, so managers could track estimated and actual costs and schedules. The managers had the possibility to visualize and track trends from the metrics.

As a result from the metric program implementation, the GQM paradigm was performed using a technique called classification-tree analysis [BFOS84]. Classification-tree analysis allows analyzing relationships among a collection of metrics. A project could be evaluated by looking at snapshots of the project taken over a period of time. It was also possible to evaluate many projects together. A decision tree is generated from a question, with the purpose to reach some goal, provided by a manager. This decision tree shows which metrics are the best predictors of the goal variable.

3.2. Metric Repository

Palza et al. went a step further in creating a metric program by designing and developing an integrated and generic metric repository to support the organization's business information needs [PFA03] [GRCP06]. The aim was to improve understanding, planning and control of metrics.

The key factors that made the successful implementation of the metric repository possible are listed below:

Metrics themselves treated as data. The metric repository integrated a metric meta-

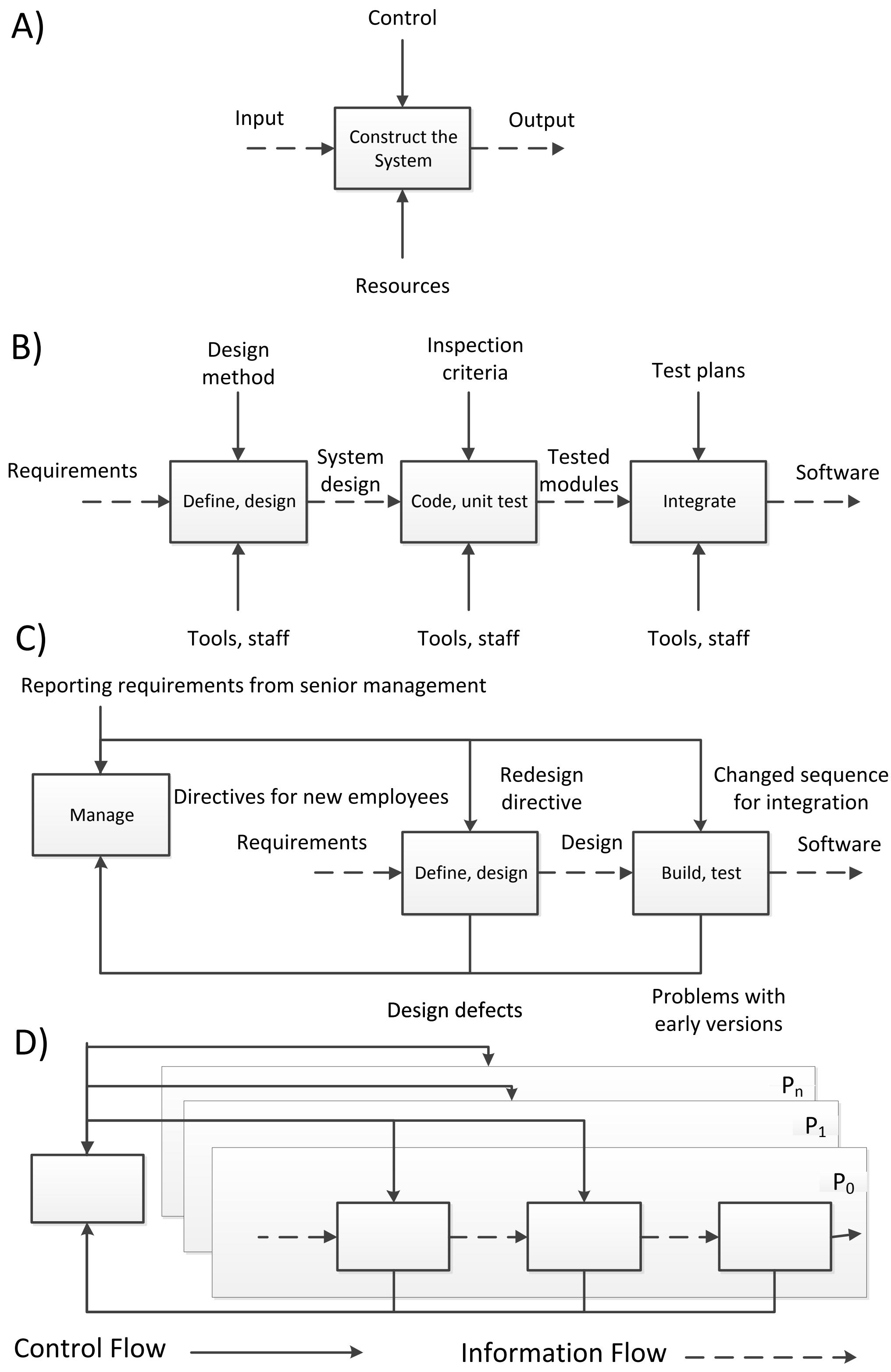

Figure 3.2.: Process maturity examples: A) a repeatable process (level 2), B) a defined process (level 3), C) a managed process (level 4) and D) an optimizing process (level 5)

model that did not presuppose any particular metric nor relationship between them. This property is known as *metadata* (see Section 2.2.1). Metadata permits to store different types of metric data, associated in between, in the context of an integrated environment.

The metadata can be seen as a level of abstraction of the metrics and each entity provided with a high level of flexibility required by the changing needs of the organization. This flexibility to change the definition of metrics was the principal characteristics of the repository; and it was covered by the definition of a metric meta-model structure. Another benefit from this characteristic was the possibility to add or remove metrics at any time without affecting the integrity of the measurement data.

Definition of metrics based on multidimensional modeling. The implementation of metric repositories has emerged within the organizations due to the necessity of improved project control. This control is achieved by using relevant quantitative information for the appropriate estimation of the current project state. Two very popular terms surrounding this repository context are Data Warehousing [KR02] and Business Intelligence (BI) Systems [Apo10].

A data warehouse (DW) is a database used for data analysis and reporting. The features from a DW are based on Structured Query Language (SQL) [ISO08] and Online Analytical Process (OLAP) cubes [CCS93]. Business Intelligence (BI) refers to techniques used in spotting, digging-out and analyzing data. The difference between DW and BI is that BI Systems often use data gathered from a data warehouse.

The technique of dimensional modeling uses the concepts of facts (measures) and dimensions (context) [KR02]. These fact and dimensions are represented as tables. There exist only one fact table that is associated with multiple dimension tables. A fact table is the primary table in a dimensional model where the numerical performance measurements of the organization are stored. Dimension tables contain the textual descriptors of the organization. These tables must include as many meaningful descriptions as possible. It is not uncommon for a dimension table to have 50 to 100 attributes. Dimension attributes serve as the primary source of queries constraints and report labels. Dimension tables are the entry points into the fact table. Robust dimension attributes deliver robust analytic slicing and dicing capabilities. The dimensions implement the user interface to the tool.

An example of this modeling is depicted in Figure 3.3 [KR02]. The fact table consisting of measurements is joined to a set of dimension tables filled with descriptive attributes. This way, the dimension attributes supply the report labeling, whereas the fact tables supply the report's numeric values.

The schema type used when designing the measure data model is the star join schema. In the star schema design, the fact table is located in the middle and is connected to other surrounding dimension lookup tables like a star, just as shown

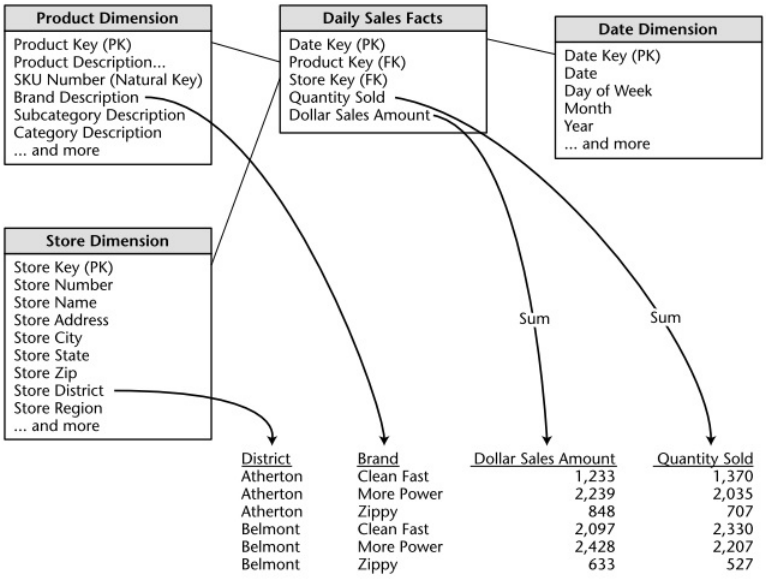

Figure 3.3.: Fact and dimension tables in a dimension model

in Figure 3.3. Each dimension is represented as a single table. The primary key in each dimension table is related to a foreign key in the fact table. All measures in the fact table are related to all the dimensions that fact table is related to.

The multidimensional modeling facilitates the establishment of the collection of metrics and their storage, analysis and reporting according to specific goals and work products. Thus, a mapping between the information needs and the indicators proposed to satisfy those information needs is established. This mapping is done with the objective to help decision makers to evaluate objectively the evolution of the products and processes related to defined information needs on each of their projects.

3.3. Metric Framework

Mendonça et al. leveraged the idea of a metric repository by providing an understanding on the metrics with regards on their structure, which were re-organized in the form of a metric framework [MB00]. Metric framework was then defined as a set of related metrics, data collection mechanisms and data uses inside a software organization.

The approach was conducted following three steps, depicted in Figure 3.4. The first step is the measurement framework characterization. The objective is to identify data user groups and figuring out how they are using the data. The second step, top-down analysis, is based on the GQM Paradigm [BCR94]. On this step, the goals from the different data user groups are captured. Thereafter, the metrics from the measurement framework can be matched. The last step, bottom-up analysis, is based on the *Attribute*

Focusing (AF) technique[1]. It is executed with the aim to extract knowledge from the already existing data. Figure 3.4 also illustrates the information and control flows with dashed and solid lines respectively. Below, the three steps from the approach are explained in more detail.

Metric framework characterization. This step is known as "characterization" because it was the starting point from the whole approach, which either followed a top-down analysis or bottom-up. The objective was to identify metrics, attributes, data, user groups and data uses of the metric framework. A set of structured interviews and reviews from the existing metric framework documentation was used. The characterization followed a four-step process:

1. Identify metrics. The first components to identify are metrics which were already contained in the metric framework, including how they worked (their measurement instrument, scale and value domain).

2. Identify available data. This data included information about when the metrics were used, how these could be accessed and what data was available in the repository.

3. Identify data uses and user groups. The third type of component referred to data uses. Such information included which users were using the data; the frequency of the usage and the relevance of the data for the user groups.

4. Identify attributes. On this step, the attributes conforming the metrics are identified and analyzed.

Top-down analysis. The purpose is to analyze the goals from the different data user groups and map them to the data that is being collected. This helps to gain a better understanding of the information needs. The GQM Paradigm is used to build a structure that maps the goals to the metrics. The analysis follows a three-step process:

1. Capture user group goals. A structure interview is used to know the achievements that want to be reached. These achievements are, later on, expressed in the form of GQM goals.

2. Identify relevant entities and attributes. The next step is to identify what are the entities and attributes that the data user group wanted to measure to reach their goals. A list of entities and attributes in the form of checklist is prepared and reviewed by the user group.

3. Map attributes to existing metrics. Finally, metrics and attributes used in the organization are mapped. An attribute is a single characteristic of all entities in a particular entity class, e.g. "size" of source code. Metric is the mapping model used to assing values to a specific attribute of an entity

[1] AF technique focuses a domain specialist on a small, potentially interesting part of a large amount of data in a manner that leads him to discover useful, interesting and nontrivial knowledge [Bha94].

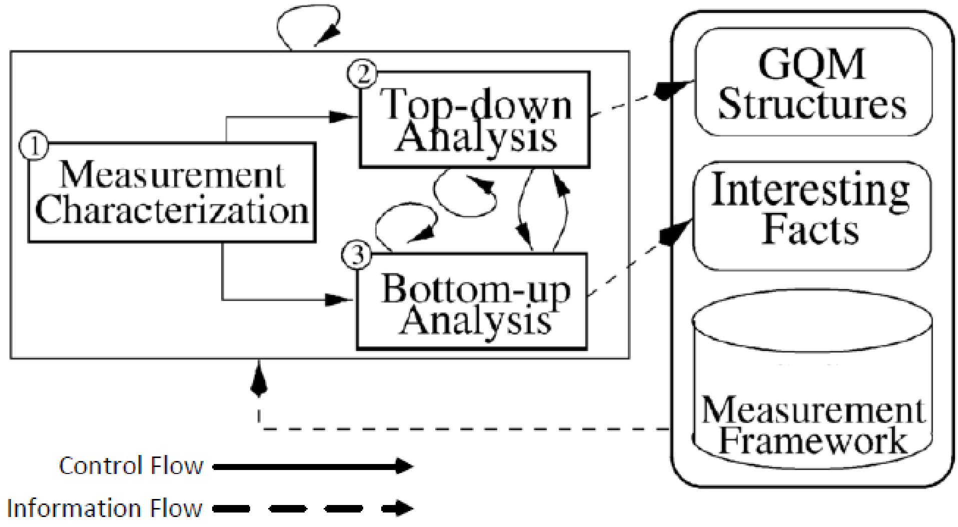

Figure 3.4.: The metric framework and its approach

class. A metric states "how" to measure something. At this step, a GQM structure is created for the user group.

Bottom-up analysis. The aim from this step was to discover new and useful information in the existing data. A data mining technique called Attribute Focusing (AF) [Bha94] was used. This technique is started by the responsible for performing the analysis. This person interacts with the data users to abstract the data information that these users require. The documented information, result from the interviews, feeds the AF Tool. The tool generates some diagrams, which are interpreted by the same person that started the process. Another review is performed with the data users and that is when knowledge discovery takes place.

The approach permitted a better understanding about the metrics, the data and, how these were fulfilling the business information needs.

3.4. Discussion

The literature revealed the scope and diversity of software metrics within organizations as well as the approaches followed in order to keep these metric programs.

The importance of a metric program into the organization was first demonstrated [Pfl93] [OJ97]. Even when the establishment of the metric program showed benefits and succeed in their objective, some drawbacks were identified.First, the organization did not have a standard process, language, environment, development philosophy, or set of procedures to build the software. This was reflected by the selection of multiple tools and the necessity to store the metrics in individual project databases [Pfl93].

The major benefit from a metric repository is the existence of a complete solution for the organization, which subsequently provides project specific solutions [PFA03] [GRCP06]. The implementation based on a metric meta-model seem to be feasible for flexibility on the structure and content. Also metric repositories offer a more clear control over projects by accurate quantitative presentation of the project information needs. The fact that the repository obeys the structuring of multi-dimensional data (in some of the cases), conducts to have repeated attributes in the dimensional tables associated with different fact tables. This could cause overhead for configuration, when applying it on trivial projects.

A metric framework permitted a better understanding and structuring of the metrics [BCR94]. Even when the conducted research produced very positive results, there is no tool supporting the approach. As deficit of the framework, the absence of explicit concept for the constitution of the metrics and their variation management is noticed.

Metric framework is one key aspect that this work focuses on when dealing with variation management. The current work will treat metric framework from another perspective, whereas the work covered in this chapter treated it as a repository. Moreover, the metric framework will require a meaningful methodology approach.