ELSEVIER

Knowledge-Based Systems 11 (1998) 125–143

Knowledge-Based SYSTEMS

Requirements for the knowledge-based support of software engineering measurement plans

Christiane Gresse, Lionel C. Briand*

Fraunhofer Institute for Experimental Software Engineering, Sauerwiesen 6, 67661 Kaiserslautern, Germany

Accepted 31 March 1998

Abstract

In order to improve the quality of software systems, measurement programs have been implemented in many companies to support process improvement activities. The planning and implementation of a successful measurement program requires, in practice, a significant amount of effect. Cost may be reduced and quality of measurement may be improved by providing knowledge-based support and reusing experiences gathered on past measurement programs. In this article, we state the requirements for the knowledge-based support of planning measurement programs based on the Goal/Question/Metric paradigm. Reuse opportunities are precisely identified, the knowledge to be captured for effective reuse is identified and structured, and reuse scenarios are provided.

Keywords: Reuse; Software measurement; Process improvement

1. Introduction

For the continuous improvement of software quality and productivity within organizations, capturing and reusing explicit software development know-how is essential. What represents relevant software know-how differs among software organizations, depending on their specific development environment and objectives. Therefore, this software know-how has to be derived by organization-specific measurement programs, taking into account the environment characteristics and company-specific improvement goals, e.g. reduce development effort by 10% during the next year. For continuous improvement of software processes and products, the establishment of goal-orientated measurement programs is essential for deriving relevant quantitative and qualitative data on software processes and products. In this context, the Goal/Question/Metric Paradigm (GQM) [1,5] is of particular interest, since it helps in defining and implementing operational and measurable software improvement goals. However, measurement programs are known to be still difficult to plan and implement, especially when one lacks the required experience. Similarly to software development know-how, in order to institutionalize systematic and continuous learning in the organization, measurement know-how has also to be explicitly captured, modeled and reused to support the planning of future measurement programs. Therefore, experiences regarding the use of GQM, the know-how related to its products and processes, needs to be captured and made reusable organization-wide.

In this paper, we focus on requirements and principles for the knowledge-based support of planning GQM-based measurement programs. A motivation for the reuse of experiences and know-how from past measurement programs is given in Section 2. The potential benefits of experience-based support are discussed in Section 3. The knowledge which should be captured in a reusable form for future measurement programs is identified in Section 4. Section 5 addresses the modeling and representation of this knowledge in an experience base. Scenarios for the reuse of the knowledge are provided in Section 6. Conclusions and future research directions are discussed in Section 7.

2. Motivation

Building up organization-specific software development know-how, the company has to learn continuously from its software projects by developing tailored, context-specific quality and resource models based on quantitative and qualitative data derived through measurement programs. To be effective and efficient, measurement programs must also be tailored to the characteristics of the specific organization, the software processes and products and

* Tel.: +49 6301 707 251; fax: +49 6301 707 200; e-mail: gresse,briand@iese.fhg.de

0950-7051/98/$ - see front matter
PII: S0950-7051(98)00043-4

company-specific goals. Therefore, one of the major success factors of a measurement program is its appropriate planning. Presently, the planning of measurement programs is performed, in most cases, from scratch or based on an ad hoc, non-supported, reuse process. This process is usually costly and the likelihood of committing mistakes is high since the adequate planning of the measurement program requires expertise and specific know-how. In general, all products of a GQM-based measurement program are potentially reusable, from measurement goals to data-collection instruments. Although there is often a need to tailor these products to specific project characteristics, they are reusable to a large extent. However, reusing products already developed in the organizational context will require less effort and will be more likely to address the needs. But today, in most cases, the organization-wide dissemination of the products and lessons learned gathered in measurement programs is not supported, since only project-specific data is stored. Project-specific knowledge is not generalized or formalized in order to widen the context in which it can be reused. This would require sophisticated ways of storing knowledge to allow intelligent search, adaption of knowledge, and specific navigation mechanisms in knowledge bases. Concerning the performance of measurement programs, experiences with respect to the development of GQM products or the use of the derived quality and resource models should be captured to help improve the planning of measurement programs over time.

In conclusion, great benefits can be expected by supporting the planning phase of a measurement program through the reuse of knowledge from past measurement programs. However, the complexity of measurement plans makes the understanding and the identification of relevant and reusable measurement products difficult. This is exacerbated by the complex net of interdependences between GQM products. In this paper, we focus on the identification of the requirements for the knowledge-based support for the extensive reuse of GQM products in order to facilitate the planning phase of a measurement program.

3. What activities need knowledge-based support?

In this section we describe the GQM planning process for which we claim knowledge-based support ought to be provided. The GQM approach is a specific technology for goal-orientated measurement in software projects. In order to be tailored to the needs, GQM-based measurement programs have to be specified precisely and explicitly by a detailed measurement goal. The measures are derived in a top-down fashion, based on goals via a set of questions and quality/resource models. This refinement is precisely documented in a GQM plan, providing a rationale for the selection of the underlying measures. The data collected is interpreted in a bottom-up fashion in the context of the GQM goal, questions and models, considering the limitations and assumptions underlying each measure. Based on experience in applying the GQM approach across several companies, we briefly describe the process to plan and execute a GQM-based measurement program [1,7,6].

GQM1—Prestudy: The environment in which the measurement program takes place is characterized. This characterization includes the description of the business and improvement goals of the organization, its development processes, its organizational structure and the projects to be measured.

GQM2—Identification of GQM goals and development of GQM plan: Based on the characterization of the measurement environment, the goals of the measurement program are defined according to a precise template (Section 3.1). The explicitly defined measurement goals are refined into a set of relevant measures via questions and quality and resource models, resulting in a GQM plan consisting of a goal, questions, related quality/resource models and measures.

GQM3—Development of measurement plan: A measurement plan integrating the measures of the GQM plan into the development process of the studied software project(s) is developed. The measurement plan defines when, how and by whom the required data can be collected. Appropriated data-collection instruments are developed, e.g. questionnaires, static analysers.

Table 1
GQM goal template summary [1]

Dimension	Definition	List of examples	Example
Object	What will be analysed?	Processes, products, resource.	Software development process.
Purpose	Why will the object be analysed?	Characterization, evaluation, monitoring, prediction, control, improvement.	Characterization.
Quality Focus	What property of the object will be analysed?	Cost, fault-proneness, defect removal capability, user friendliness.	Fault-proneness.
Viewpoint	Who will use the data collected?	User, senior manager, project manager, developer, system tester, quality assurance manager.	Software developer.
Context	In which environment does the analysis take place?	Organization, project, problem, processes, etc.	Company XYZ.

GOAL	software development process	characterization	fault-proneness	sw developer	company xyz
Quality Focus 1 total # of failures 2 # of failures per criticality (uncritical, critical, other) 3 # of faults per life cycle phase of introduction (REQ, HLD, LLD/IMP) 4 total rework effort (in hours)			**Variation Factors** **Process Conformance:** 1 type of code inspections **Domain Understanding:** 2 experience of development team members		
Baseline Hypothesis 1 total # of failures: 100 2 75% uncritical, 25% critical 0% others; 3 REQ 20, HLD 20, LLD/IMP 40 4 total rework effort: 1000 h			**Impact on Baseline Hypothesis** **Process Conformance** 1 with ad-hoc code inspections less faults will be found than with other types of inspections **Domain Understanding** 2 more experienced development team members introduce less faults		

Fig. 1. Abstraction sheet—example.

GQM4—Data collection and analysis and interpretation: During the execution phase of the measurement program, the data are collected according to the data-collection procedures specified in the measurement plan. Following the GQM plan bottom-up, the collected measurement data are analysed and interpreted in feedback sessions involving development participants. These sessions have two objectives: to provide feedback to the ongoing software project, and to help the analyst interpret the data-analysis results.

GQM5—Post-mortem analysis: The measurement results including the collected data and their interpretation are analysed after project completion, focusing on feedback to the organization as a whole, e.g. enrich the corporate cost model with this new project data point. Knowledge gained through measurement is formulated in relation to relevant context factors, e.g. project cost factors.

GQM6—Packaging: The analysis results are packaged in a way suited to the organization context so that reuse of this knowledge in future software projects and measurement programs is effective, e.g. cost model packaged in a tool allowing users to specify project uncertainties regarding certain cost factors and perform sensitivity analysis of these factors on the model's outputs.

In this paper, we focus on the knowledge-based support for the planning of a measurement program. Therefore, the following process steps are mainly considered: identification of measurement goals, development of GQM plans, and development of measurement plans. These steps are described in the following sections.

3.1. GQM2A: Identification of GQM goals

When planning a goal-orientated measurement program,

Process definition

Q_1Has the experience of the development team members an impact on the number of faults?
Hypothesis: more experienced development team members introduce less fault.

Q_1.1What is the distribution of experience among the development team members?
Hypothesis: 30% 3 or more years, 20% 1-3 years, 50% less than one year of experience

Q_1.2What is the proportion of faults detected before delivery?
Hypothesis: 90% of the faults will be detected before delivery

Quality definition

Q_2What is the overall number of failures reported before delivery?
Hypothesis: there will be about 120 failures reported before delivery.

Q_3What is the distribution of failures reported before delivery by criticality?
Hypothesis: 75% uncritical failures, 25% critical failures.

Q_4What is the distribution of faults by life cycle phase of detection before delivery?
Hypothesis: estimated distribution is REQ: 20, HLD 20, LLD/IMP: 40

Q_5What is the total rework effort proportion?
Hypothesis: total rework proportion is 20 percent.

Fig. 2. GQM plan—questions and hypotheses.

Table 2
Question categories [1]

Category	Description
Quality Focus	Questions concerning the quality model(s) to be used which define further the quality focus stated in the goal.
Process/Product Definition	Questions concerning factors that may have an impact on the values of the quality models. Depending on whether the object of study is a process or product, this category is referred to as either product or process definition.
Process definition–Process conformance	Questions attempting to capture information concerning the adherence of the actual to the official organizational process.
Process definition–Domain understanding	Questions concerning the attributes of objects used by the process under study and the actors performing the process.
Product definition	Questions concerning logical and physical attributes of the product, development cost related to the product, changes to the product and the operational context of the product.

the first step is to specify the goals to be achieved by measurement. Depending on the business and improvement goals of the organizations and existing problems in the software process (e.g. ineffective reviews), potential measurement goals have to be identified with great care by all stakeholders of the measurement program. The relationship between the business and improvement goals and the potential measurement goals under consideration is usually not evident, especially for people lacking experience regarding measurement. Reuse can address this issue, as discussed in the next sections. A template [1,5] guiding the definition of goals is structured according to five facets as described in Table 1.

3.2. GQM2B: Development of the GQM plan

Based on the measurement goal, a GQM plan is developed, consisting of the following components [1,7]:

- a goal, defining the object, purpose, quality focus, viewpoint, and the context of the measurement program;
- a set of questions, operationalizing the goal;
- a set of models, specifying how to answer the questions;
- a set of measures, operationally defining the data to be collected to feed the models.

Activities supporting the development of the GQM products above are described in the subsections below.

3.2.1. Knowledge acquisition

To refine the goal into operational measures, the quality focus needs to be defined adequately and fit the viewpoint's need. Relevant knowledge is acquired by interviewing the people stated in the viewpoint of the GQM goal. This information is used to derive valid and correct quality models, to identify relevant context factors, and therefore relevant measures. The interviews cover the following topics [9].

- Quality focus: It specifies what the quality focus means to the interviewees. The quality focus is usually composed of a set of quality dimensions. See, for example, how fault-proneness is defined in the upper-left quadrant of Fig. 1.
- Baseline hypothesis: For each quality dimension pertaining to the quality focus, an expected distribution of values may be stated, based on the interviewee's intuition and understanding of the environment.
- Variation factors: The factors that are expected to have an impact on the quality dimensions are stated.
- Impact on baseline hypothesis: For each variation factor, the expected impact of the variation factor on the quality dimension should be specified, when possible.

A commonly used instrument for the acquisition and structuring of knowledge during the interviews is the abstraction sheet [9]. An abstraction sheet is a one-page document with four quadrants, one for each of the above-mentioned topics, and the respective GQM goal in the header (see Fig. 1). Abstraction sheets are usually developed from scratch. However, parts of abstraction sheets may be relevant from one project to another or from one division to another within an organization. Therefore, reuse of abstraction sheets from past measurement programs could provide a first overview on potentially relevant knowledge regarding

Predictive Model: Effectiveness of inspections

Context: company xyz, project type abc

Assumptions: The larger the document inspected, the larger the fatigue effect, the more complex the inspection, i.e., its cognitive load.

Model description: The model predicts the effectiveness of inspections.

effectiveness e = a * (size of document)^b [a,b parameters; b<0]

Attributes: size of documents, effectiveness

Measures: count of operations specified

Fig. 3. GQM plan—models.

Q_2What is the overall number of failures reported before delivery?

M_2.1count of validated failure reports turned in before delivery [ratio: integer][1]

Q_3What is the distribution of failures reported before delivery by criticality level?

M_3.1: classification by criticality [ordinal:uncritical;critical] broken down according to a classification scheme of defect criticality

Q_4What is the distribution of faults per life cycle phase of detection before delivery?

M_4.1count of fault per life cycle phase where the fault was introduced [nominal: REQ, HLD, LLD/IMP]

Q_5What is the total rework effort?

M_5.1for all failures reported before delivery: effort to isolate the faults that caused the failures (person-hours) [ratio: integer]

M_5.2for each fault detected before delivery: effort to correct the fault (person-hours) [ratio:integer]

1.[level of measurement: range]

Fig. 4. GQM plan—measure level—example.

a measurement goal, e.g. possible relevant quality dimensions and baseline hypotheses. Furthermore, potential problems which may arise while refining the questions, models and measures, due to possibly incomplete or insufficiently defined knowledge acquired during the interviews, could be addressed in advance. For example, if experience of developers is considered as a variation factor, it is also necessary to capture how experience is classified in the specific environment and define the factor and its categories explicitly.

3.2.2. Development of the question level

The GQM plan is developed based in part on the knowledge acquired during the interviews. For each quality dimension and variation factor documented in the abstraction sheet, questions in the GQM plan are defined, expressing a need for information [1,7]. The question represent a first step towards the operationalization of the measurement goal. A hypothesis may specify an answer's type or pattern which is expected for a question and be stated explicitly. For example, hypotheses commonly specify expected distributions and relationships (and changes thereof) concerning quality dimensions or variation factors which are the focus of questions. Since GQM plans usually consist of large numbers of questions, questions categories [1,5] have been defined in order to drive the derivation of questions and help structure GQM plans and ensure their completeness (see Table 2). Continuing the example above, questions and hypotheses are illustrated in Fig. 2. Deriving a complete set of precisely formulated questions is a complex and difficult task. It can be supported by a precise mechanism suggesting potentially interesting questions with respect to the measurement goal and the specific context.

3.2.3. Development of quality/resource models

Each question in the GQM plan is formalized by defining detailed quality or resource models [1]. These models formalize and provide ways of answering the questions (see Fig. 3). By defining such models, we operationalize the questions of the GQM plans, provide ways of quantifying attributes, and define precisely how quality/productivity comparisons, evaluations, and predictions are to be performed [1]. The models have to be developed by taking into account the studied environment's specifics, since they usually have to make simplifying assumptions. Therefore, the environment's characteristics, standards and terminology have to be well known. The models are often developed based on abstract concepts, e.g. size or complexity. Thus, they have to be refined into operational descriptive models, leading to measurement. Models have to be carefully analysed in order to determine the validity of their underlying assumptions and their applicability in the particular environment under study. Developing these models from scratch for each question of the GQM plan is a complex intellectual task and requires a large amount of effort. By suggesting models, which have been developed in the past, this activity could be supported by reuse. The applicability of these models could be assessed directly, based on their underlying assumptions. In addition, instantiations of these models based on data collected in past measurement programs can provide insights on plausible expected values and be used as a basis of comparison between projects.

3.2.4. Definition of measures

The questions are refined quantitatively into a set of measures via quality//resource models [1]. The measures define what data have to be collected in order to feed the models. For a given attribute, the definition of a measure includes the selection of a level of measurement, unit and range. Continuing the example above, Fig. 4 provides simple and incomplete examples of measures. The definition of measures can be strongly supported by the reuse of measures from past measurement programs which happen to be suited to the selected models. Appropriate measures can be selected based on their properties and assumptions.

Table 3
Measurement plan—example

Name	Object/ Attribute	Unit	Level	Range	Event	Point in time	Resource/ Data-collector	QA responsible	Instrument ID
Project effort	sw project/ effort	[person-months]	ratio	≥ 0	periodic	week	Human/ developer	Project manager	ER-*;1
Failure criticality	failure/ criticality	—	ordinal	[uncritical, critical, other]	process_end	failure_ handling	Human/tester	GQM team	PR-d;1
Fault intro. phase	fault/ intro_phase	—	nominal	[REQ, HLD, LLD/IM]	process_end	fault_handling	Human/ developer	GQM team	PR-d;2

3.3. GQM3: Development of measurement plan

The main focus of the measurement plan is the appropriate integration of measurement into the software development process. Data collection procedures are defined by determining for each measure identified in the GQM plans, when, how and by whom the data are collected [1,7]. When the required data can be collected, e.g. at the beginning or end of an activity, has to be defined in relation to the software process. For each measure, the people or tools that could possibly provide the data have to be identified, e.g. project manager, and the persons responsible for the quality assurance and handling/storage of the data have to be determined. Data-collection instruments have to be designed or reused in order to support data collection, for example, tools (e.g. static code analysers), questionnaires (e.g. NASA SEL forms [12]) or structured interviews. A small excerpt from a measurement plan is given in Table 3. The definition of the measurement plan is a difficult and complicated process, e.g. many decisions have to be made with respect to procedures, instruments, etc. Therefore, a detailed knowledge about the software process and the organizational structure is necessary. To implement the measurement plan, the required data-collection instruments have to be developed. Depending on the type of data-collection instrument, either a tool has to be developed, questionnaires have to be designed or structured interviews have to be prepared and planned. The data-collection instruments have to be tailored to the development environment and this requires a detailed knowledge of the organizational structure, workflow and information flow, standards and terminology. For example, the reliability of the data collected by a questionnaire and its usability must be ensured. In order to collect reliable data, the questions have to be carefully formulated, all unknown terms explained, and open questions limited, to the extent possible. Usability has to be achieved in order to reduce the effort related to the data collection. Therefore, the questionnaire has to be well structured, stating questions in a logical order, etc. As those examples show, the questionnaires should be pretested before they are actually used for the data collection. Based on the description above, it is clear that the design of measurement procedures and instruments, such as interviews and questionnaires, is an expensive and complex task. This strongly motivates the need for their extensive reuse.

4. What kind of knowledge is reusable?

In the previous section, we provide an overview on the GQM planning process and the activities that need knowledge-based support. In this section, we examine the opportunities in terms of knowledge reuse for the GQM planning process. A taxonomy of reusable components in the planning phase is defined in detail in the next subsections.

4.1. Types of knowledge

For each product or process related to the GQM planning process, knowledge with potential benefit for reuse can be identified. The reusable knowledge can be classified into certain types according to their objective, content and advantages. For example, as a basis for the development of a measurement program, GQM products from past measurement programs can be used, reducing the necessary development effort. In general, the following types of reusable knowledge in the GQM planning process can be identified.

Templates: They facilitate the development of GQM products, e.g. goal templates in Section 3.1. They are usually derived from common patterns in past measurement programs and provide a very detailed, structured and direct support. The (re-)use of templates can significantly reduce effort and risks. Furthermore, a consistent view and understanding across various projects and measurement programs is more likely to be achieved through the use of common templates. Thus, organizational learning takes place owing to the (re-)use of templates which are improved and evolved over time.

Taxonomies: They represent ordered arrangements of entities according to their presumed relationships, e.g. the IEEE quality terms standard. Taxonomies of entities related to measurement programs can be used for an appropriate refinement of the objects of interest in the measurement program. Choices are directed and measurement planning is guided by presenting decision guidelines based on taxonomies.

Glossaries: They define a terminology related to GQM entities as used in a specific environment, e.g. the *IEEE Standard Glossary of Software Engineering Terminology*. The appropriateness of the developed GQM products can

be increased by using a consistent, environment-specific terminology and standards. The (re-)use of a glossary can support the adequate use of terms, their consistency across an organization, and prevent misunderstandings and communication problems between the stakeholders of the measurement program, e.g. developers and senior management. Thus, it facilitates the definition and use of the GQM entities.

Alternative GQM entities: GQM entities developed in past measurement programs and with similar contexts can be (partly) reused and adapted for a new project. The particular preconditions for their use and their underlying assumptions provide an explicit rationale for decisions regarding their reuse. An example of a GQM entity, an abstraction sheet, is shown in Fig. 1. Reusing this knowledge is expected to reduce the effort required by the GQM planning procress and also to improve the quality of the GQM entities with respect to their reliability, adequacy, completeness and consistency. However, it is likely that the reused GQM entities possibly have to be adapted to specific characteristics of the new projects where they are applied.

Lessons learned: They explicitly capture strategies for the mapping between problems and solutions that have been adopted on past measurement programs, their context of use, and information regarding their degree of success. For example, while it has been difficult to define the appropriate object of study of the goal in a past measurement program, the software process has been modeled first and the descriptive process model has been used as a basis for the identification of the goal. This supports the finding of an adequate solution fitting the application context and helps set reasonable expectations.

For each of the phases/activities of the GQM planning process, support can be provided through the following elements:

Description of phases/activities: They guide the execution of the phase/activity. The explicit availability of a process description (see Fig. 18), will improve the consistency and correctness of its execution and, therefore, the quality of the GQM product and its cost-effectiveness. This can be further improved by using more formal process models which have already been tailored to the organization-specific needs in past measurement programs. Reusing these models also provides a basis for the continuous improvement of the process.

Description of the GQM products: Describing the structure and expected content of products can assist their development, e.g. ensure their completeness and consistency. An example is question categories (see Table 2) structuring the GQM plan. As a result, the development effort is expected to decrease and the quality of the products to improve. The usage of a unified, explicit product structure can also increase consistency across projects and allow the iterative improvement from one project to another.

- GQM products
  - Goal
    - Object
      - *Glossary*
      - *Taxonomy of objects*
    - Purpose
      - *Template*
    - Quality Focus
      - *Glossary*
      - *Taxonomy of qualities*
    - Viewpoint
      - *Organizational structure*
      - *Roles and their responsibilities*
    - Environment
      - *Template*
      - *Glossary*
    - *Lessons learned*
  - Questions
    - *Question categories*
    - *Lists of questions*
    - *Lessons learned*
  - Models
    - Quality Models
      - *Taxonomy*
      - *Alternative models*
      - *Lessons learned*
    - Resource Models
      - *Taxonomy*
      - *Alternative models*
      - *Lessons learned*
  - Attributes
    - *Glossary*
    - *Taxonomy*
    - *Lessons learned*
  - Measures
    - *Taxonomy*
    - *Catalog of measures*
    - *Lessons learned*
  - Data Collection Procedure
    - *Taxonomy*
    - *List of procedures*
    - *Lessons learned*
  - Data Collection Instruments
    - *Tools*
    - *List of tools*
    - *Lessons learned*
    - Questionnaires
      - *Taxonomy*
      - *Lessons learned*
  - Knowledge Acquisition Instruments
    - Abstraction Sheet
      - *Alternative abstraction sheets*
      - *Lessons learned*
    - Interview plans
      - *Alternative abstraction sheets*
      - *Lessons learned*
- GQM Planning Process
  - For each activity:
    - *Process description*
    - *Descriptions of products*
    - *Guidelines on how to do it*
    - *Problem/solution statement*
    - *Cost/effort information*

Fig. 5. Taxonomy of GQM entities.

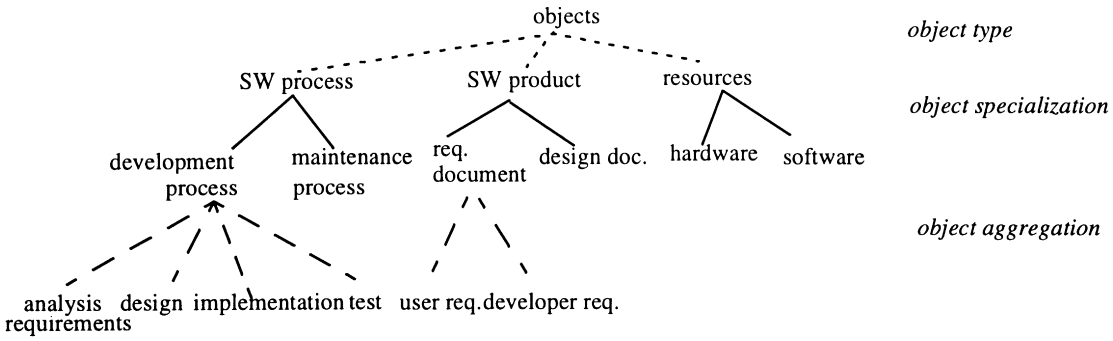


Fig. 6. Taxonomy of objects—example.

Guidelines: Guidelines and heuristics provide additional practical and intuitive support for the performance of processes/activities, e.g. select the GQM goals rigorously: concentrate on only the most central issues and a few important goals. They can be derived from experiences gathered during the establishment of GQM-based measurement programs. This again can increase effectivity and efficiency of the process performance.

Problem/solution statements: They provide knowledge about problems which were encountered in the past and the extent to which the solutions applied in the past have been successful. For example, in a past measurement program invalid data were provided owing to missing motivation of the participants. As a solution, additional goals were included in the measurement program, reflecting the interests of all persons involved. Problem/solution statements can help prevent problems early on and also provide suggestions for potential solution strategies.

Cost/effort information: Knowledge about the cost and effort related to the GQM phases/activities can help the management of a measurement program by providing quantitative data for planning and control. An example of effort data is that about 70% of the measurement effort is related to the planning phase and 30% to the collection, analysis and interpretation of the data [8]. The knowledge can be derived by setting up measurement accompanying all measurement programs focusing on the required cost and effort.

4.2. Taxonomy of GQM entities

Products and processes related to the GQM planning process which may be reused are structured through the concept of GQM entities. The term GQM entity summarizes two main classes: phases/activities of the GQM planning process and GQM products consumed or produced by the process. A taxonomy of GQM entities is defined in Fig. 5. It shows the decomposition of GQM products and process into single GQM entities. For each of these GQM entities, different types of knowledge can be reused, as indicated in Fig. 5 in italics.

In the next subsections, we detail the types of reusable knowledge for all relevant GQM entities.

4.2.1. Knowledge about the measurement goal

Goals from past measurement programs can be suggested as initial candidates for the goals of the present measurement program. Adequate goals are identified considering measurement programs with similar organizational/project characteristics and improvement goals.

Knowledge about objects of study can be provided in the form of a taxonomy of objects describing a stepwise specialization of objects which can be analysed in a measurement program. The taxonomy represents a faceted classification [13] by classifying the objects along several dimensions, as illustrated in Fig. 6. For example, a software product can be classified as a requirement document, design document, etc., or be decomposed into modules. The structure of the taxonomy reflects the organization-specific understanding of these objects. The identification process is guided by using the taxonomy as a decision tree, refining the objects step-by-step, until an object (or several) of interest in the present measurement program has been identified. A glossary defining the environment-specific terminology related to potential objects of study enables the consistent

maintenance process: The process of modifying a software system or component after delivery to correct faults, improve performance or other attributes, or adapt to a changed environment.
requirement analysis: The process of studying user needs to arrive at a definition of system, hardware, or software requirements.
requirement document: A document that specifies the requirements for a system, hardware item, or software item are presented to project personnel, mangers, users, customers, or other interested parties for comment on approval. software: Computer programs, procedures, and possibly associated documentation and data pertaining to the operation of a computer system.
test: An activity in which a system or component is executed under specified conditions, the results are observed or recorded, and an evaluation is made of some aspect of the system or component.

Fig. 7. Excerpt of a glossary for 'objects of study'—example [10].

Table 4
Measurement purposes [1]

Purpose	Description
Characterization	Aims at forming a snapshot of the current state/performance of the software development products and processes.
Evaluation	Aims at comparing and evaluating products and processes.
Monitoring	Aims at following the trends/evolution of the state/performance of processes and products.
Prediction	Aims at identifying relationships between various process and product factors using these relationships to predict relevant external attributes of products and processes.
Control	Aims at identifying causal relationships that influence the state/performance of processes and products.
Change	Aims at identifying causal relationships in order to change the development process to obtain higher product qquality and process productivity.

definition of the object of study across projects. For example, terms may be defined based on the *IEEE Standard Glossary of Software Engineering Terminology* (see Fig. 7). This can be used as a starting point and be refined according to environment-specific characteristics.

A template for possible purposes of a measurement program [1,5] has been defined. The template, as shown in Table 4, explicitly defines the potential motivations for measurement and can be refined in a given environment. The (re-)use of the template can assist the selection of the appropriate purpose, and a consistent use of these terms across projects is more likely.

As the objects of study, the quality foci can be classified with respect to several dimensions. An example of a taxonomy of quality foci specialized with respect to one dimension is presented in Fig. 8. An explicit taxonomy supports the identification of relevant quality attributes for a given project and the precise specialization of this quality focus to an appropriate level of granularity is facilitated. Explicit definitions of quality attributes in a glossary (e.g. [11] as shown in Fig. 9), at all levels of refinement, support a consistent and precise understanding about the quality foci. Therefore, misunderstandings are prevented and repeated redefinitions of these terms are no longer necessary.

Support on the identification of the viewpoint can be provided by specifying the organizational structure, roles and responsibilities. The organizational structure captures important properties of the organizational context, such as information flow and coordination between process participants (see Fig. 10). As organizational structures are often complex, an explicit model helps determine the potential uses and users of the measurement results by exposing interdependences between all roles involved in the software process. Explicit, organization-specific descriptions of the roles and their responsibilities involved in the software process further enables the precise and appropriate selection of potential stakeholders in the measurement program (see Table 5).

A characterization template structures the context information for the measurement program and points out relevant characteristics of the environment and the software project, where the measurement program takes place (see Table 6). Characterization can be facilitated by reusing these templates across the organization. The templates can be developed based on past characterizations and are likely to be refined over time. A glossary defining the context-specific terminology, e.g. *version* can be specified as a initial release or re-release of a software product, supports the adequate and consistent use of terms in the characterization of the actual measurement program. Lessons learned about the measurement goals in past measurement programs can inform whether they have been achieved successfully, which problems they encountered, and how they have been solved. This can be done in the form of problem/solution statements, describing the problem that occurred, its solution and the resulting outcome. Concerning measurement goals, common types of lessons learned are:

- Interdependences of the GQM goal to the underlying improvement program through concerning problems with respect to the contribution of the measurement program to the organization improvement goals. For example, if the organization focuses on the improvement of quality in terms of reliability, the definition of a measurement goal focusing on effort distribution is inappropriate.
- Interdependences between different goals in a measurement program, e.g. regarding a cost/benefit analysis it can be advantageous to split the measurement goal into

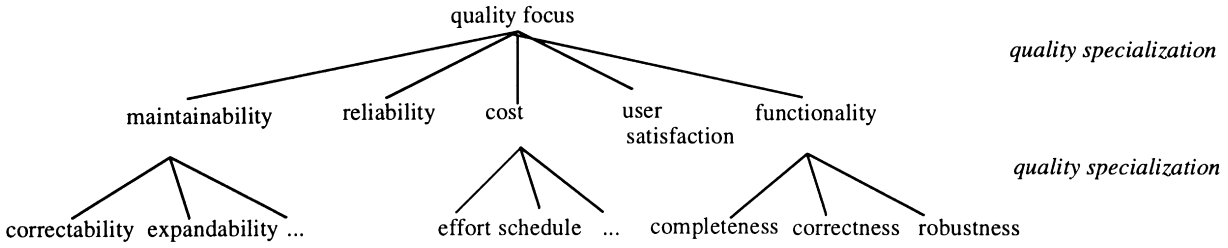


Fig. 8. Taxonomy of quality foci—example.

Correctability: The degree of effort required to correct errors ins software and cope with user complaints.
Efficiency: An attribute that bears on the relationship of the level of performance to the amount of resources used under stated conditions.
Expandability: The degree of effort to improve or modify the efficiency or functions of software.
Maintainability: An attribute that bears on the effort needed for specific modifications.
Reliability: An attribute that bears on the capability of software to maintain its level of performance under stated conditions for a stated period of time.

Fig. 9. Excerpt of a glossary of quality attributes [11]—example.

two separate goals, one focusing on costs and a second one focusing on benefits.
- Relationships to organizational and project characteristics, e.g. between the object of study and the existence of a descriptive process model in organization. For example, if a problem occurred owing to the high-level definition of the object of study (implementation process), because no explicit process model was available, which would have allowed a more precise definition of the object. As a solution, a descriptive process model was developed before planning the measurement program. Then, due to the availability of a descriptive process model, the phase and activities of the software process could be identified precisely.

Capturing explicitly problem occurrences regarding specific GQM products can prevent the occurrence of these problems in the future. Capturing solution strategies, beside the problem descriptions, helps address emerging problems by providing knowledge on solutions which have been applied successfully in the past.

4.2.2. Knowledge about questions in the gqm plan

Questions defined in a GQM plan can be structured according to question categories defined by [1,5], as described in Table 2. Reusing these categories guides the complete identification of relevant types of questions and helps structure the GQM plan. Questions from past measurement programs used for similar goals can be suggested as potential candidates for questions in the current measurement program. Examples of questions have been illustrated in Fig. 2. Reusing questions from past measurement programs can reduce the effort and help achieve higher completeness, although it usually requires some amount of tailoring. In addition, recurrent question patterns are created and refined across software projects and over time. Lessons learned on the use of questions can be provided, improving the effectiveness and efficiency of measurement:

- Interdependences of questions to goals, e.g. if a specific question was derived with respect to. a GQM goal, but turned out to be irrelevant or had to be reformulated in order to express the need for information more adequate.
- Interdependences between different questions of the GQM plan. For example, if a specific question on the impact of tool availability on required redevelopment effort, only becomes relevant in relation to the analysis of development effort.

4.2.3. Knowledge about models

Each question in the GQM plan is operationalized by defining detailed quality or resource models. Resource models describe, evaluate, or predict resource consumption in the context of software development and maintenance, e.g. effort distribution per phase of the software development process. Quality models describe quality attributes of all kinds of products or processes. Quality models can be, e.g. model of fault distribution per phase of origin in the software development process. A taxonomy describing the refinement of abstract concepts into detailed

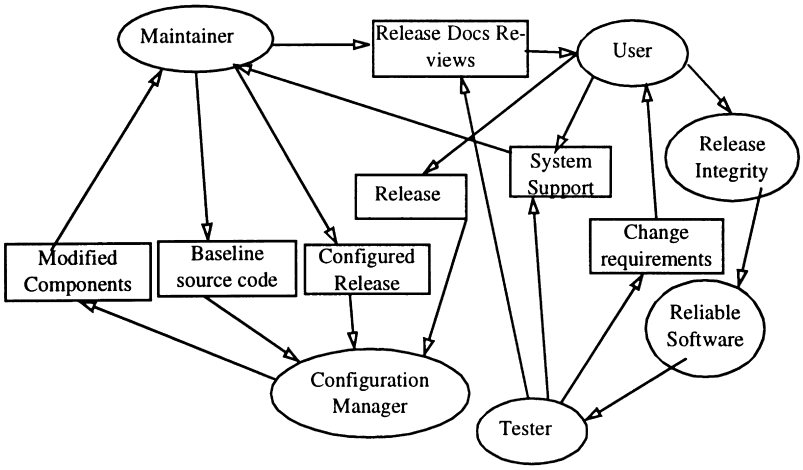


Fig. 10. Excerpt of an actor-dependence model of a maintenance organization [4]—example.

Table 5
Descriptions of organizational roles [4]—example

Roles	Responsibilities
Testers	Present acceptance test plans, perform acceptance test and provide change request to the maintainers when necessary.
Users	Suggest, control and approve performed changes.
QA Engineer	Controls maintainers' work (e.g. conformance to standards), attends release meetings and audits delivery packages.
Maintainers	Analyse changes, make recommendations, perform changes, perform unit and change validation testing after linking the modified units to the existing system, perform validation and regression testing after the system is recompiled by the *Configuration Manager*.
Management	Is officially responsible for selecting software changes, gives official authorization and provides the budget.

quality/resource models, guides the selection of suitable models for the questions in the GQM plan and provides a decision rationale (see Fig. 11). It also structures the refinement of models into different levels of abstraction. This drives the derivation of appropriate models in a specific organization. Descriptions of alternative models in each category and their underlying assumptions can be used for the more refined selection of an appropriate model. An example of a quality model is provided in Fig. 3. it can be further supported by modeling relations between questions and models used in past measurement program, pointing at possible relevant quality/resource models for a given question. The definition of these models requires in particular a large amount of effort and expertise in measurement. Reusing models developed in the past may, therefore, considerably reduce this effort, although adaptations to specific project characteristics might be necessary. Further improvement of the effectiveness and efficiency can be achieved by lessons learned on the development and use of the models:

- Interdependences of models to context characteristics. For example, if the use of a specific cost model has been inadequate for a specific type of software project.
- Interdependences of models to questions of the GQM plan. For example, if a specific model has been advantageous for answering a specific question in the past.
- Interdependences of models to quality foci. For example, regarding the appropriateness of certain models in order to achieve a measurement goal.
- Interdependences of context factors on models. For example, if in the past the expected impact of a particular context factor, e.g. availability of tool support, on an effort model could not be confirmed.
- Interdependences between different models, such as, for example, between fault and failure models.

Table 6
Environment characterization template—example

Environment information	Company XYZ
Size of organization (No. people)	100
Percentage of SW people	20%
Industrial sector(s)	telecommunication
Products and/or services marketed	telephone systems
Life-cycle model	waterfall
Tools used	CASE tools
Type of SW produced	embedded systems
Average No. of installations at the customer	1000
Crucial quality aspects	reliability, maintainability

4.2.4. Knowledge about attributes

Attributes are the properties of the objects analysed in the measurement program. The attributes are measured and integrated into quality/resource models in order to answer questions. A glossary of standards and environment-specific terminology [11] can be reused, providing a consistent overview on the organization-specific terminology (see Fig. 12). For the derivation of measures from questions and quality/resource models, existing knowledge about relevant object attributes in the software process has to be considered. The definition of measures by defining the attributes operationally can be supported by providing a taxonomy of attributes in relation to the quality/resource models and possible measures for each attribute category (see Fig. 13). (Re-)using this knowledge is expected to reduce the effort for the definition of the measures and the tailoring to objects of the software process under study. Various types of lessons learned regarding the attributes can be provided:

- Relationships of attributes to models describing the relevance of particular attributes for the definition of a model.
- Relationships of attributes to quality foci, e.g. the complexity of a software systems has an impact on the effort for the software project.
- Interdependences between different attributes, e.g. with increasing size of the system, the complexity will also increase.

4.2.5. Knowledge about measures

The operational definition of the attributes can be supported by taxonomies, a catalog of measures, their properties, and lessons learned gathered from their past application. The measures for each type of attribute can be refined into a taxonomy (see Fig. 14). The taxonomy guides the identification of the appropriate and applicable measures for each attribute, by representing a decision tree capturing different measurement properties. For example, during the definition of a coupling measure, various

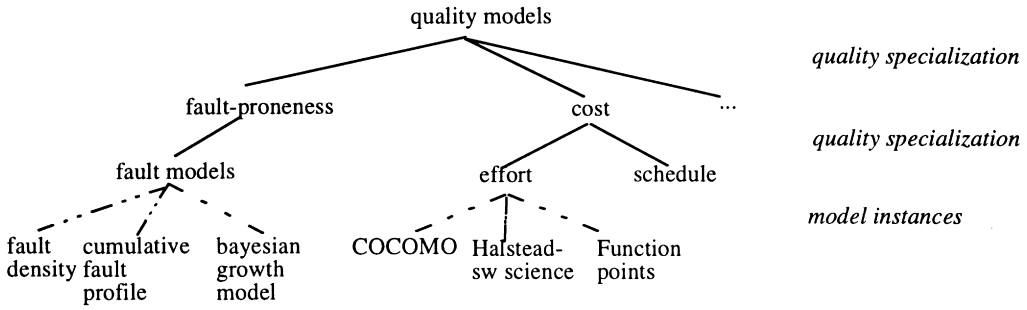


Fig. 11. Taxonomy of quality models—example.

decisions have to be made. The taxonomy directs the decision by providing decision criteria at each level, e.g. the level of granularity at which information is gathered, and possible alternatives, e.g. regarding the criteria treatment of inheritance, the distinction between inheritance-based coupling and non-inheritanced-based coupling. Traversing the taxonomy, finally, specific measure candidates are indicated. Pointers to relevant measures can be provided for the quality/resource models which have also been used in the past by providing a catalog of measures providing their definitions, properties and underlying assumptions (see Fig. 15). To ensure the adequate reuse of measures, the assumptions about the contexts in which they can be applied and their properties must be explicitly stated. This facilitates the adequate selection of measures and their eventual adaptation to characteristics of the present environment. Various types of lessons learned regarding the measures can be provided:

- Relationships (casual or not) to other measures, e.g. a measure on complexity on measures on size.
- Sensitivity analysis to phenomena such as reuse, programming language features, etc.
- Typical range/distribution of values. For example, for a specific type of software project fault density varies between 2 faults/KSLOC to 5 faults/KSLOC.

4.2.6. Knowledge about data-collection procedures

Various facets of data-collection procedures can be captured in a taxonomy (see Fig. 16) which provides guidance for the specification of the data-collection procedure through the explicit representation of alternative decisions. Decision guidelines may be associated with each branch. A list of alternative procedures for each measure and their usage prerequisites can support the definition of data-collection procedures. An example for a measurement plan, as a specification for the data-collection procedures, is provided in Table 3. The reuse of knowledge about the data-collection procedures from past measurement programs in similar environments is expected to reduce the effort for the development of the procedures and should offer reuse opportunities in terms of collection instruments. Furthermore, by reusing data-collection procedures which have been successful in the past, the adequacy of the procedures to the process and organization can improve and be refined over time. Lessons learned about the collection procedures can improve the reliability and validity of data collection:

- Interdependences of data-collection procedures to measures, e.g. by describing which measures can be collected automatically by which tools.
- Interdependences of data-collection procedures to software process, products or resources. For example, in order to keep overheads minimal and avoid resistance, fault data has to be collected as an integrated part of change documentation.
- Analysis of reliability and validity of data-collection procedures, e.g. collecting development effort data weekly instead of monthly increases its validity, without excessive augmentation of data-collection effort.

4.2.7. Knowledge about data-collection instruments

In general, support for those different instruments, e.g. tools or questionnaires, can be provided in several forms, e.g. through usage prerequisites, guidelines, or the instruments themselves. An explicit overview on existing and available tools in a particular environment, and their application preconditions guides the selection of a specific tool based on an explicit rationale. The application of the tools can be supported by knowledge on past problem occurrences, e.g. certain tools cannot interface and exchange data and how this was resolved. The development of questionnaires for the collection of subjective data, e.g.

cohesion: The manner and degree to which the tasks performed by a single software module are related to each one another. Types include coincidental, communical, functional, logical, procedural, sequential, and temporal.
complexity: The degree to which a system or component has a design or implementation that is difficult to understand and verify.
maintainability: An attribute that bears on the effort needed for specific modifications.

Fig. 12. Excerpt of glossary of attributes [11]—example.

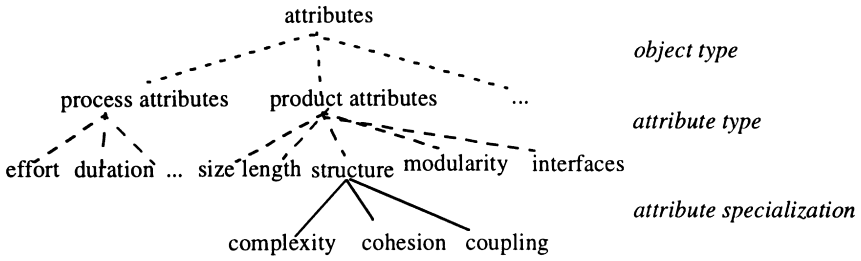


Fig. 13. Taxonomy of attributes—example.

subjective product measures and process measures, can be guided by a taxonomy of questionnaires used in past measurement programs (see Table 7). Questionnaires from past measurement programs can be retrieved and reused. Lessons learned with respect to the use of these questionnaires can help an organization improve the reliability and validity of the data collected:

- Analysis of validity of data-collection instruments, e.g. stating a high variation between answers of a questionnaire owing to missing explanations of the answer categories.
- Analysis of application of instruments for data collection, e.g. if questionnaires were not answered owing to the lack of a submitting process.

Reusing knowledge about tools and questionnaires from past measurement programs is expected to reduce the effort of designing appropriate collection instruments for supporting data-collection procedures, the development of the instruments, and their tailoring to organization-specific needs.

4.2.8. Knowledge about knowledge-acquisition instruments

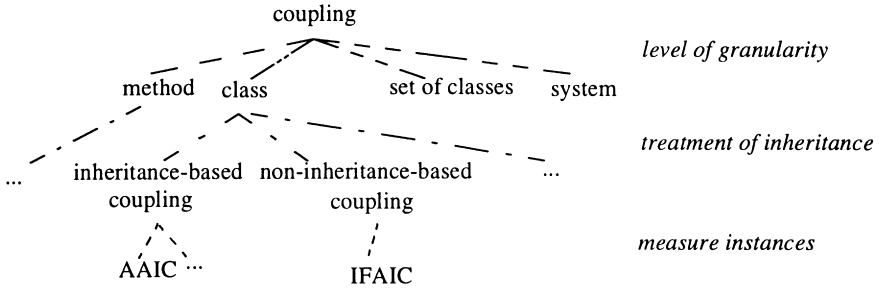
Abstraction sheets developed in past measurement programs can be (partly) reused in a measurement program with similar goals and context, improving the quality and appropriateness of the output of the interviews. They can provide an overview on relevant aspects regarding certain measurement goals. Knowledge stated in the quality focus can be used in order to indicate potentially relevant quality dimensions. The respective baselines hypotheses can be reused as a checklist. Variation factors indicate factors which might have an impact on the quality dimensions in the current measurement program. The stated impact on the baselines hypotheses can be used in order to direct the reuse of these variation factors according to which quality dimensions are influenced. Lessons learned on the use abstraction sheets can point out potential problems in future interviews:

- Relationships of the entries in the abstraction sheet to the goal, e.g. if a particular quality model is addressed in all individual abstraction sheets on one measurement program, the importance of the model with respect to the goal is emphasized.
- Relationships of the entries to the questions/models/measures, e.g. concerning difficulties, while developing a model based on an abstraction sheet entry on 'tool support (low, high, medium)' due to the lack of a definition of the classification categories.

Interview plans describe the steps for performing interviews, their supporting material, and their usage, e.g. distribution of reading material in advance (see Fig. 17). They can be reused in order to prepare interviews in the current measurement program, reducing the effort of the preparation of the interview plan and the tailoring of the plan to the organization-specific needs.

Besides (re-)using knowledge regarding the products of the GQM process, knowledge about the performance of the process itself can provide additional support.

A description of process/activities includes a description of the activity, consumed/produced products in the process/activity, roles involved, and entry/exit criteria. Such a



...

Fig. 14. Taxonomy of measures [2]—example.

Import coupling of a software part (IC)
Assumption: The more dependent a software part on external data declarations, the more external information needs to be known in order to make the software part consistent with the rest of the system.
Definition: Import coupling is the extent to which a software part depends on imported external data declarations.
Property: Monotonicity: Let *m1* be a module and *II(m1)*, its set of important interactions. If *m2* is a modified version of m1 with the same sets of data and subroutine declarations and one more import interaction so that *II(m2)* includes *II(m1)*, then *import_coupling(m2)* ≥ *import_coupling(m1)*.
Metric: Given a software part *sp*, Import Coupling of sp (denoted by IC(*sp*)) is the number of interactions between data declarations external to *sp* and the data declarations within *sp*.

Fig. 15. Measure—example [3].

process description, defined based on experiences from past measurement programs, can guide the performance of the GQM process (see Fig. 18 for a short example). By creating organization-specific GQM process models, subsequent tailoring efforts are reduced. Reusing the tailored GQM process descriptions allows the iterative improvement of the process to the organization-specific characteristics from one project to another. Descriptions of the GQM products outline the structure and expected content of a product and can guide its development. For example, the performance of interviews can be guided by pointing out the issues to be addressed during the interview according to the structure of an abstraction sheet (Fig. 1). Based on experiences gained from the establishment of GQM-based measurement programs in past projects, guidelines and heuristics about the application of the GQM approach can be derived (see Fig. 19). They can be stated as lists of Do's and Dont's. If possible, if-then-else rules can be derived, referring to relevant context factors for the application of the guidelines. They provide additional guidance for the application of the GQM approach. Reusing knowledge on potential problems can help to prevent the repetition of mistakes made in the past and provide adequate solution strategies for emerging problems. This knowledge can be captured by describing the problem, proven solutions, and providing an assessment of the solutions. For example, because project manager and software developers were interested in different measurement goals, no common measurement goals could be defined. As a solution, at least one goal from each perspective was selected and, consequently, all involved people felt concerned by the measurement program and supported it. Cost/effort information, specifying the typical cost and effort related to the application of the GQM process in the past, can support the planning of a measurement program with respect to resource allocation and scheduling. Examples of effort data are [8]:

- The total effort needed for introducing GQM-based measurement is about 1 person year.
- The relation of the effort of the planning and execution phase is about 2/3 (planning) to 1/3 (execution).

4.3. Levels of abstraction

The reusable knowledge as described in the previous section can be provided on different levels of abstraction:

- *Project-specific instances of GQM entities*: On the lowest level, project-specific instances of the GQM entities gathered from one particular software project are represented, e.g. instances of GQM processes customized to a specific software project, or instances of GQM products, which have been produced during the performance of the measurement program. The knowledge represented on this level of abstraction, reflecting only experiences gained in one specific software project, has to be packaged and integrated into the existing organizational experiences for effective reuse on future projects. Therefore, from this project-specific knowledge, more general entities have to be developed,

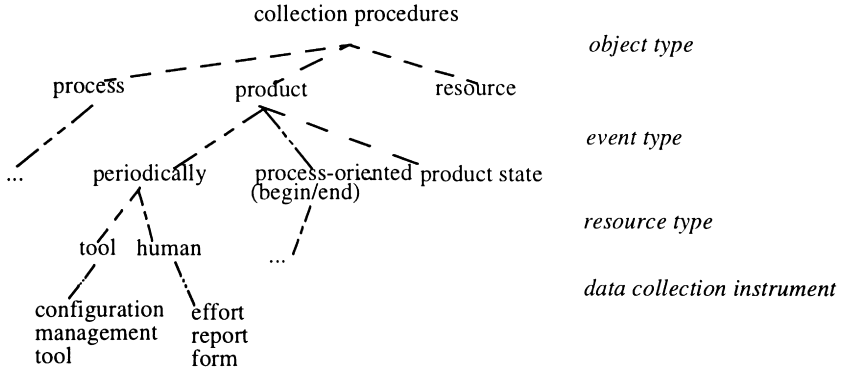


Fig. 16. Taxonomy of data collection procedures—example.

Table 7
List of questionnaires—example [12]

Category	Name	Description
Project Forms	*Project Start-up Form*	Records general project information collected at the project start-up meeting.
	Project Estimates Form	Records the completion estimates for project parameters; is filled out by project managers.
Effort Forms	*Personnel Resources Form*	Provides information on hours spent on a project and how the effort was distributed; is filled out weekly by software developers or maintainers.
Maintenance Forms	*Maintenance Change Report Form*	Characterizes the maintenance performed in response to a change request.
	Change Report Form	Records information on changed units; is filled out each time a configured unit is modified.

subsuming experiences across individual software projects.

- *Generic templates of GQM entities*: The project-specific instances are integrated and represented on a more abstract level in order to be applicable across various software projects with similar characteristics. The integration of project-specific instances also has to be done in order to synthesize all knowledge in a given environment. The resulting generic instances of GQM entities are reused in future software projects with similar characteristics by providing templates for the GQM entities. According to project-specific parameters, these templates may have to be instantiated, tailored and adapted to the actual application project characteristics.
- *General structure of GQM entities*: On a higher level, general GQM entities can be formulated in order to provide a general framework for the application of processes, and the development of products for measurement programs across environments.

5. Representation and packaging of knowledge

In the last section, we identify the knowledge which can be reused and support the planning of measurement programs. In this section we examine how the knowledge gained on individual projects has to be packaged and represented.

In order to allow intelligent and efficient reuse, additional knowledge has to be captured. This additional knowledge will be used to identify adequate experiences with respect to the characteristics of a new project. It can be classified into the following categories:

- Environment information describing the context from which the knowledge originate
- Basic information explaining the acquisition and representation of the knowledge
- Interdependences between individual GQM entities
- Information regarding the reuse of the specific GQM entity in the past

5.1. Environment information

For correct retrieval of the knowledge from the knowledge base, an appropriate and unambiguous characterization of the environment from which the GQM entity has been obtained is essential. Characterizing the environment is important for relating a GQM entity to the appropriate context from which it has been derived. The entities have to be classified with respect to a variety of characteristics. This allows the clustering of projects with similar characteristics and goals. According to these clusters, generalized templates for GQM products can be packaged in order to increase the effectiveness of reuse. Such a description of the environment includes, for example, the characterization of the following aspects:

- *Organizational environment*, e.g. business sector, standard process models, organizational structure;
- *Business and improvement goals of the organization*,

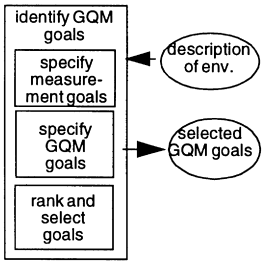
Interview Plan

1.Motivate the establishment of GQM-based measurement
2. The interviewees have to be informed about the purpose of the interviews and the usage of the acquired information.
3. Give confidentiality guarantees
4. Perform the interview
4.1 Explain the GQM goal
4.2 Ask the interviewee to specify the quality focus
4.3 Ask for baseline hypotheses for each quality dimension
4.4 Ask which variation factors may have an impact on the quality dimensions
4.5 Ask for each variation factor, on which quality dimensions it will have an impact.
5. Finish the interview by pointing out whether further interviews are necessary and when feedback will be provided.

Fig. 17. Simple interview plan—example.

Identification of GQM Goals

Based on the environment information collected in the previous phase, GQM goals are derived from the high-level business goals, the organization's strategic goals and mission, or more directly from the organizational improvement goals regarding major concerns/problems and the project goals.The different goals are formulated as GQM goals in terms of: object of study, purpose, quality focus, viewpoint and environment.In order to concentrate on the most important goals, the identified GQM goals are prioritized and the most important ones wrt. the needs of the organization and the specific project are selected.



entry criteria:The *description of environment* is updated and the GQM goals of the measurement program are not yet defined or have to be changed.
exit criteria:The *goal documentation* is complete.
roles involved: GQM experts, project team, project manager, senior manager

Fig. 18. GQM process description—short example.

e.g. to reduce effort by 30% in the next 3 years, or to dominate a specific market niche;
- *Software project including*, e.g. used process models, tools, planned effort and duration;
- *Software project-specific goals*, e.g. to complete the project within the planned schedule;
- *Measurement program*, e.g. restrictions with respect to duration, availability of expertise and tool support.

The environment characteristics have to be analysed with respect to their impact on the experiences gathered in the measurement programs, aiming at the identification of a minimal set of relevant context factors. Reusable experiences are captured in relation to these context characteristics in order to provide adequate support.

5.2. Basic information

In order to support the appropriate usage of the available experiences, basic knowledge on the entities has to be provided. This includes the following information about:

- *Purpose of capturing the entity*, e.g. a GQM goal template is captured in order to provide guidance for the comprehensive definition of the measurement goal by pointing out the relevant dimensions;
- *Viewpoint stating the role from whom the knowledge was acquired*, e.g. manager, developer;
- *Representation form in which the entity is captured in the knowledge base*, e.g. textual, graphical, rule;
- *Acquisition technique* specifying how the entity was derived, e.g. interview, statistical analysis;
- *Representativeness* in terms of the number of individual software projects from which the entity was derived;
- *Administrative information*, such as access rights, ownership.

The purpose of this information is to determine whether the respective GQM entity is relevant in a given context before being evaluated for reuse. For example, when a GQM entity was acquired from a specific role, e.g. a measurement expert, and is planned to be reused by a project manager, its reuse could cause misunderstandings due to their different viewpoints and understandings. Providing basic information will explicitly indicate these problems and prevent the reuse of inadequate knowledge for particular reuse needs.

5.3. Interdependences

The GQM entities to be reused are related by a complex net of interdependences. These interdependences have to be explicitly modeled to facilitate the comprehensive reuse of GQM entities. Structural interdependences specify interconnections among lower-level and higher-level entities, thus supporting the management of complexity via abstractions. The following structural interdependences have been identified.

Generalization: It denotes a relationship (*is_a*) between an object type and one or more refined or specialized versions of it. The *is_a* relationship for GQM products is represented by the taxonomy of classes of GQM products in Section 4 and for QM processes in Fig. 20.

Instance-of: The *instance-of* relation is a special case of

Guidelines: Identification of GQM goals

> All roles affected by the measurement program should participate in the goal-identification, e.g., management representatives, project manager, project team members, and the GQM team.

> The GQM goals should be selected rigorously: concentrate on only the most central issues and a few important goals.

> If GQM is introduced into the organization, then focus on building a basic understanding and the setting of quantitative baselines.

Fig. 19. Guidelines [6]—example.

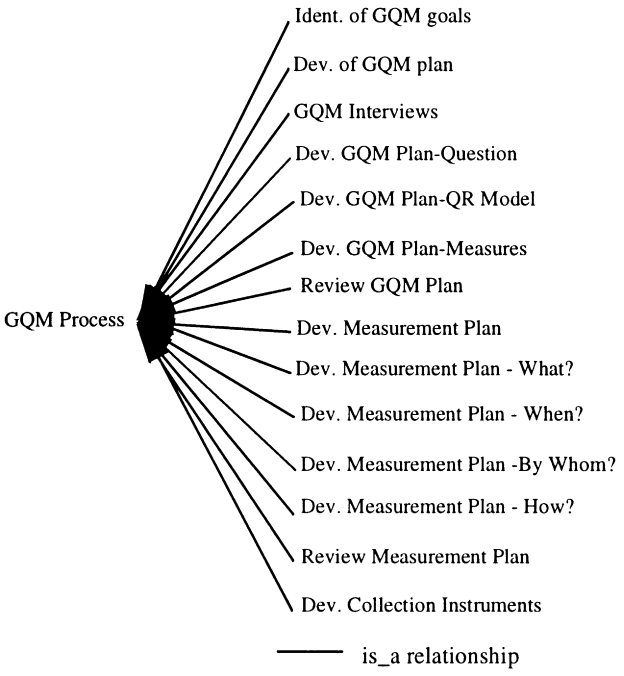


Fig. 20. Generalization.

generalization. This interconnection is established between a concrete object (an instance) and its type. The type of objects is defined as a precise characterization of properties with respect to structure or behavior, which are common for a certain set of entities (*classes of GQM entities*). An instance of a class of GQM entities is a concrete project-specific instantiation of this entity, e.g. the GQM plan of the measurement program MP-2 in the software project TRANS-X in the company ABC.

Aggregation: This is used to combine objects to form a higher-level aggregate object. The individual constituents of an aggregate constitute components of the representation. In this role, they are parts of an object's representation (*part_of*). The aggregation relation is illustrated in Fig. 21.

Defines-relationship: In the context of GQM-based measurement program a specific interdependence between GQM products has been identified: the *defines* (or conversely, *is _defined_by*) relationship, see Fig. 22. This relationship describes the fact that a GQM product (e.g. a question in the GQM plan) is defined based on another GQM product (e.g. a quality dimension of the abstraction sheet). The explicit modeling of this interdependence guarantees the traceability between the individual GQM products.

5.4. Information regarding the reuse of GQM entities

In order to improve continuously the reuse of entities and evolve a methodology for the systematic reuse of GQM entities, information is captured about their reuse instances. This information is the basis for organizational learning regarding the reuse of GQM entities. It includes information about:

- *Preconditions for reuse*: description of necessary preconditions for the reuse of the entity, e.g. for the reuse of a GQM plan, the measurement has to be implemented in the context of a well-defined and planned organizational improvement program;
- *Expected adaptations*: description of adaptations done in the past when reusing the entity and the relevant factors which motivated the adaptations;

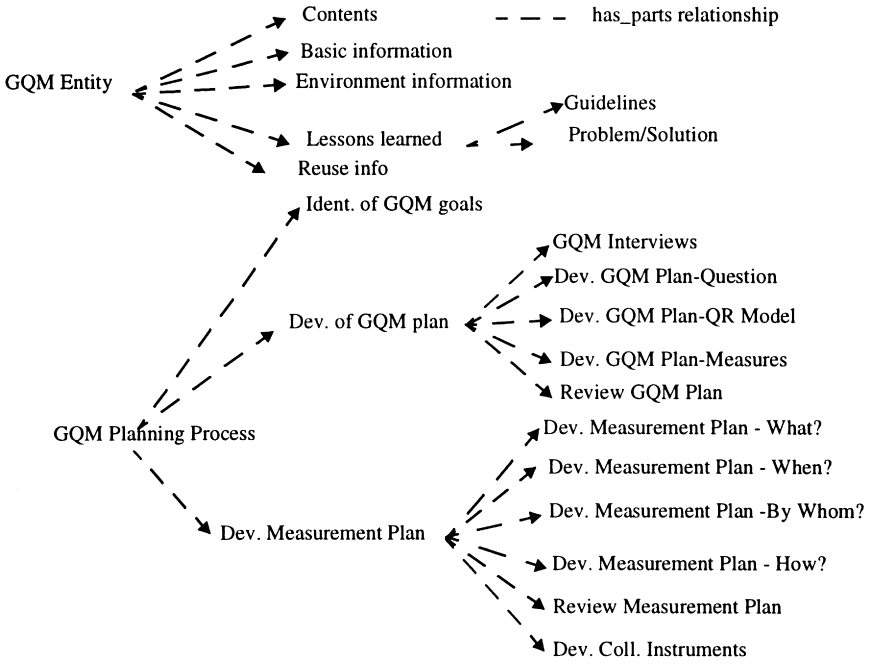


Fig. 21. Aggregation.

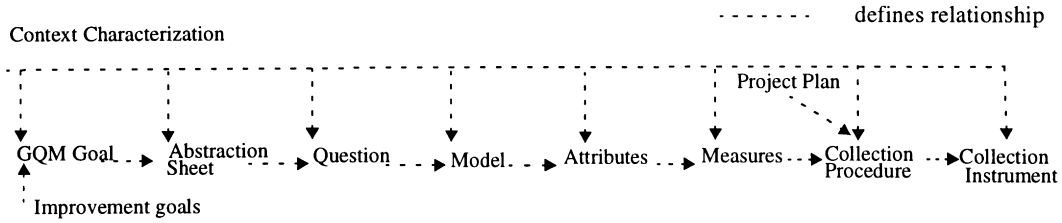


Fig. 22. Defines-relation.

- *Expected cost of reuse*: description of the expected cost of reusing the entity as a basis to decide whether the entity should be reused or rather be developed from scratch;
- *Dates of reuse*: dates of reuse in order to provide an overview on when and how often this entity was reused;
- *Guidelines of reuse*: how to reuse the entity, e.g. crucial quality aspects in an organization, as stated in the context characterization, can indicate required adaptation of the quality focus of the GQM goal.
- *Problem/solution statements*: problems which arose in the past reusing this entity, associated with applied solutions. For example, if measures reused in a project in which a different software process model was used, turned out to be inadequate and were carefully adapted to the particular software process in place.

6. Reuse scenarios

In Section 4, we described which kind of knowledge can be reused in order to support the GQM planning process. In order to illustrate the potential reuse of this knowledge, we provide some scenarios focusing on particular steps of the planning of a measurement program.

6.1. Scenario 1: Definition of GQM goals

As specified in Section 3.1, the definition of the GQM goals is performed in terms of object, purpose, quality, focus, viewpoint and context of measurement. Based on knowledge captured from past measurement programs, support for the definition of each of these dimensions is provided. For example, the identification of the object to be studied in the measurement program can be facilitated by a taxonomy of potential objects with respect to the specific environment. Assuming we focus on the study of development processes, the decision of the object of study is guided by traversing the process taxonomy (see Fig. 6), providing relevant decision criteria, e.g. scope of the measurement program (e.g. requirements phase, inspection activity), application domain (e.g. embedded real-time systems) and indicating possible alternatives at each level of specialization. During the discussion about the definition of the measurement goal, in which different roles are involved (e.g. developers, senior management and the quality assurance team), the communication is facilitated by a glossary of the organization-specific terminology regarding potential objects of study providing a consistent definition of these objects. Reuse of lessons learned can warn for potential problems and support the effective definition of the GQM goal with respect to specific project characteristics. For example, if the expected duration of the software project is 10 years and the complete software development process has been selected as the object of study, a lesson learned indicating the refinement the object of study focusing on a specific phase could warn for a failure due to a long measurement period until actually achieving benefits. The definition of the GQM goal is completed in a similar manner for each of the other dimensions.

6.2. Scenario 2: Definition of quality/resource models

During the development of the GQM plan, quality/resource models have to be defined to answer questions. Abstract concepts as described in the questions can be refined into operational quality/resource models using a taxonomy on quality/resource models. For example, if the questions focuses on the distribution of defects, a taxonomy can guide the derivation of the appropriate quality model by selecting the appropriate defect definition (e.g. fault, failure) and type of distribution (e.g. per phase of detection, per detection mechanism). Based on the environment characteristics and the measurement goal, appropriate model definitions can be identified from past measurement programs. These models are suggested as potential candidates in the current project. Before reusing the models, it has to be verified whether their underlying assumptions are still valid. If necessary, the models have to be adapted to specific environment characteristics. For example, assuming that inspectors capabilities vary extensively, the effectiveness of inspections is expected to depend not only on the size of the inspected document, but also on the training of inspectors. If not already considered, this new factor has to be included in the reused model. The definition of the models is further supported through lessons learned particularly related to this model, e.g. the effectiveness model is only applicable when the inspected document is complete, i.e. contains all specification, design and code documentation.

6.3. Scenario 3: Definition of data-collection procedures

The development of the data-collection procedures can be supported by providing knowledge on this activity by describing what has to be done and how. For example, a process description on what are the objectives (e.g. the integration of measurement into the software process), and what has to be done (e.g. determination of when, by whom, and how the data has to be collected), supports process performance. By evolving the measurement process based on experiences from past measurement programs, the process description reflects explicitly the knowledge gathered on the process performance in a specific environment. Providing knowledge on the products consumed (e.g. GQM plan and software project plan) and produced (e.g. measurement plan), further facilitates the execution of the activity, especially if dependences are shown explicitly (e.g. between the activities in the software process and points of time for data collection). The development of the resulting products (e.g. measurement plan) is further supported by additional knowledge on the product, how it is structured, and what content is required. This also ensures a consistent form of these products across projects. If further assistance is required, guidelines on how to perform the process can be suggested, e.g. the selection of the data collector depends on who has the expertise, who has access to the object being measured, etc. Explicit available guidelines and heuristics based on experiences gained in past measurement programs anticipates most probable problems in the specific environment. If the determination of an adequate collection strategy (e.g. level of granularity, frequency) is questionable, an overview of similar problems from past measurement programs regarding this process step can be provided and their corresponding solution strategies can be used as suggestions for the current problems.

7. Conclusions

For continuous learning in a software organization, the software development know-how available from past projects has to be packaged and reused in new projects. In this paper, we focus on gained software measurement know-how. We show that there is a great potential for reducing the effort related to the measurement planning phase and for improving the adequacy, consistency and effectiveness of a measurement plan through reuse. Within the context of the GQM paradigm for software measurement, we describe the type of knowledge and GQM entities which can be reused, what reuse strategies seem adequate and what knowledge-base support is needed to enable an effective and efficient planning for measurement programs. Examples of reuse scenarios, showing how reuse can be integrated into the planning process, have also been provided. Future research includes the development of specific knowledge-based techniques for the retrieval and tailoring of experiences from past measurement programs, the acquisition of new experiences, and the packaging of these experiences into a knowledge base.

References

[1] L.C. Briand, C.M. Differding, H.D. Rombach, Practical Guidelines for Measurement-Based Process Improvement. Software Process Improvement and Practice, 1997.

[2] L.C. Briand, J.W. Daly, J. Wüst, A unified framework for coupling measurement in object-orientated systems. Technical Report ISERN-96-14, Fraunhofer Institute IESE, Germany, 1996.

[3] L.C. Briand, S. Morasca, V.R. Basili, Defining and validating high-level design metrics. Technical Report CS-TR-3301, Department of Computer Science, University of Maryland, June 1994.

[4] L. Briand, W. Melo, C. Seaman, V. Basli, Characterising and assessing a large-scale software maintenance organisation. Proceedings of ICSE, 1995.

[5] V.R. Basili, H. Dieter Rombach, The TAME project: towards improvement-orientated software environments, IEEE Transactions on Software Engineering SE-14 (6) (1988) 758–773.

[6] The CEMP Consortium, Customised establishment of measurement programs. Final Report, ESSI Project No. 10358, Germany, 1996.

[7] C. Gresse, B. Hoisl, J. Wüst, A process model for GQM-based measurement. Technical Report STTI-95-04-E, Software Technology Transfer Initiative, University of Kaiserslautern, Department of Computer Science, D-67653 Kaiserslautern, Germany, October 1995.

[8] C. Gresse, B. Hoisl, H.D. Rombach, G. Ruhe, Kosten-Nutzen-Analyse von GQM-basiertem Messen und Bewerten: Eine replizierte Fallstudie. Wirtschaftinformatik, Springer, 1997.

[9] H. Günther, H.D. Rombach, G. Ruhe, Kontinuierliche Qualitätsverbesserung in der Software Entwicklung — Erfahrungen bei der Allianz Lebensversicherungs-AG (in German). Wirtschaftsinformatik 38, 1994.

[10] IEEE Standard Glossary of Software Engineering Terminology (IEEE Std 610.12-1990). Institute of Electrical and Electronics Engineers, Inc., USA, 1990.

[11] IEEE Standard for a Software Quality Metrics Methodology (IEEE Std 1061-1992). Institute of Electrical and Electronics Engineers, Inc., USA, 1992.

[12] National Aeronautics and Space Administration, Software Measurement Guidebook. Technical Report SEL-84-101, NASA Goddard Space Flight Center, Greenbelt MD 20771, July 1994.

[13] R. Prieto-Diaz, P. Freeman. Classifying Software for Reusability. IEEE Software, January 1987.