Electronic Research Archive of Blekinge Institute of Technology
<http://www.bth.se/fou/>

This is an author produced version of a journal paper. The paper has been peer-reviewed but may not include the final publisher proof-corrections or journal pagination.

Citation for the published Journal paper:

Title:

Author:

Journal:

Year:

Vol.

Issue:

Pagination:

URL/DOI to the paper:

Access to the published version may require subscription.

Published with permission from:

A Decision Support Framework for Metrics Selection in Goal-Based Measurement Programs: GQM-DSFMS

Cigdem Gencel^a, Kai Petersen^{*,b,c}, Aftab Ahmad Mughal^d, Muhammad Imran Iqbal^b

^aFree University of Bolzano, Faculty of Computer Science, Bolzano, Italy

^bBlekinge Institute of Technology, School of Computing, Karlskrona, Sweden

^cEricsson AB, Box 518, SE-371 23, Karlskrona, Sweden

^dAuckland University of Technology, New Zealand

Abstract

Software organisations face challenges in managing and sustaining their measurement programs over time. The complexity of measurement programs increase with exploding number of goals and metrics to collect. At the same time, organizations usually have limited budget and resources for metrics collection. It has been recognized for quite a while that there is the need for prioritizing goals, which then ought to drive the selection of metrics. On the other hand, the dynamic nature of organisations require measurement programs to adapt to changes in the stakeholders, their goals, information needs and priorities. Therefore, it is crucial for organisations to use structured approaches that provide transparency, traceability and guidance in choosing an optimum set of metrics that would address the highest priority information needs considering limited resources. This paper proposes a decision support framework for metrics selection (DSFMS) which is built upon the widely-used Goal Question Metric (GQM) approach. The core of the framework includes an iterative goal-based metrics selection process incorporating decision making mechanisms in metrics selection, a pre-defined attributes/metrics repository, and a traceability model among GQM elements. We also discuss alternative prioritization and optimization techniques for organizations to tailor the framework according to their needs. The evaluation of the GQM-DSFMS framework was done through a case study in a CMMI Level 3 software company.

Key words:

Software Measurement Program, Goal Based Measurement, Goal Question Metric, GQM, Decision Support, Optimization, Prioritization

1. Introduction

Software organizations have been aware of the significance of measurement processes in taking informed decisions to better manage software processes, products and projects. Measurement is itself regarded as a driving force for software process improvement [1]. It also facilitates effective communication between software organizations and customers [2].

*Corresponding author

Email addresses: cigdem.gencel@unibz.it (Cigdem Gencel), kai.petersen@bth.se (Kai Petersen)

A number of frameworks, models and standards have been developed to support software organizations in planning their measurement programs. The main idea behind most of these approaches is to link measurement needs to organizational goals, strategies and/or information needs of decision makers, and thereby to develop cost-effective measurement programs by collecting only useful data with clear purposes. A large number of empirical studies in different contexts have shown that goal-based measurement frameworks are both rigorous and adaptable [3]. Among these, Balanced Score Card (BSC) [4], Goal Question Metric (GQM) [5, 6], Model Measure Manage Paradigm (M3P) [7], Goals Questions Indicators Measures (GQIM) [3], Decision Maker Model [8], Measurement Information Model in ISO/IEC 15939 [2002] standard, and GQM+ Strategies [9] are some of the well-known ones.

However, even though such approaches help in planning well-established measurement programs at the beginning, managing and sustaining established measurement programs becomes challenging over time due to various reasons (see Section 2). The number of goals and metrics to be collected explode increasing the complexity of the framework [10][11]. Furthermore, measurement programs need to adapt to dynamic environments where the size of the organization, the number of stakeholders, their goals and priorities change [11]. If a measurement framework evolves without continuously establishing and aligning the links between the goals of different stakeholders in an organization, their information needs and the measures, this might eventually lead to poor use of data and loss of global understanding of the outcomes regardless of how well it was planned at the beginning [12].

On the other hand, what can be collected is limited by the budget, time and resources that organizations are willing to spend on establishing a measurement program. These constraints might change over time as well. Previous research emphasized not to collect metrics which provide only little value to the organization, while requiring high effort in collection. Value here refers to contribution of metrics in fulfilling the information needs of the stakeholders [12]. This is particularly essential, given that they map the organizational and project goals to the metrics collected in measurement programs (cf. ISO/IEC 15939). Niessink and van Vliet emphasized the importance of the mapping, stating that *“a value-generating measurement program requires the attention be given to the proper mapping of some identifiable organizational problem or goal onto the measurement program, and a translation back of measurement results to organizational actions”* [13].

This paper proposes a *Decision Support Framework for Metrics Selection* built upon the widely used GQM approach (GQM-DSFMS). GQM-DSFMS support practitioners continuously in making informed decisions for which metrics to collect considering the available budget as well as focusing efforts on the most important goals, and thereby most valuable metrics. In particular, this framework help in highly dynamic environments where organizational goals, constraints, information needs and their priorities change over time. GQM-DSFMS extends GQM by: 1) definition of a structured and guided iterative process for identifying and deciding on the metrics, and 2) incorporation of engineering mechanisms, methods and techniques for establishing traceability among GQM elements, for prioritization of goals and inheritance of the priority values to metrics, and for deciding on an optimum set of metrics.

The GQM-DSFMS process consists of mainly four phases; goals elicitation, question elicitation, questions/attributes/metrics mapping and decision making. For each phase, we defined example mechanisms, methods and techniques that could be used for importance rating of the goals and inheritance of the importance values down to metrics and continuous optimization. We also provided guidelines for each phase of the process so that organizations can customize the process and use in their own context.

A case study has been conducted evaluating GQM-DSFMS with respect to the total cost for measurement collection and the extent of achievement in fulfilling information needs for measurement goals. We compare expert opinion (selection of metrics without decision support) with the results provided by GQM-DSFMS.

In Section 2, we present the related work on the challenges of measurement programs. In Section 3, we first identify the requirements of an integrated framework that would support decision making in selecting the optimum set of metrics when planning measurement programs. Then, based on the identified requirements, we introduce an iterative process for identifying and selecting metrics as well as mechanisms and methods that the GQM-DSFMS comprised of. Section 4 presents the case study design. Section 5 provides the details of the case study conducted in a CMMI Level 3 company to evaluate the developed decision support framework. Section 6 discusses threats to validity. Section 7 discusses the results and implications of this study. Finally, the conclusions are given in Section 8.

2. Related Work

Mendonça and Basili [12] defined the properties of a good measurement framework as *soundness* (when its metrics and measurement models are valid in the environment where they are used), *completeness* (when it measures everything that its users need to achieve their goals), *leanness* (when it measures what is needed and nothing else) and *consistency* (when its metrics are consistent with the user goals). Below, we summarize the reasons we identified from literature for not achieving these properties well.

Poor buy-in and commitment. The benefits of a measurement program in improvement and informed decision-making should be made clear across the organization. Without being able to show the value of data collection to all stakeholders, it is hard to motivate data providers to collect correct and up to date data, and other stakeholders such as top management to continue supporting the measurement program (cf [9]).

One common recommendation in literature is to communicate the results obtained through data collection to stakeholders who needed the information [14]. Hence, lobbying, communicating and training people for metrics collection should be done only for the metrics that really matter for the organization, i.e., no less no more. Otherwise, this would mean not only high effort for measurement program planners, but little commitment from the stakeholders, as well. These require organizations to be aware of why and how much each metric matters to an organization so that measurement programs could stay complete and consistent over time.

Narrow process definition and guidance. During initial planning, existing goal based measurement approaches state what information (such as goals, strategies, tactics to identify information needs) should be elicited to decide on the metrics to be collected. Offen and Jeffery [7] proposed an extension of GQM, called M3P, by adding extra steps to make the links between the collected data and the surrounding business and development context explicit. They introduced steps for understanding business strategies, identifying business goals, sub strategies, risks and tactics, and determining critical success factors. Recently, Basili et al. [9] introduced GQM+Strategies to explicitly link software measurement goals to higher-level goals and strategies of an organization.

However, measurement frameworks evolve over time as new needs arise. The current goal based approaches neither explicitly define how to, nor provide supporting guidelines for maintenance of measurement frameworks. Furthermore, Mendonça et al. [15] maintained the need

to allow a bottom-up approach as well as a top-down approach to avoid ignoring or overlooking useful legacy data in an organization. Olsson and Runeson [16] proposed V-GQM, a bottom-up structured analysis approach extending GQM, which can be used in an iterative way after a measurement framework is planned and instantiated following standard GQM. Therefore, there is a need to define a process which would support organizations not only in planning a measurement program from scratch (when no measurement program is in place), but also modifying or replacing an existing one without losing legacy data already accumulated in an organization while keeping the consistency and leanness properties.

Lack of mechanisms to save cost. Collection of metrics cost and organizations have limited resources for establishing measurement programs. Boyd [17] suggested that the amount of metrics should be limited, and goals and questions could be reused to reduce cost for managing the measurement program. Card [18] pointed out how quickly a measurement framework can grow out of control due to lack of reuse of goals, questions and metrics. In [19], an operational process was defined for goal-driven definition of metrics so that organizations could better justify, interpret and reuse metrics. Ackermann et al. [20] also mentioned about reuse support and discussed reusable measurement patterns for successful software measurement programs.

Staron et al. [2009; 2011] defined procedures to enable reuse of metrics and automate collecting and combining metrics. Structured templates have been introduced to help in making the attributes of goals and questions explicit [3, 9]. However, these templates are difficult to maintain over time due to lack of formal traceability models. This also makes it very difficult for the organizations to use automated tools to save time, cost and enhance reusability.

Lack of resources for measurement expertise. According to Lavazza and Barresi [2005], software measurement programs fail due to lack of expert judgment for selecting appropriate number of metrics in relation to organizational goals. Two out of ten problems leading to failure in the implementation of software measurement programs are reported as the intensive use of a single measure or, conversely, the use of too many [21].

A recent study on software measurement education revealed the significance of the problem. Villavicencio and Abran [22] found out that software engineering graduates do not gain adequate knowledge on measurement as measurement topics are covered mostly only at the graduate level, and often are not mandatory.

In addition to lack of measurement experts, there is no commonly agreed upon pre-defined set of metrics that organizations can utilize when selecting metrics. A number of studies such as [23, 24, 25, 26, 27] investigated the theoretical and empirical validity of metrics. Recently, the results of a systematic literature review made to assess the trends and current status of software metrics, are presented in [28]. Significant problems were observed in empirical validation studies, such as evaluating theoretically invalid metrics or failing to give information regarding the context in which the data were collected.

To the best of our knowledge, there exists only ISO/IEC 25021 standard [29] and ISO9126 standard family [30, 31, 32] that provide a defined set of metrics for software product quality evaluation. Therefore, there is a need to have pre-defined set of standard metrics which would help organizations to decide on collecting standard metrics when planning measurement programs. This would also in turn enable re-usability of metrics as well as aggregation of metrics. It is reported that Nokia lowered the cost of their measurement program by using a predefined set of metrics from a metrics library[33].

The problem of lack of resources becomes more significant for the small to medium enterprises (SMEs), which are more disadvantaged in comparison to mature and larger organizations regarding their software measurement knowledge, experience and infrastructure. Díaz-Ley et

al. in their studies (c.f. [34, 35]) stated that a measurement program was destined to fail if it was not adapted to the measurement maturity level of the company. They investigated the link between the measurement program and the measurement maturity level of an organization and developed a methodological framework for defining software measurement programs with a focus on SMEs. Wangenheim et al. [36] developed a lightweight GQM approach that aims to reduce the measurement overhead of SMEs by integrating systematic reuse of context specific quality and resource models.

Therefore, there is a need to define mechanisms to support not only large and mature organizations that already have established measurement programs, but also for SMEs who would like to benefit from measurement programs for improving quality.

Complexity. The dimension and the complexity of the measurement programs change over time. With changing stakeholders in dynamic environments, horizontal and vertical dependencies among goals, information needs and metrics also change [10]. Therefore, even if the measurement program is well planned at the beginning, it is highly likely that over time these might lead to collection of redundant data, loss of global understanding, collecting data that might be useful to people who are not aware that such data exist, or poor use of collected data to contribute to decision-making at the upper levels [12].

Therefore, a traceability mechanism that would allow organizations to always see the big picture over time is very important. The links between GQM elements should be transparent and traceable [37] so that informed decisions on metrics selection could be made when stakeholders, their goals or priorities, or constraints of an organization change,

3. A Decision Support Framework for Metrics Selection: GQM-DSFMS

To address some of the above mentioned challenges software organizations facing, we developed a *Decision Support Framework for Metrics Selection* built upon the widely used GQM approach [6]. We chose GQM approach as it has been shown to be rigorous, adaptable to many different environments, and to align with organizational directions and goals in many empirical studies [10, 3].

Below, we present the mapping between the identified requirements for such a framework and our proposed solutions in GQM-DSFMS:

- *Collect metrics:* There is a need to have a pre-defined set of widely used and standard metrics, which organizations can make use of at the initial planning phase when deciding on the metrics. In GQM-DSFMS, an Attributes/Metrics Repository with the associated data model is introduced to help in collecting *sound* metrics for defined software attributes.
- *Collect all metrics to achieve the stakeholder goals and keep them consistent in time:* There is a need to enable traceability and transparency, which would support cross organizational alignment and continuous establishment of the horizontal and vertical dependencies between GQM elements. This would in turn also help in combining and/or aggregating the metrics in an organization to make better use of already collected data. In our decision support framework, a Traceability Model is introduced, which assures that each goal raised has to have a linked metric in the end (i.e. no goals without metrics) and that there is traceability and transparency of links between all GQM elements, which would lead to a *complete* and *consistent* framework over time.

- *Collect only what is needed and nothing else:* There is a need to capture the priorities of metrics before deciding on which ones to collect as organizations have limited resources. The priorities of the metrics are dependent on the priorities of the goals of the stakeholders in an organization as well as of the information needs to meet these goals. Therefore, the priorities of the goals should be captured and in some way be inherited to measurement needs before deciding on the metrics. Thereby, the metrics that are more important are given the highest priority. The framework makes use of engineering mechanisms for continuous optimization of metrics selection under the constraints of an organization (e.g. maximum budget set) and considering the importance of metrics. The pre-requisite for being able to reassess the importance of metrics is a traceability mechanism that allows determining which metrics fulfill which goal(s). The traceability is also necessary to inherit importance values from goals to metrics.

The main idea behind the GQM-DSFMS is to continuously optimize the selection of metrics for a given budget available for the measurement program by prioritizing goals and keeping track of the traceability of metrics to goals. The priority of goals derive the importance of metrics, while the metrics have an associated cost. Both pieces of information (priorities and cost) are then used to make an optimal choice of metrics. In particular, this framework would help in highly dynamic environments where organizational goals, constraints, information needs and their priorities change.

In previous work, we introduced a heuristic approach for supporting metrics selection (cf [38]). However, this approach was based on heuristics, requiring high effort in reasoning on metrics selection, and quickly becoming too complex as the number of goals and metrics increase. In particular, we observed that this to be a challenge in GQM+Strategies where goals and metrics are elicited on multiple levels [9]. To the best of our knowledge, no further integrated approach have been defined to support such informed decision making in metrics selection, considering the available budget and focusing efforts on the most important goals, and thereby metrics.

To this end, we first defined a process that extends the GQM process by adding extra steps to support the use of the defined Traceability Model and the Attributes/Metrics Repository, and iterations for selecting an optimum set of metrics under given constraints. An overview of the process is shown in Figure 1. The process consists of mainly four phases and there defined example mechanisms, methods and techniques that can be used for importance rating of the goals and inheritance of the importance values down to metrics and continuous optimization. We also provided guidelines for each phase of the process so that organizations can define practical mechanisms for using the defined process.

In order to provide all information needed by practitioners to apply the process and the framework (i.e. use as a manual), we included a running example in Appendix A and referred to relevant parts from the text. The running example for Phase 1, 2, and 3 is shown in Appendix A.1.

3.1. Phase 1: Goals Elicitation

In this phase, long-term (strategic) and operational (short-term) goals of an organization are elicited, documented and their importance values are rated. Then, the associated measurement goals are identified. See the running example for Phase 1 in Appendix A.1.

Figure 1: GQM-DSFMS Process

3.1.1. Step 1: Identify Goals and their Relationship

Eliciting goals requires first identifying the characteristics of the organizational unit and the stakeholders. For eliciting goals, a process should be defined based on the type and size of an organization. As this has to be customized to the needs of the organization, QQM-DSFMS framework provides some suggestions. For example, the measurement analysts can use techniques similar to those for software requirements elicitation process used during interviews, focus group meetings, or workshops. Then, using a goal definition template, the necessary information about the goals are captured. Examples of goals definition templates exist in literature, such as in [9].

QQM-DSFMS distinguishes two types of traceability requirements for capturing dependencies among goals: vertical and horizontal [10]. Vertical dependencies capture relationships between long-term and short-term goals. For example, a strategic goal “be quality leader as a provider for software services in the e-commerce domain” should be traceable to short-term goal, e.g. “We want to reduce the number of customer reported defects by x%”. Horizontal dependencies indicate that one goal cannot be fulfilled without the fulfillment of another goal at the same level. For example, the strategic goal “be quality leader as a provider for software services in the e-commerce domain” cannot be fulfilled without the goal “provide multiple support channels (e.g. e-mail support, hotline, web-form) for the customer”.

Figure 2 shows the defined Traceability Model. The dependencies among the goals and sub-goals as well as the links to Questions and Metrics are established using this model and capturing traceability information in Goals and Questions definition templates (e.g., for goals through assigning a unique identifier, version number, relevant business goal(s), measurement goal(s) and question(s)).

Figure 2: Traceability Model of QQM Elements in QQM-DSFMS

3.1.2. Step 2: Rate Importance of Goals

Goals are of different importance in organizations and so are the measurement requirements. One organization might highlight quality in order to gain a competitive advantage, while another prioritizes time to market over quality.

In QQM-DSFMS, the importance of the goals is captured in order to inherit these importance values to the metrics. This is an essential input to the decision making phase (Phase 4) where optimization takes place considering the trade-off between importance value and cost of candidate metrics under a budget constraint. For example, if time to market is the most important metric, lead time would be an important measure to collect, and should be preferred over measures related to less important goals.

The importance of the goals can be rated using different approaches [39]. There are mainly two alternative approaches. Below, we provide some example techniques for each approach:

Approaches without trade-off between the goals. Different scale types can be used to rate the goals independently of each other; i.e., goals are not in direct competition with each other. For example, a numerical scale of 1 to n can be used where 1 is unimportant and n is very important.

Or, grouping can be used where items are grouped into categories according to their importance (e.g. Planning game defines three categories as critical, significant and nice to have).

Approaches with trade-off between the goals. Some example techniques which can take into account trade-offs between goals are as follows:

- Analytical Hierarchical Process (AHP): AHP [40] is a prioritization method to compare hierarchically classified items (e.g. requirements, goals, and so forth) with each other. Thereby, every pair is compared on a nine-point scale. Overall, $n * (n - 1)/2$ comparisons have to be made, which makes the method insufficient to be used when many items have to be prioritized. However, the method allows checking consistency in the prioritization.
- Cumulative Voting (CV): CV was first introduced in [40]. This approach is also called 100-dollar or 1000-dollar method. Here, weights are assigned to prioritization items by distributing a specific number of points between the items. Items of higher importance receive more points. For example, when three items are prioritized and the first item receives as twice as many points as the other items, then this item is twice as important as the rest (see example below).

Item 1: 50 points

Item 2: 25 points

Item 3: 25 points

Prioritizing items in hierarchies (called Hierarchical Cumulative Voting (HCV) [41]) is an extension of cumulative voting.

Overall, we recommend that:

- Use approaches without trade-off when many goals should be prioritized on one level, but the drawback being that the “everything is important” problem has been observed in the context of prioritizing software requirements [39] since no trade-offs have to be made as prioritization items are not in competition.
- Use approaches with trade-off (pair wise comparison through AHP or HCV showing the distance between items, and at the same time enforcing trade-off decisions between prioritization items) when there are very few goals on each level of the hierarchy.

For this study, as part of GQM-DSFMS Tool that we developed to serve as a proof of concept when conducting the case study (see Section 4), we implemented an importance rating approach without trade-off between the goals so that we would not be constrained if many goals exist at the same level when conducting further case studies. Particularly, a ratio scale of 1 to 20 is used.

3.2. Phase 2: Questions Elicitation

During this phase, the questions that need to be answered for achieving the goals of different stakeholders are elicited and recorded. This requires a repeatable and defined process. Otherwise different analysts can come up with different questions. GQIM [3] includes also indicators in addition to questions to further constrain the free will of measurement analysts and lets stakeholders be more involved in the process. However, this version of GQM-DSFMS does not include indicators in the data model. We plan to incorporate this in the next version.

A many to many relationship exists between the goals and the questions (see Figure 2). Further, questions can be broken down into sub-questions. For example, the goal we want to reduce the number of customer reported defects by x % could lead to a question such as “How does the inflow of defects change over time?” This question then can lead to sub-questions, e.g. “How many defects are reported per week?” and “What is the distribution of defects with respect to fault criticality in weekly time intervals?” In this research we used examples in [6] to guide us how to formulate the questions. Further examples of how to formulate the questions can be found in recent books on software measurement [42, 43]

In GQM-DSFMS, a structured template (e.g. as defined in [9]) is used to capture the relevant information about the questions. The traceability information required by the Traceability Model (see Figure 2) is recorded by using the defined template (e.g., a unique identifier, version number, relevant business goal(s), measurement goal(s) and question(s)) (See the running example for Phase 2 in Appendix A.1).

3.3. Phase 3: Mapping Questions to Standard Attributes/Metrics Repository

GQM-DSFMS incorporates a pre-defined “Attributes/Metrics Repository”, which is a repository containing a set of attributes and associated metrics defined for different software entities (see Figure 1).

The reason for such a mapping is to address the need of the organizations to make use of metrics defined and classified according to the attributes of software entities in well-known standards. Organizations can tailor the attributes and metrics included in this repository based on their needs. For example, before conducting the case studies, we populated this repository with the attributes and metrics defined in ISO 25021 [29] and ISO9126 standard family [30, 31, 32]. There are also few studies which provide lists of core measures for capturing important information on projects [44] or required to be collected according to the needs of different CMMI levels [45].

This phase includes the following two steps (see the running example for Phase 3 in Appendix A.1).

3.3.1. Step 1: Map Questions to Corresponding Attributes

The questions are mapped to their relevant attributes in the Attributes/Metrics Repository. Furthermore, the traceability between the identified goals, questions and the attributes is established using the Traceability Model (see Figure 2).

3.3.2. Step 2: Investigate Candidate Metrics in Attributes/Metrics Repository

In the Attributes/Metrics Repository, each attribute is associated to a number of alternative candidate metrics that can be used to measure that particular attribute. For example, if there is a need to collect data about product size attribute, then all metrics defined to measure product size (e.g. Function Points, Lines of Code, etc.) are identified as “candidate” metrics to be input to decision making phase discussed below. When an attribute is measured through derived metrics (e.g., effort in person months and size in FP are direct metrics, which together are used to measure productivity), the alternative metrics for each are also identified.

3.4. Phase 4: Decision Making

Using the GQM-DSFMS, the measurement analysts make decision for the metrics among all alternative candidate metrics based on how much information the metrics set provides within the

given budget considering the cost for each metric and the importance value a metric inherits from the goals. This phase consists of the following six steps:

3.4.1. Step 1: Inherit the Importance Values of Goals to Metrics

The importance values for metrics are obtained by inheriting the importance values of the goals which were determined in Phase 2. In this study, we defined the following mechanism and implemented as part of the GQM-DSFMS Tool to serve as a proof of concept when conducting the case study (see Section 4). A running example is given in Appendix A.2

Here, we should state that this approach requires to have importance values in ratio scale. Otherwise, an organization can define another mechanism and use a different scale for the importance inheritance step.

Each goal, question, attribute, and metric is represented by a node with an importance value; α . The final importance value of each node is represented as $\alpha_{l,m}^f$ where l is the level of the node (e.g. strategic goal, operational goal, question, sub-question, etc.), m is the number of a particular node on level l , and f indicates the final importance value. Thereafter, on each level l , the importance values are normalized by the common divisor of the nodes at level l . Furthermore; the importance values from the nodes on level l are determined based on the importance values of level $l - 1$. In case of the root nodes (top-level goals), the weight is 1. The importance values are calculated as follows:

Assume that a node at level $l - 1$ has P children nodes, where the importance of this node is $\alpha_{l-1,m}^f$.

Calculate the common divisor (CD) and repeat for the children of each node at level $l - 1$ separately. :

$$CD = \sum_{p=1}^P \alpha_{l,p}. \quad (1)$$

Normalize the importance values of the nodes by the common divisor:

$$\alpha_{l,m}^n = \frac{\alpha_{l,m}}{CD} \quad (2)$$

Inherit the importance value of the node on level $l - 1$ to its children nodes on level l and repeat for the children of each node at level $l - 1$ separately.:

$$(\alpha_{l,1}^f, \alpha_{l,2}^f, \dots) = \alpha_{l-1,1}^f (\alpha_{l,1}^n, \alpha_{l,2}^n, \dots) \quad (3)$$

If one node on level l is connected to multiple nodes on level $l - 1$ the importance obtained due to all these nodes is added. An example of how to calculate the inheritance is shown in Appendix A.

3.4.2. Step 2: Calculate Cost of each Candidate Metric

GQM-DSFMS considers a number of cost factors for calculating the collection cost for each candidate metric. We identified these factors through industrial surveys in our previous study [38]. The measurement analysts should make estimations for each factor to be input to calculations. How GQM-DSFMS calculates the cost for each metric is described below and a running example is shown in Appendix A.3.

First, the total collection effort (TCE) is calculated by:

$$TCE = T * F * R \quad (4)$$

where, Time (T) is the time needed to collect a metric once, the Frequency (F) refers to how many times that metric is collected per a period of time (e.g. on a weekly/monthly/yearly) and the Number of Resources (R) is the number of human resources utilized during collection.

Then, the Total Cost for that metric is calculated by:

$$TC = TCE * RC + E \quad (5)$$

where Resource Cost (RC) is the cost per unit time per resource that is involved in the collection process and Expense (E) involves any other additional expenses such as other non-human resources cost, training, tools, hardware, etc. A calculation example is provided in Appendix A.

In case the costs are estimated through expert opinion (as is the case in this study) then source triangulation is recommended to increase the reliability of the results. The same also applies to goal prioritization.

3.4.3. Step 3: Find an Optimum Set of Metrics

This is an optimization problem in which the objective is to maximize the total accumulated importance of the set of chosen metrics (see Phase 2) and the constraint is to keep the cost within the total allocated budget for collecting these metrics. Here, to keep it simple, we modeled the problem with a single objective function. However, this could also be treated as a multi-objective problem where a trade-off between total accumulated importance and total cost is made. Below, we explain how it works and provide a running example in Appendix A.4

Assume we have K candidate metrics, B is the budget, the vectors $C = \{c_1, c_2, \dots, c_K\}$ and $I = \{i_1, i_2, \dots, i_K\}$ represent, respectively, the cost and the importance associated to each of these metrics, and $M = \{m_1, m_2, \dots, m_K\}$ represents the metrics to choose from.

The inclusion of a metric into the set of metrics to be collected is indicated in M as:

$$m_k = \begin{cases} 0, & m_k \text{ will not be collected} \\ 1, & m_k \text{ will be collected} \end{cases} \quad (6)$$

The cumulative importance $I(M)$ for any M is given as:

$$I(M) = \sum_{k=1}^K i_k m_k. \quad (7)$$

Similarly, the total cost $C(M)$ associated with any M can be computed as:

$$C(M) = \sum_{k=1}^K c_k m_k. \quad (8)$$

Finally, the optimization problem can be considered as finding the M that maximize the cumulative importance, given that the total cost for that M is less than or equal to the total cost budget allocated for collecting these metrics. Alternatively, it can be stated as:

$$P = \begin{cases} \max_M I(M) \\ \text{subject to } C(M) \leq B \end{cases} \quad (9)$$

Any optimization approach that can solve the above problem can also be used as part of GQM-DSFMS. Several techniques can be used (see [46] for alternatives) for finding the optimal set of metrics for the given total cost. Example optimization approaches providing “optimal” solution are as follows:

- *Exhaustive (Brute Force) Search* is the simplest, but most time consuming way to find the optimal solution as the number of the candidate solutions typically grows exponentially with the number of metrics K .
- *Dynamic Programming* is used to solve complex problems that can be split into smaller sub-problems (divide and conquer). For each sub-problem the optimum is determined and later combined, hence being faster than exhaustive search.
- *Branch and Bound* consists of a systematic enumeration of all candidate solutions, where large subsets of fruitless and non-feasible candidates are discarded, by using estimated upper and lower bounds of the objective function and cost.

Brute force and dynamic programming algorithms with variables defined for this approach are provided in Appendix A.4.

Example optimization approaches providing good solution but not guaranteeing the optimum include heuristics. These techniques seek good enough solutions at a reasonable cost. These techniques are experience-based and depend on characteristics of the optimization problem at hand. One drawback is that sometimes they are trapped into local optima and are incapable of escaping from it. For this reason, meta heuristics such as Tabu search or simulated annealing algorithms have been developed.

The choice of the optimization approach depends on the number of candidate metrics and the maximum time allowed for performing optimization. Considering the processing powers of today’s ordinary computers and the nature of our problem, we suggest the following rule-of-thumb: use exhaustive search for small problem instances (few candidate metrics set) due to its simplicity, otherwise use a more advanced exact approach or a heuristic approach.

In this study, we formulated the optimization problem using Brute Force (see Algorithm 1) and Dynamic Programming (see Algorithm 2). Brute Force algorithm was implemented as part of the GQM-DSFMS Tool we developed to serve as a proof of concept when conducting the case studies (see Section 4).

3.4.4. Step 4: Check the Extent of Achievement in Fulfilling the Information Needs

After finding the optimum set of metrics, one more step is necessary to assess whether the solution is satisfactory. GQM-DSFMS framework provides a mechanism for the measurement analysts to check the level so that this won’t create major problems about the *completeness* (when it measures everything that its users need to achieve their goals) of the framework. If, for example, the budget is kept too low, the selected metrics will not be sufficient to give the information necessary for achieving the goals. In this case, either the company might decide to increase the budget or else return to the goals importance determination phase and re-think about the importance of each goal. In GQM-DSFMS, % achievement in fulfilling the information needs at m^{th} node on level l having p children nodes; $\beta_{l,m}$ is calculated as:

$$\beta_{l,m} = \{(\beta_{l+1,1} * \alpha_{l+1,1}^n) + (\beta_{l+1,2} * \alpha_{l+1,2}^n) + \dots + (\beta_{l+1,p} * \alpha_{l+1,p}^n)\} * 100 \quad (10)$$

where $\alpha_{l+1,p}^n$ is the normalized importance value of the m^{th} child of a node of level l . If any node does not have a child node, $\beta_{l+1,p} = 1$ by default.

A running example of calculating the achievement for a selected set of metrics is given in Appendix A.5.

3.4.5. Step 5: Decide on the Metrics

After obtaining the extent of achievement to fulfill the information needs considering the constraints of the organization, the measurement analysts either accept the solution and continue with the next step, or else return back to Phase 1 where the priorities of the goals or the constraints are set. This might require a number of iterations until a satisfactory solution is obtained.

3.4.6. Step 6: Store Links in the GQM Tree and Cost factors

Once the optimum set of metrics is accepted, all links among the GQM elements as well as the cost factors with their timestamps are stored in the database.

4. Case Study Design

The empirical research method used for evaluating the GQM-DSFMS framework is case study. In order to assure rigor in case study design, we followed the guidelines provided in [47]. We investigated two embedded cases within the case company (ABC).

4.1. Goal of the Case Study

The goal of this study is formulated as follows:

- Assess *GQM-DSFMS* for the purpose of *evaluation by comparing to traditional expert opinion approach*
- with respect to *Total Cost (TC) for measurement collection* and *Extent of Achievement (EA) in fulfilling the information needs for measurement goals*
- from the *project manager, business analyst and team leader perspectives*
- in the context of *members of two software projects providing input to GQM-DSFMS in the forms of goals, their priorities, and cost for metric collection.*

4.2. Case Description

The case is ABC, an IT-based multinational company established in 1987 in US and currently having branches and satellite offices in Canada, Kuwait, Dubai and Pakistan. It has around 700 employees. ABC provides IT solutions such as data warehousing, customized software development, package implementation and modification as well as services for telecommunication, human resource management and financial management.

ABC has been certified as CMMI level 3 and targets CMMI level 5 in the near future. Currently, the decision to collect metrics in ABC has been performed in a bottom-up way as a need arises. There is no standard or formal process followed in the organization to run the measurement program. So far, the focus was primarily on time, cost, and resources needed. Therefore, the company was very interested in testing the GQM-DSFMS framework in their context conducting a case study.

For the selection of the embedded cases within the company, we presented the framework and made an informal interview with the project manager. The project manager has been working for ABC for the last 6 years and prior to this he worked as a software architect, business analyst and senior software engineer. During the interview, it was decided to test the framework at the project level and two cases were determined: Project 1 (P1) and Project 2 (P2), representing different contexts in terms of measurement requirements (i.e., new development and maintenance).

P1 was a new development project for an e-banking application in Kuwait Bank. A business analyst, a requirements engineer, a team lead, a software developer and a quality assurance engineer worked for this project and the duration for the development was 6 months. The team used J2EE and SQL databases to develop the software.

P2 was a maintenance project for an e-management system. It was developed for the company's own use and would be launched in the market after maintenance. One team lead, one business analyst, two quality assurance engineers, one requirements engineer, one technical writer and six developers worked for this project and the duration was 8 months, and was also developed using J2EE.

4.3. Data Collection and Analysis

After deciding on the cases together with the project manager, we scheduled two workshops to be conducted through video conferencing. We invited the project manager, one business analyst and one technical team leader as the participants of the case study, who were involved in both P1 and P2 so that we could capture three different perspectives; the overall project management perspective, the business benefits perspective and the lower level management (e.g. technical risk) perspective.

We used two operational measures when conducting the case study: Total Cost (TC) of metrics (see Section 3.4.2 for details) and Extent of Achievement (EA) in fulfilling the information needs (see Section 3.4.4 for how to calculate).

During the workshops, we followed the phases of the GQM-DSFMS process by instantiating it considering the needs of the case organization (please see Section 5.1 for details). Below we provide an overview of how we conducted the workshops:

- *Workshop 1:* The first workshop lasted in two hours to elicit the goals and sub-goals for the projects making informal interviews with the participants. Google Docs was used so that the participants could see the information documented. Two of the authors acted as moderators while eliciting the goals.
- *Workshop 2:* The second workshop lasted in three hours to elicit the questions and to identify metrics based on the goals and sub-goals of the projects identified during the first workshop. We asked estimated values for the factors determining the cost of metrics as well as the budget available for metrics collection. We also asked which metrics they would choose for their projects using their current conventional approach so that we could evaluate the decision making framework for metrics selection in comparison to expert opinion. This allowed to afterwards compare with the results of applying the structured GQM-DSFMS approach for the same cases.

In order to be able to use DSFMS-GQM in ABC, we developed an automated tool to support capturing the information needed for optimal metrics selection. In particular, the tool provided support for implementing the structured templates to collect and store information about

goals and questions, generating and storing traceability links among GQM elements using the Traceability Model, developing an Attributes/Metrics Repository and implementing some of the techniques (such as for importance rating of goals, importance values inheritance to metrics, cost calculation for metrics, optimization, and information needs achievement calculation) suggested in the framework as a proof of concept. The details are provided in the next section.

After holding the workshops, we provided the tool to the practitioners and supported them in installing it on their servers as well as to enter the results obtained in the workshop into the tool, the intention being to transfer the knowledge of how to use the tool and enable the organization to continuously update their measurement program for future projects.

Then, we calculated the TC (see Section 3.4.2 for details) and the EA in fulfilling the information needs (see Section 3.4.4 for how to calculate) when using GQM-DSFMS. The experts participating in this study estimated the cost of metrics composed of the time to collect, number of persons needed, and frequency of collection. The expert opinion approach was used as it was the common way of decision making at the company. The practitioners collected measures before in the organizations, and hence have experience in measurement collection. Prior to estimating the cost the practitioners received the metric repository beforehand for review. This helped to assure that they have a good understanding of the metrics to estimate the cost. In the meeting the team lead and manager discussed and agreed on an estimate. There were no issues with disagreement on the cost figures.

Finally, we compared the results to evaluate the GQM-DSFMS in terms of cost effectiveness and efficiency.

Next section presents the details of case study conduct and the results.

5. Case Study Conduct and Results

This section is divided into two parts. First, we describe and motivate the framework instantiation of GQM-DSFMS for the case study conduct. Second, we compare the outcomes of GQM-DSFMS to the outcomes of expert opinion.

5.1. Framework Instantiation

GQM-DSFMS is instantiated by choosing one of three prioritization approaches for the goals, and one of three optimization approaches in decision making.

For this study, we chose one baseline instantiation to compare the process of GQM-DSFMS, the baseline being based on one of the simplest prioritization approach (rating on a scale). This was chosen for the reason that this approach does not require any training for the participants, and hence can be easily applied without prior knowledge about prioritization techniques.

As for the optimization algorithm, we chose exhaustive search, as this allowed us to find the optimum, given that the problem could be solved in reasonable run-time. In the following we explain the GQM-DSFMS process followed in the case company in further detail.

5.1.1. Phase 1: Goals Elicitation

In this phase, one common business goal for both P1 and P2 was determined. Then, for each case project, one measurement goal is identified and associated with its subgoals (see Figure 3 for the GQM-DSFMS tree elicited from the two projects). The data on goals were collected interviewing the participants and then entering the information using the GQM-DSFMS Tool¹.

¹The tool and other materials can be found on <http://www.bth.se/tek/aps/kps.nsf/pages/gqm-dsfms>

Figure 3: GQM-DSFMS Tree for Project P1 (White Boxes) and Project P2 (Grey Boxes)

The white nodes represent the information elicited for P1, and the gray nodes for P2. It is visible that P2 also requires metrics from P1. Only for two relatively small projects the graph indicates the extent of complexity of the decision problem. Finally, the importance of each goal was rated using a numerical scale of 1 to 20, where no trade-offs between goals have been made (see Section 3.1.2). The importance of goals determines the importance of metrics. The project manager rated the importance of the goals from management perspective; the technical team leader from the organization/technical perspective and the business analyst on a general level. For each goal, the average of the importance values were used as the input for the importance inheritance step (see Section 3.4.1).

5.1.2. Phase 2: Questions Elicitation

During this phase, the questions that need to be answered for achieving the measurement goals are elicited and recorded using the structured templates. The traceability information are also stored. For P1: 8 questions and for P2: 9 questions were identified (see Figure 3).

5.1.3. Phase 3: Mapping Goals/Questions to Standard Attributes/Metrics Repository

The identified questions were mapped to relevant attributes in the Attributes/Metrics Repository. Initially, we populated the Attributes/Metrics Repository with the attributes and metrics for product quality in ISO/IEC 25021 standard [29].

As the identified and selected goals for these case projects were related to only product quality (specifically associated with product reliability, availability, portability, maintainability and usability), we did not need to add other attributes and metrics.

The GQM-DSFMS Tool automatically identified all possible candidate metrics from the Attributes/Metrics Repository and showed them to the participants. The participants provided estimations for the factors to calculate the cost for each candidate metric (see Section 3.4.2) and these were stored using the GQM-DSFMS Tool. That is, the metrics were given through expert estimates. We also asked the manager to decide on the maximum budget that can be allocated for metrics collection. He set the budget limit for P1 as \$2000 and for P2 as \$7000.

Table 1 provides the cost of collecting all metrics associated with the goals. Collecting all metrics implies completeness as defined in [12], and would imply an achievement of 100% in fulfilling the information needs. However, for GP1 the total cost is 6,225\$ (4,225\$ over budget) and for GP2 the total cost is 19,985\$ (12,985\$ over budget). Hence, in the case study there is a clear need for goals and metrics prioritization.

Table 1: Collection of all Metrics Identified for the Elicited Goals and Information Needs

Goal No.	Cost (all metrics)
MGP1.1 (reduce likelihood of failure)	2,330\$
MGP1.2 (decrease work to portability)	1,595\$
MGP1.3 (decrease failure of functionality)	2,300\$
Total cost GP1 (improve reliability)	6,225\$
MGP2.1 (decrease errors that cause failure in operation)	8,760\$
MGP2.2 (improve project maintenance)	7,025\$
MGP2.3 (improve requirements maintainability)	4,200\$
Total cost GP2 (improve maintainability)	19,985\$
Total cost GP1 and GP2	26,210\$

Before proceeding with the decision-making phase, we asked the participants to decide on the metrics that they would choose from the identified candidate metrics using their current approach to metrics selection. The reason was to be able to compare the optimum set of metrics obtained by carrying out the decision making process of GQM-DSFMS to the set selected set by the measurement stakeholders based on their expertise in the traditional way. For P1: 12 and for P2: 15 metrics were selected when the traditional expert opinion approach was used.

5.1.4. Phase 4: Decision Making

For the decision making process, the Brute Force algorithm was used due to its simplicity in implementation (see Section 3.4.3). The GQM-DSFMS tool provided the optimum set of metrics based on the budget limits set by the project manager and the importance values for the metrics.

5.2. Comparing GQM-DSFMS with Expert Opinion

In this section we compare GQM-DSFMS with the expert opinion approach based on the Total Cost (TC) for metrics and the Extent of Achievement (EA) in fulfilling the information needs associated with the goals for each case.

Table 2 shows the results for EA (%) in fulfilling the information needs by the optimum set of metrics in comparison to that of expert opinion approach. As described in Step 4, the extent of achievement allows to determine to what degree the information needs arisen from measurement goals are fulfilled with respect to selected metrics. This also takes into account the priority of the goals (e.g. if a goal has a priority of zero, then the goal does not have to be fulfilled, and should not be considered). In the case of expert opinion the practitioners made an informed choice of which metrics they would collect for their projects. In GQM-DSFMS the practitioners prioritized the goals and this was a driver for metrics selection. In the case of expert opinion a marginally higher information need fulfilment is achieved for GP1 (improve reliability). For GP2 (improve maintainability) GQM-DSFMS achieved a significantly higher achievement.

Table 2: Extent of Achievement (EA) in Fulfilling the Information Needs (%) (GQM-DSFMS versus Expert Opinion)

Goal No.	GQM-DSFMS-EA (%)	Expert Opinion-EA (%)
MGP1.1 (reduce likelihood of failure)	92.64	–
MGP1.2 (decrease work to portability)	3.16	–
MGP1.3 (decrease failure of functionality)	76.41	–
GP1 (improve reliability)	73.35	80.00
MGP2.1 (decrease errors that cause failure in operation)	87.50	–
MGP2.2 (improve project maintenance)	78.47	–
MGP2.3 (improve requirements maintainability)	0.00	–
GP2 (improve maintainability)	82.95	55.50

Figure 4 shows the candidate metrics for each sub-goal of P1 and P2. We used Venn diagram notation to illustrate the common metrics which would be required to provide information for different goals. For example; M5, M6, M7 and M8 are common candidate metrics for all three sub-goals of P1. Which metrics have actually been selected is visible from Figure 3.

Figure 4: The Selected Metrics for P1 and P2 using GQM-DSFMS

For P1, the experts selected 12 metrics out of 18 candidate metrics using the traditional approach whereas GQM-DSFMS selected 9 metrics (shown in red color in Figure 4) when the budget is limited to 2000 \$. GQM-DSFMS decided on the metrics, which in total cost 1955\$ and achieve about 73.4% in fulfilling the information needs. The cost of selected metrics was 2780\$ with expert opinion with about 80% achievement.

For P2, both the experts and the GQM-DSFMS selected 15 metrics out of 24 candidate metrics. However, even though the number of metrics were the same, the selected metrics were not the same. GQM-DSFMS decided on the metrics (shown in red color in Figure 4), which cost 6885\$ in total with about 83% achievement in fulfilling the information needs under a budget limited to 7000\$. On the other hand, the experts selected metrics which in total cost 9251\$ with around 56% achievement.

The case study results show that GQM-DSFMS is an improvement over the traditional expert opinion approach with respect to achievement of fulfilling the information needs and cost. At the same time unimportant metrics have not been selected, that would have been selected when expert opinion approach was used. Therefore, we conclude that GQM-DSFMS framework has a potential to aid in metric selection in terms of cost effectiveness and efficiency (i.e. extent of achieving to fulfill the maximum number of information needs required by goals while keeping the cost within available budget).

6. Threats to Validity

According to [47] four types of validity should be considered when conducting research, namely construct validity, internal validity, external validity, and reliability.

Construct Validity: This validity is concerned with whether the constructs studied really represent what the researcher intended to study [47]. In particular, it was important to make sure that the participants understand the concepts of goals, questions, and metrics. This was achieved through templates (and associated definitions). Furthermore, as the authors documented the elicitation process with the interviewers and interviewees seeing what is documented and understood through Google Docs, this threat is considered to be under control.

We used two operational measures in the case study: TC and EA in fulfilling the information needs. For the cost calculation, we used the approach and the cost factors as defined in [38]. Any significant factor not captured in that study might have had impact on the case study results. For calculating the extent of achievement in fulfilling the information needs, we defined a simple metric based on the inherited importance values of the metrics. As the participants from the company agreed on the use of this metric, we believe that we could capture the information we require to a reasonable extent.

Metrics that measure exactly the same thing (e.g. different types of function point metrics) are not differentiated by the decision support algorithm. That is, if only one of the redundant metrics is chosen, the EA might be reduced, even though the metrics are interchangeable. However, in the case study no interchangeable measure was included. Hence, this is no threat in the comparison of expert opinion and GQM-DSFMS. In order to avoid this issue when implementing GQM-DSFMS in different contexts there is a need for classifications allowing to identify interchangeable metrics.

The cost of metric collection was estimated. However, the estimation does not constitute a threat to the evaluation goal of this study. If there would not be a cost constraint and costs, the selected metrics would always fulfill 100% of the information needs, as those would be collected. The goal of this study was to investigate how, under a constraint (cost), the metrics selected by practitioners would be different with respect to fulfillment of information needs. We found that the approach helps them to achieve higher information need under that constraint. That is, even with a potential estimation error, this could be demonstrated and was the goal of the research.

Internal Validity: Internal validity is important to consider when causal relationships are analyzed, or inferences are made [47]. Some of the techniques defined as part of GQM-DSFMS such as cost calculation for metrics, importance rating of goals, importance values inheritance to metrics, and formal problem formulation in decision making might raise some validity threats. For example, as for the optimization problem formulation, we simplified the solution by using single objective function and only considered cost as the constraint, while other factors might be of relevance as well (e.g. availability with respect to time).

Another influence on the result might be bias (e.g. by only considering one source of data). In order to reduce this threat, we studied two different projects. Furthermore, different perspectives/roles were included in the analysis to avoid bias.

Given that the company achieved CMMI level 3, they do not have an experience base of cost for metric collection. Therefore, there is a risk of reduced estimation accuracy, for example duration of metric collection.

External Validity: External validity is the ability to generalize the findings of the studied context to other (different or similar) contexts [47]. We evaluated the framework only in one case organization at project level for only two case projects. Moreover, only two main goals were selected and the measurement stakeholders were the same for both projects. Therefore, we were not able to show how GQM-DSFMS would behave when different levels in the organization were involved (here only the project management level was considered, another level of relevance could be e.g. organization or business). In fact, we expect to see more benefits of this framework if applied at organizational level where the complexity is much higher. However, due to limited resources we were not able to make this evaluation. Therefore, the results are likely to be generalizable to project level measurement planning. External validity can be further strengthened by, in future studies, applying GQM-DSFMS in a large-scale organization that has many stakeholders on different levels, namely business level, software level, and project level [9].

To further strengthen external validity there is a need to evaluate the approach in companies of different scales and contexts as well. Furthermore, we could not show exactly under which condition which instantiation of the framework would provide more benefits, as this would require conducting more case studies in different contexts.

Given that the organization is currently CMMI level 3, but aims at higher levels, it has no defined measurement process yet. In this case, our approach helps them in defining the measures that should be collected when they mature up to CMMI level 4, where the process is measured and controlled. That is, a CMMI level 3 company would be in a situation to select the initial set of metrics, while a higher level company would use the approach to consistently evolve their measurement programs, and also continuously improve it and evaluate the effectiveness of the current one. From an outcome point of view a company with a lower CMMI level, and where people are not experienced in measurement program planning, the first version of the tree would be incomplete. While learning, we expect that the tree will become more complete over time with added maturity (see Section 7, *Evolvability*).

Reliability: This threat is concerned with the degree to which the outcome of the study is dependent on the researcher [47]. This threat was addressed in two ways in this study. First, the study was conducted by a set of researchers allowing to discuss interpretations and hence avoiding bias of a single person. Secondly, as a form of member checking the elicited information in workshops always made visible to the practitioners, making sure that the interpretation of what is being said during the workshops was not biased by our interpretation.

7. Discussion

In this section we discuss the practical and research implications of the solution proposed in relation to the results of the case study.

Achieving cost effectiveness in metrics collection: In the case study we evaluated the capability of GQM-DSFMS to provide decision support so that a cost-effective solution with respect to selected metrics is achieved. In the case we saw that when collecting all metrics in relation to the identified goals, the measurement program would be over budget by approximately 200%. We found that in one project (P1) the EA was slightly lower for GQM-DSFMS in comparison to expert opinion. However, expert opinion was over budget by approximately 40%, hence in this case GQM-DSFMS was more cost-effective. In the second case (P2) expert opinion led to the cost being over budget, and at the same time a low value for EA was achieved. GQM-DSFMS provided a solution with higher information need achievement and within budget. Overall, this indicates that as a decision support framework GQM-DSFMS is useful in finding a cost-effective solution for metrics selection. The necessity for the selection became obvious considering the high cost for collecting all metrics to achieve 100% achievement in addressing information needs for goals.

Complexity and scalability: The case projects studied were relatively small and were focused on quality aspects that were specifically important to the case company. That is, other process related goals were not in focus, and would have made the decision problem even more complex. Already in this situation the goal tree was growing quite large. Furthermore, collecting all metrics would have been quite expensive. We have demonstrated that the EA in fulfilling the information needs for important measurement goals was high through the optimization approach while keeping the cost of metrics within budget. In future studies, there is a need to study GQM-DSFMS on large scale organizations: (1) involving multiple levels of the organization will lead to having multiple hierarchies of goals, and also conflicting opinions and dependencies between

levels, such as business level, software level, and project level [9]; (2) the number of metrics will explode as every level is likely to propose a high number of metrics for their particular level and interest, making support for metric selection essential in relation to the investment one is prepared to make; (3) different priorities for different metrics will be identified by the multiple stakeholders, hence requiring mechanisms to reach a consensus among measurement stakeholders. Furthermore, different units of the organization as well as different stakeholders might have different decision criteria, which are not incorporated in the model yet. These also have to be negotiated and agreed upon in the measurement program planning. The decision criteria also depend on the current capabilities of the organization.

Reusability: One aim of GQM-DSFMS is to enable re-usability of metrics by making their links to information needs and goals on multiple levels of the organization explicit. By making the links explicit, we achieve transparency of whether there is a common interest in metrics and information needs between levels. Metrics on one level (project) could together provide new indicators that are important for decision-making in levels higher up in the hierarchy (e.g. business level). It is in this context also important to highlight that effectiveness is emphasized as the degree to which the selected metrics satisfy the information needs. Hence, if information needs are shared between different groups of stakeholders, GQM-DSFMS will make the degree of achievement for these information needs explicit. Given that the approach has been used on project level in this case study, there is a need to evaluate GQM-DSFMS in a case company with many stakeholders on the different levels (business, software, and project [9]).

Evolvability: Staron et al. [3, 9] defined procedures to enable reuse of metrics and automate collecting and combining metrics. Structured templates have been introduced to help in making the attributes of goals and questions explicit. However, these templates are difficult to maintain over time due to lack of formal traceability models. This also makes it very difficult for the organizations to use automated tools to save time, cost and enhance re-usability. Especially in a dynamic environment where a measurement program evolves over time [11]. We designed the framework so that the traceability allows for updates in the goal-tree (e.g. addition or deletion). This also leads to automatic updates in the priorities for metric selection, and hence technically supports to ease the update and re-prioritization of metrics, and recalculation of information needs achievement. For example, when removing a goal (due to that it became irrelevant), traceability links and priority values are updated consistently as traceability between all elements of the model is clearly defined and recorded. Evolvability becomes even more important in the GQM+Strategies where goals at different levels are considered, that change at a different pace and still need to be consistent with each other (e.g. a software goal is deleted and hence a project goal becomes irrelevant, thus metrics become irrelevant or less important). Evolvability is also important from an agile organization point of view, as goals and metrics are likely to change over time. Three types of evolvability are important to distinguish, namely bottom-up, top-down and bottom-up/top-down:

- Bottom-up: Based on the metrics an organization is currently collecting, it can determine which information needs can be fulfilled with the current set. The organization can also determine which new information needs can be fulfilled by adding a particular metric.
- Top-down: Based on the goals and information needs an organization wishes to fulfill, it can identify all metrics that could be collected achieve and sustain completeness in the measurement framework.
- Bottom-up/Top-down: Due to that the framework enables both types of traceability, or-

ganizations can start at any level to evaluate scenarios of how to plan their measurement programs with respect to metric selection.

Overall, GQM-DSFMS has been designed for evolvability, the evaluation of its capability in handling evolvability being an important part of future research work.

Tool support: When the complexity grows there is a need for tool support, which was also highlighted by the practitioners. The tool keeps track of the traceability links and allows for a consistent evolution of measurement program relevant information. Furthermore, the tool relies on templates, which aids practitioners in entering complete information so that an automated identification of metrics becomes possible after goals, information needs, and attributes have been entered.

Attributes/Metrics repository: The Attributes/metrics repository used in this study was focused on software quality, and could be easily populated due to availability of quality standards. When applying GQM-DSFMS at different levels (see GQM⁺ Strategies) the repository has to be populated with a wide range of metrics (e.g. from standards). In future work, building a pre-defined Attributes/Metrics repository is an important task for applying GQM-DSFMS in a variety of contexts. As a starting point for building such a repository we recommend [48].

Using the results of GQM-DSFMS: GQM-DSFMS aids practitioners in selecting metrics so that their information needs are achieved in a cost-effective way. Relating goals to information needs and seeing to what degree the selected metrics satisfy the information needs is a first and necessary step in measuring, which provides valuable insights. However, beyond that the success of a metric depends on other factors. For example, metrics could be used for learning on a project level, but here is a risk that they become a means of comparing projects and being used for finger pointing by upper management. Hence, the degree of success of GQM-DSFMS is of course influenced by whether it is embedded in a fair and communicative process for measurement program planning. We also highlight that GQM-DSFMS is providing decision support, and shall not make direct decisions for organizations of what metrics to collect, but support them.

8. Conclusion

In this paper, we proposed a framework to support software organizations making informed decisions in metrics selection when planning measurement programs. The so-called GQM-DSFMS framework was evaluated through a case study with two embedded cases in a CMMI Level 3 software company.

The case study results showed that, by using GQM-DSFMS, not only the cost for metrics, but also the value of collecting those metrics (in terms of fulfilling the information needs) has become transparent for the decision makers. GQM-DSFMS provided an improvement over the conventional expert opinion approach (i.e. without support of a structured method) in terms of extent of achieving to fulfill the maximum number of information needs required by goals while keeping the cost within available budget. At the same time, unimportant metrics have not been selected, that would have been selected in case of applying expert opinion approach.

The participants of the case study mentioned that a prerequisite for using such a framework in a software organization is an automated tool support so that they do not require to know the details about each technique used as part of it. Hence, tool support was identified as an enabler for successfully deploying the framework in practice. Furthermore, the project managers of the cases reported difficulties in rating the importance values of goals, by stating that a clear definition of what importance means and a definition of the scale should be made clear.

A significant advantage of GQM-DSFMS is that, being a framework, it can be tailored to different organizational contexts. Here, we provided a number of feasible example alternative techniques and methods that can be used as part of GQM-DSFMS such as for calculating cost of metrics based on some factors, importance rating of goals (such as HCV, independent importance scale etc.), inheritance of importance values down to metrics and optimization techniques for decision making.

However, the framework is not restricted to their usage; i.e., any new approach or technique can be plugged into the framework to replace another according to the needs of the organization. For example, the choice of whether to use a simple heuristics based or sophisticated optimization method depends on the estimated number of metrics, and the choice of importance rating method on the number of stakeholders and their goals.

As a future study, we plan to incorporate a number of ideas in literature that we believe are important to integrate into GQM-DSFMS framework. One example is to add the 'Indicator' concept as defined in GQIM [3]. We also would like to elaborate on use of some ideas presented in GQM+ Strategies [9], which enables monitoring of the deployment strategy for each goal and getting feedback on the effectiveness of the chosen strategy. Last but not the least, we plan to extend the schema of Attributes/Metrics Repository considering the conceptual model developed in [19].

From an empirical point of view, given that we studied GQM-DSFMS on project level, future studies conducted in large-scale organizations adopting GQM-DSFMS should focus on: (1) reuse of metrics by making links between metrics, information needs, and goals across multiple levels (business, software, and project); (2) evaluating GQM-DSFMS with respect to total cost for measurement collection and extent of achievement in fulfilling the information needs for measurement goals when multiple levels are involved and a very high number of potential metrics is of relevance; and (3) evaluating how well GQM-DSFMS supports the longitudinal evolvability of a measurement program.

9. Acknowledgement

Thanks to all participants of the study. We would also like to thank the reviewers for their valuable comments on the paper, which helped to improve it. This work has been supported by Ericsson AB and ELLIIT, the Strategic Area for ICT research, funded by the Swedish Government.

References

- [1] J. J. P. Schalken, H. van Vliet, Measuring where it matters: Determining starting points for metrics collection, *Journal of Systems and Software* 81 (5) (2008) 603–615.
- [2] R. van Solingen, E. Berghout, Integrating goal-oriented measurement in industrial software engineering: Industrial experiences with and additions to the goal/question/metric method (GQM), in: *Proceedings of the 7th International Software Metrics Symposium (IEEE Metrics 2001)*, 2001, pp. 246–258.
- [3] A. J. Boyd, The goals, questions, indicators, measures (gqim) approach to the measurement of customer satisfaction with e-commerce web sites, *Aslib Proceedings: New Information Perspectives* 54 (3) (2002) 21–24.
- [4] R. S. Kaplan, D. P. Norton, *The balanced scorecard: translating strategy into action*, Harvard Business School Press, Boston, Mass., 1996.
- [5] V. R. Basili, D. M. Weiss, A methodology for collecting valid software engineering data, *IEEE Trans. Software Eng.* 10 (6) (1984) 728–738.
- [6] V. Basili, C. Caldiera, H. D. Rombach, Goal question metric paradigm, encyclopedia of software engineering, in: J. J. Marciniak (Ed.), *Encyclopedia of Software Engineering*, John Wiley and Sons, 1994, pp. 528–532.

- [7] R. J. Offen, D. R. Jeffery, Establishing software measurement programs, *IEEE Software* 14 (2) (1997) 45–53.
- [8] J. McGarry, *Practical software measurement: objective information for decision makers*, Addison-Wesley, Boston, MA, 2001.
- [9] V. R. Basili, M. Lindvall, M. Regardie, C. B. Seaman, J. Heidrich, J. Münch, H. D. Rombach, A. Trendowicz, Linking software development and business strategy through measurement, *IEEE Computer* 43 (4) (2010) 57–65.
- [10] P. Berander, P. Jönsson, A goal question metric based approach for efficient measurement framework definition, in: *Proceedings of the International Symposium on Empirical Software Engineering (ISESE 2006)*, 2006, pp. 316–325.
- [11] P. Ardimento, M. T. Baldassarre, D. Caivano, G. Visaggio, Multiview framework for goal oriented measurement plan design, in: *Proceedings of the 5th International Conference on Product Focused Software Process Improvement (PROFES 2004)*, 2004, pp. 159–173.
- [12] M. G. Mendonça, V. R. Basili, Validation of an approach for improving existing measurement frameworks, *IEEE Trans. Software Eng.* 26 (6) (2000) 484–499.
- [13] F. Niessink, H. van Vliet, Measurement program success factors revisited, *Information & Software Technology* 43 (10) (2001) 617–628.
- [14] ISO/IEC15939, *Iso/iec 15939 international standard 1st edition 2002: Software engineering - software measurement process*, reference number ISO/IEC 15939:2002(e)(2002) (2002).
- [15] M. G. Mendonça, V. R. Basili, I. S. Bhandari, J. Dawson, An approach to improving existing measurement frameworks, *IBM Systems Journal* 37 (4) (1998) 484–501.
- [16] T. Olsson, P. Runeson, V-gqm: A feed-back approach to validation of a GQM study, in: *Proceedings of the 7th International Symposium on Software Metrics (IEEE Metrics 2001)*, 2001, pp. 236–245.
- [17] A. J. Boyd, The evolution of goal-based information modeling: Literature review, *Aslib Proceedings: New Information Perspectives* 57 (6) (2005) 523–538.
- [18] D. N. Card, What makes for effective measurement?, *IEEE Software* 10 (6) (1993) 94–95.
- [19] L. C. Briand, S. Morasca, V. R. Basili, An operational process for goal-driven definition of measures, *IEEE Trans. Software Eng.* 28 (12) (2002) 1106–1125.
- [20] C. Ackermann, M. Lindvall, R. Cleaveland, Towards behavioral reflexion models, in: *Proceedings of the 20th International Symposium on Software Reliability Engineering (ISSRE 2009)*, 2009, pp. 175–184.
- [21] H. A. Rubin, The top 10 mistakes in IT measurement, *IT Metrics Strategies* 2 (11).
- [22] M. Villavicencio, A. Abran, Facts and perceptions regarding software measurement in education and in practice: Preliminary results, *Journal of Software Engineering and Applications* 4 (2011) 227–234.
- [23] N. E. Fenton, Software measurement: A necessary scientific basis, *IEEE Trans. Software Eng.* 20 (3) (1994) 199–206.
- [24] B. Kitchenham, S. L. Pfleeger, N. E. Fenton, Towards a framework for software measurement validation, *IEEE Trans. Software Eng.* 21 (12) (1995) 929–943.
- [25] N. Habra, A. Abran, M. Lopez, A. Sellami, A framework for the design and verification of software measurement methods, *Journal of Systems and Software* 81 (5) (2008) 633–648.
- [26] N. E. Fenton, S. L. Pfleeger, *Software metrics: a rigorous and practical approach*, 2nd Edition, PWS, London, 1997.
- [27] Y. Wang, The measurement theory for software engineering, in: *Proceedings of the Canadian Conference on Electrical and Computer Engineering (IEEE CCECE 2003)*, Vol. 2, IEEE, 2003, pp. 1321–1324.
- [28] B. A. Kitchenham, What’s up with software metrics? - a preliminary mapping study, *Journal of Systems and Software* 83 (1) (2010) 37–51.
- [29] ISO/IEC2502-1, *Software engineering: Software product quality requirements and evaluation (square) quality measure elements*, iso, geneva, switzerland (2005).
- [30] ISO/IEC9126-2, *Software engineering: Product quality, part 2: External metrics*, iso, geneva, switzerland, 2001. (2001).
- [31] ISO/IEC9126-3, *Software engineering: Product quality, part 3: Internal metrics*, iso, geneva, switzerland, 2001. (2001).
- [32] ISO/IEC9126-4, *Software engineering: Product quality, part 4: Quality in use metrics*, iso, geneva, switzerland, 2004. (2001).
- [33] T. Kilpi, Implementing a software metrics program at nokia., *IEEE Software* 18 (6) (2001) 72–77.
- [34] M. Díaz-Ley, F. García, M. Piattini, Mis-pyme software measurement maturity model-supporting the definition of software measurement programs, in: *Proceedings of the 9th International Conference on Product-Focused Software Process Improvement (PROFES 2008)*, 2008, pp. 19–33.
- [35] M. Díaz-Ley, F. García, M. Piattini, Implementing a software measurement program in small and medium enterprises: a suitable framework, *IET Software* 2 (5) (2008) 417–436.
- [36] C. G. von Wangenheim, T. Punter, A. Anacleto, Software measurement for small and medium enterprises, in: *Proceeding 7th International Conference on empirical Assessment in Software Engineering (EASE)*, 2003.
- [37] W. Goethert, W. Hayes, Experiences in implementing measurement programs, *Tech. Rep. CMU/SEI-2011-TN-026*, Software Engineering Institute, Carnegie Mellon University, Carnegie Mellon University, USA (2001).

- [38] A. M. Bhatti, H. M. Abdullah, C. Gencel, A model for selecting an optimum set of measures in software organizations, in: Proceedings of the 16th European Conference on Software Process Improvement (EuroSPI 2009), 2009, pp. 44–56.
- [39] P. Berander, A. Andrews, Requirements prioritization, in: A. Aurum, C. Wohlin (Eds.), Engineering and Managing Software Requirements, Springer Verlag, Berlin, 2005, pp. 69–94.
- [40] J. Sawyer, D. MacRae, Game theory and cumulative voting in illinois: 1902-1954, American Political Science Review 56 (1962) 936–946.
- [41] P. Berander, M. Svahnberg, Evaluating two ways of calculating priorities in requirements hierarchies - an experiment on hierarchical cumulative voting, Journal of Systems and Software 82 (5) (2009) 836–850.
- [42] L. M. Laird, M. C. Brennan, Software measurement and estimation: a practical approach, Wiley, Hoboken, N.J., 2006.
- [43] C. Ebert, R. Dumke, Best Practices in Software Measurement, Springer-Verlag, Berlin Heidelberg, 2007.
- [44] P. H. Southehal, G. Levin, Validation of a generic GQM based measurement framework for software projects from industry practitioners, in: Proceedings of the 10th IEEE International Conference on Cognitive Informatics and Cognitive Computing (ICCI*CC 2011), 2011, pp. 367–372.
- [45] Cmmi product team: CMMI for development version 1.2 (CMMI-DEV, v1.2): Improving processes for better products. cmu/sei-2006-tr-008, esc-tr-2006-008, software engineering institute, sei (2000).
- [46] R. L. Rardin, Optimization in operations research, Prentice Hall, Upper Saddle River, N.J., 1998.
- [47] P. Runeson, M. Höst, Guidelines for conducting and reporting case study research in software engineering, Empirical Software Engineering 14 (2) (2009) 131–164.
- [48] M. Unterkalmsteiner, T. Gorschek, A. K. M. M. Islam, C. K. Cheng, R. B. Permadi, R. Feldt, Evaluation and measurement of software process improvement - a systematic literature review, IEEE Trans. Software Eng. 38 (2) (2012) 398–424.

A. Running Example

In order to make it clear how the GQM-DSFMS process works, we provide a running example. The running example is used to provide practitioners all information needed to apply the process presented as part of GQM-DSFMS, i.e. it serves as a manual.

A.1. Results of Phase 1, 2, and 3

Please note that the running example is trivial, and only used for illustrative purposes to allow readers to easily understand and implement the process. Hence, the solution to metric selection in the example is obvious, which is not the case when having a high number of goals and metrics. Figure 5 shows an example illustrating the elicited one main goal (a1.1) and three sub-goals (a2.1, a2.2, and a2.3). The goals are answered through questions, where the goal a2.1 requires the answer to question a3.1, the goal a2.3 requires the answer to question a3.2, and the goal a2.2 requires the answer from both a3.1 and a3.2.

Associated to the questions we have attributes that need to be captured as well as the metrics quantifying the attributes. Note that the metrics available for different attributes are available in the Attributes/Metrics Repository. The goals are prioritized by the stakeholders, and in this case the example shows an example of prioritization without trade-off between goals, where the stakeholders provide priority values in the range of 1 to 20.

A.2. Results of Phase 4, Step 1

Figure 6 illustrates how the inheritance of priority values from goals down to metrics works. Nodes representing attributes and metrics have a priority value of 1 as these are not prioritized by the user, their priority is a consequence of the priority of the goals. It is apparent that the highest priority is given to *Function Points* and *Lines of Code* as metrics of Size, followed by *Work-days* as a measure of Duration, and *Person-days* as a measure of Effort.

Figure 5: Example for Elicitation Result

A.3. Results of Phase 4, Step 2

Table 3 shows the calculation for Total Collection Effort (TCE) on a yearly basis. We should note that these figures are just examples to show how the calculations are made, not reflecting real figures. If the effort figures should be collected once per month within a year and it takes 10 minutes to collect this measure for one resource (person) where there is only one person who collects the metric, this gives a total collection effort of 2 person-hours. In analogy, the other TCEs are collected.

Table 3: Total Collection Effort on a Yearly Basis (Person-Days)

Attribute/Metric	T (Time to collect) (hours)	F (Frequency) (within a year)	R (number of persons)	TCE (person hours)
Effort (person days)	10 min=1/6h	12	1	2
Duration (work days)	50 min=5/6h	12	10	100
Size (FP)	200 min=3 2/6h	12	5	200
Size (LOC)	50 min=5/6h	12	2	20

Knowing the TCE, the RC is multiplied with the TCE, resulting in the Total Cost (TC) of metrics on a yearly basis in dollars. Note that tool costs might occur, and still require some manual involvement, as is the case for Lines of Code in this example.

Level 1:

$$CD = 16$$

$$a_{1,1}^n = \frac{a_{1,1}}{CD} = 16/16 = 1, a_{1,1}^f = a_{0,1}^f(a_{1,1}^n) = 1(1) = 1$$

Level 2:

$$CD = a_{2,1} + a_{2,2} + a_{2,3} = 5 + 15 + 20 = 40$$

$$a_{2,1}^n = \frac{a_{2,1}}{CD} = 5/40 = 0.125, a_{2,1}^f = a_{1,1}^f(a_{2,1}^n) = 1(0.125) = 0.125$$

$$a_{2,2}^n = \frac{a_{2,2}}{CD} = 15/40 = 0.375, a_{2,2}^f = a_{1,1}^f(a_{2,2}^n) = 1(0.375) = 0.375$$

$$a_{2,3}^n = \frac{a_{2,3}}{CD} = 20/40 = 0.5, a_{2,3}^f = a_{1,1}^f(a_{2,3}^n) = 1(0.5) = 0.5$$

Level 3:

$$a_{3,1}^f = a_{2,1}^f * \left\{ \frac{1}{1+1} \right\} + a_{2,2}^f * \left\{ \frac{1}{1+1} \right\} = 0.125 * \left\{ \frac{1}{2} \right\} + 0.375 * \left\{ \frac{1}{2} \right\} = 0.3125$$

$$a_{3,2}^f = a_{2,2}^f * \left\{ \frac{1}{1+1} \right\} + a_{2,3}^f * \left\{ \frac{1}{1+1} \right\} = 0.375 * \left\{ \frac{1}{2} \right\} + 0.5 * \left\{ \frac{1}{2} \right\} = 0.6875$$

Level 4:

$$a_{4,1}^f = a_{3,1}^f * \left\{ \frac{1}{1+1} \right\} = 0.3125 * \left\{ \frac{1}{2} \right\} = 0.1563$$

$$a_{4,2}^f = a_{3,1}^f * \left\{ \frac{1}{1+1} \right\} + a_{3,2}^f * \left\{ \frac{1}{1+1} \right\} = (0.3125 + 0.6875) * \left\{ \frac{1}{2} \right\} = 0.5$$

$$a_{4,3}^f = a_{3,2}^f * \left\{ \frac{1}{1+1} \right\} = 0.6875 * \left\{ \frac{1}{2} \right\} = 0.3438$$

Level 5:

$$a_{5,1}^f = a_{4,1}^f * \left\{ \frac{1}{1+1} \right\} = 0.1563 * \left\{ \frac{1}{2} \right\} = \mathbf{0.1563}$$

$$a_{5,2}^f = a_{4,2}^f * \left\{ \frac{1}{1+1} \right\} = 0.5 * \left\{ \frac{1}{2} \right\} = \mathbf{0.5}$$

$$a_{5,3}^f = a_{4,3}^f * \left\{ \frac{1}{1+1} \right\} = 0.3438 * \left\{ \frac{1}{2} \right\} = \mathbf{0.1719}$$

$$a_{5,4}^f = a_{4,3}^f * \left\{ \frac{1}{1+1} \right\} = 0.3438 * \left\{ \frac{1}{2} \right\} = \mathbf{0.1719}$$

Figure 6: Inheritance of Priority Values

Table 4: Total Cost (TC) of Metrics on a Yearly Basis (Dollars)

Attribute/Metric	TCE (person hours)	RC per unit Effort (\$/person-hours)	E (Expenses) (\$)	TC (\$)
Effort (person days)	2	12	--	24
Duration (work days)	100	18	--	1.800
Size (FP)	200	22.5	--	4.500
Size (LOC)	20	15	200\$ (tool)	500

A.4. Results of Phase 4, Step 3

Table 5 shows the input values for the optimization. As input for the simulation model we have to provide the cumulative importance for each metric (calculated through the priority inheritance) and the total cost of each metric (calculated using total collection effort multiplied with resource cost). In addition, we require the input of B, which is the total cost (B) that the metric program is allowed to consume. The optimization problem is formulated as follows:

$$P = \begin{cases} \max_M I(M) \\ \text{subject to } C(M) \leq 5200\$ \end{cases} \quad (11)$$

The optimization problem is solved through algorithms (brute force or dynamic programming), see below.

Table 5: Input Values to Optimization Algorithm

Attribute/Metric	C(M) Cost of Metric (\$)	I(M) Importance of Metric	Total Budget (\$)
Effort (person days)	24	0.1563	5200
Duration (work days)	1.800	0.5	
Size (FP)	4.500	0.1719	
Size (LOC)	500	0.1719	

Algorithm 1: Brute Force Algorithm

```

{Input}
 $C_T = \text{Total Allocated Cost}$ 
 $K = \text{Total number of selected measures}$ 
 $c_k = \text{Cost of the } k^{\text{th}} \text{ measure}$ 
 $i_k = \text{Importance of the } k^{\text{th}} \text{ measure}$ 

{Initialization}
 $I^* = 0$  (Optimal Importance)
 $M^* = 0$  (Optimal Set of Measures)

{Optimization}
for  $M_1 = 0$  to 1 by 1 do do
  for  $M_2 = 0$  to 1 by 1 do do
    ⋮
    for  $M_K = 0$  to 1 by 1 do do
       $Cost = M_1C_1 + M_2C_2 + \dots + M_KC_K$ 
      if  $Cost \leq C_T$  then
         $I = M_1I_1 + M_2I_2 + \dots + M_KI_K$ 
        if  $I > I_O$  then
           $I^* = I$ 
           $M^* = [M_1, M_2, \dots, M_K]$ 
        end if
      end if
    end for
  end for
  ⋮
end for
end for

{Output}
 $I^* = \text{Optimal Importance}$ 
 $M^* = \text{Optimal Set of Measures}$ 

```

Algorithm 2: Dynamic Programming Algorithm: Bottom-up approach

{Input}
 $C_T = \text{Total Allocated Cost}$
 $K = \text{Total number of selected measures}$
 $c_k = \text{Cost of the } k^{\text{th}} \text{ measure}$
 $i_k = \text{Importance of the } k^{\text{th}} \text{ measure}$

{Initialization}
 $I^*(0, \text{Any Cost}) = 0$
 $I^*(\text{Any number of metrics}, 0) = 0$

{Finding optimal importance}
for $k = 1$ **to** K **by** 1 **do**
 for $c = 0$ **to** C_T **by** 1 **do**
 if $c_k \leq c$ **then**
 if $i_k + I^*(k-1, c - c_k) > I^*(k-1, c)$ **then**
 $I^*(k, c) = i_k + I^*(k-1, c - c_k)$
 else
 $I^*(k, c) = I^*(k-1, c)$
 end if
 end if
 end for
end for

{Find optimal metrics}
 $c = C_T$
 $M^*(\forall k) = 0$
for $k = K$ **to** 1 **by** -1 **do**
 if $I^*(k, c) \neq I^*(k-1, c)$ **then**
 $M^*(i) = 1$
 $c = c - c_k$
 end if
end for

{Output}
 $I^* = I^*(K, C_T)$ *Optimal Importance*
 $M^* = \text{Optimal Set of Measures}$

A.5. Results of Phase 4, Step 3 and Step 4

Figure 7 shows the outcome of the simulation, and also the calculation of EA in fulfilling the information needs. Based on the simulation results, three metrics are proposed to be collected: effort figures in person days, duration in working-days, and size in Lines of Code. The total cost required would be 2324\$ with a information need achievement of 82.81% within the budget constraint.

Figure 7: Calculation of EA (%) in fulfilling the information needs

Author Biographies

Cigdem Gencel is a senior researcher at the Faculty of Computer Science of the Free University of Bolzano, Italy. She received her PhD from the Middle East Technical University in Turkey in 2005. Her research focus is providing novel solutions to real industrial challenges, particularly in the areas of software size and effort estimating, software measurement, software project management, requirements elicitation methods and process improvement. She is a member of the International Advisory Council of COSMIC and a member of the COSMIC Measurement Practices Committee.

Kai Petersen is a senior researcher at Blekinge Institute of Technology (BTH), Sweden. He received his PhD from BTH in 2010. His research focuses on software processes, software metrics, lean and agile software development, quality assurance, and software security in close collaboration with industry partners. Kai has authored over 30 articles in international journals and conferences, and has been the industry chair of REFSQ 2013.

Aftab Ahmad Mughal is PhD student at the School of Computing and Mathematical Sciences of the Auckland University of Technology, New Zealand. His research focus is providing solutions to Small and Medium Enterprises in area of software measurement. He has eight years of industrial experience in different multinational software development companies.

Muhammad Imran Iqbal is a senior researcher at the Blekinge Institute of Technology, Sweden. He completed his PhD degree in telecommunications in 2012 at the same institute. His research interests include image/video processing, image/video quality assessment, signal processing, embedded systems and optimization.