Interner Bericht

Technology Package for the
Goal Question Metric Paradigm

C. Differding, B. Hoisl, C. M. Lott

281 / 96

Fachbereich Informatik

Universität Kaiserslautern • Postfach 3049 • D-67653 Kaiserslautern

Technology Package for the Goal Question Metric Paradigm

Christiane Differding, Barbara Hoisl
AG Software Engineering
Department of Computer Science
University of Kaiserslautern
67653 Kaiserslautern, Germany
differdi@informatik.uni-kl.de

Christopher M. Lott
Fraunhofer Institute for
Experimental Software Engineering
Sauerwiesen 6
67661 Kaiserslautern, Germany
lott@iese.fhg.de

Internal Report 281/96

Abstract

This document offers a concise introduction to the Goal Question Metric Paradigm (GQM Paradigm), and surveys research on applying and extending the GQM Paradigm. We describe the GQM Paradigm in terms of its basic principles, techniques for structuring GQM-related documents, and methods for performing tasks of planning and implementing a measurement program based on GQM. We also survey prototype software tools that support applying the GQM Paradigm in various ways. An annotated bibliography lists sources that document experience gained while using the GQM Paradigm and offer in-depth information about the GQM Paradigm.

Herausgeber: AG Software Engineering
Leiter: Prof. Dr. H. Dieter Rombach

Kaiserslautern, April 1996

Contents

1 Introduction

The Goal Question Metric Paradigm (GQM Paradigm) was developed in response to the need for a goal-oriented approach that would support the measurement of processes and products in the software engineering domain. The GQM Paradigm (sometimes called the GQM approach) supports a top-down approach to defining the goals behind measuring software processes and products, and using these goals to decide precisely what to measure (choosing metrics). The GQM Paradigm additionally supports a bottom-up approach to interpreting data based on the previously defined goals and questions. If viewed narrowly, the GQM Paradigm may be seen as purely an approach for choosing metrics. However, we encourage a broader view of the GQM Paradigm as a means for defining the "measurement view" of a software project. In other words, the analysis and evaluation of processes and products from all activities and phases of a software engineering project may be planned and performed with help of the GQM Paradigm.

The GQM Paradigm was first developed in 1984 at the University of Maryland and extended as part of the TAME project (see [BW84, BR88]). Research into using and improving the GQM Paradigm has also been in progress at the University of Kaiserslautern since 1992 (see [Dif93, GHW95, vM95]) and at the Fraunhofer Institute for Experimental Software Engineering since 1996. Over the years the GQM Paradigm has been applied by software engineering researchers and practitioners in many different contexts with good success. However, nearly every user of GQM has tailored the paradigm to suit his or her specific needs, resulting in many different views. Although unifying the many different views is impossible, this document attempts to present a view of GQM that is consistent with the views of the software engineering research groups at the University of Maryland and the University of Kaiserslautern.

We state the motivation for measurement activities in Section 2. Then we describe the GQM Paradigm in terms of its basic principles (Section 3), techniques for structuring GQM-related documents (Section 4), and methods for performing tasks of planning and implementing a measurement program based on the GQM Paradigm (Section 5). We also survey prototype software tools that support applying the GQM Paradigm in various ways in Section 6. A glossary of terms is defined in Appendix A. The report ends with an annotated bibliography of sources that document experience gained while using the GQM Paradigm and offer in-depth information about the GQM Paradigm.

2 Motivation for Measurement

It is generally accepted that measurement is not an end in itself but a means to an end. The final objective must be *improvement* of products and processes. Measurement should be viewed as an infrastructure technology that is necessary to achieve systematic improvement [BCR94c]. The relationship between measurement and systematic improvement can be summarized using the following points:

- Knowing the current state of affairs in a software engineering project is necessary for identifying the strengths and weaknesses of the processes currently in use. As stated by Humphrey, "if you don't know where you are, a map won't help" [Hum89, p. vii].

- Measurement is necessary to characterize the current state of affairs quantitatively; i.e., to derive a quantitative baseline. A "quantitative baseline" is nothing other than a model that captures some concrete information about the status quo. For example, the statement "90% of all faults in a design document are detected by project XYZ's design inspections" is a quantitative baseline.

- Once the strengths and weaknesses of currently used processes have been identified and described quantitatively, changes that might improve the process can be selected, performed, and evaluated through measurement. These changes can first be applied to a project when a comparison between actual and target values is possible. Further, the impact of changes to the process can only be determined quantitatively if a quantitative baseline is available against which comparisons can be made. Otherwise, it is impossible to determine the extent of a change, not to mention whether the change had a positive or negative impact.

The development of the GQM Paradigm for goal-oriented measurement also led to the development of the Quality Improvement Paradigm (QIP). The QIP essentially is the application of the scientific method tailored to the needs of software engineering. The improvement process due to the Quality Improvement Paradigm is articulated in [BCR94a]. Because information necessary for applying the GQM Paradigm is derived and/or used in every step of the QIP, the GQM Paradigm has also been described as the measurement view of the QIP.

3 Principles behind GQM

Definition. A "principle" as used here is a fundamental idea or doctrine. The idea behind these principles is that each must hold; i.e., if one does not, then the program is not in conformance with the dictates of the GQM Paradigm.

The GQM Paradigm is based on the idea that measurement should be goal-oriented; i.e., all data collection should be based on a rationale that is explicitly documented. This approach offers several advantages. First, it helps in the identification of useful and relevant metrics. Second, the goals provide a context for the analysis and interpretation of collected data. Third, an explicit rationale explaining the refinement of goals into metrics enables an assessment of the validity of the conclusions that were drawn. Finally, because the software development personnel helped define the rationale for data collection, and know that the data will be used for their own purposes, they offer less resistance against a measurement program than they would if they feared the data might be used against them. To yield these advantages, GQM-based measurement programs should be planned and performed according to the following principles.

1. The analysis task to be performed must be specified precisely and explicitly using a detailed measurement goal.

2. Metrics must be derived in a top-down fashion based on goals and questions. A structure of goals and questions *may not* be retrofitted onto an existing set of metrics.

3. Each metric must have a underlying rationale that is explicitly documented. This rationale is embodied in the series of questions via which a metric is derived from a goal. The rationale is used for justifying data collection and for guiding data analysis and interpretation.

4. The data that are gathered for the metrics must be interpreted in a bottom-up fashion using the GQM goal and questions. This supports interpreting the data subject to the limitations and assumptions behind the rationale for each metric.

5. The people from whose viewpoint (i.e., perspective) the measurement goal is formulated must be deeply involved in the definition and interpretation of the measurement goal. Not only will they supply the data, they are also the real experts with respect to the analysis and interpretation tasks.

4 Techniques for structuring GQM-related products

Definition. A "technique" as used here refers to some way of structuring a product required or used by the GQM Paradigm. The focus is on developing and structuring specific products.

As defined in the appendix, a GQM plan consists of a single goal plus the sets of questions and metrics needed to provide an operational definition of that goal. Three techniques are introduced for developing GQM plans. First, goal templates are shown that assist in generating a GQM goal. Second, abstraction sheets (a type of form) are defined to assist in collecting the information necessary to build a detailed GQM plan. Third and finally, two structures are given that document both product- and process-related information in GQM plans. A product that is closely related to a GQM plan is a measurement plan; the contents of such a plan are also discussed.

4.1 Goal templates

The process of setting goals and refining them into quantifiable questions is complex and requires experience. The following template for a GQM goal has been developed to indicate the required contents of a GQM goal and thereby to support the goal-setting activity. Product and process goals are handled differently; examples of both types of goals are given following the template. See also [Rom91, GHW95].

The template identifies five major aspects, namely the object, purpose, quality focus, viewpoint, and environment of a measurement program. First, the object aspect expresses the primary target of the study; i.e., the process or product that will be analyzed. Second, the purpose aspect expresses how the object will be analyzed; i.e., will it be analyzed for purposes of understanding and characterization, will it be compared with some other object, etc. Third, the quality focus aspect expresses the particular property of the object that will be analyzed in the course of the study, such as cost, reliability, etc. Fourth, the viewpoint aspect expresses information about the group of people that will see and interpret the data. By stating clearly the group to which the analyzed data will be released, issues of confidentiality can be addressed before any data are collected. Finally, the environment aspect expresses the context in which the study will be performed, and is used to make influencing factors explicit.

In previous work, the aspects 'object' and 'purpose' are sometimes grouped under the heading "purpose of study," the aspects 'quality focus' and 'viewpoint' are grouped under the heading "perspective of study," and the final aspect 'environment' is labeled the "context of study."

1. Analyze some …

   object: (one of) processes, products, other experience models, …

2. for the purpose of …

   purpose: (one of) characterization, assessment, prediction, evaluation, control, improvement.
   (Recent work at the University of Maryland restricts the "purpose" keyword to the prior six choices; prior research made no such restriction, and included words such as motivation, engineering, certification, ….)

3. with respect to: …

   quality focus: (one of) cost, correctness, defect removal, changes, reliability, user friendliness, maintainability, …

4. from the point of view of the …

   viewpoint: (one of) user, customer, manager, developer, researcher, corporation, …

5. in the following environment: …

   environment: (one or more of) problem, people, resources, processes, organization, project, …

An example product goal constructed using this template might be:

Analyze the **final product**	(object)
for the purpose of **characterization**	(purpose)
with respect to **reliability**	(quality focus)
from the point of view of the **tester**	(viewpoint)
in the context of **Project X**	(environment).

An example process goal constructed using this template might be:

Analyze the **testing process**	(object)
for the purpose of **improvement**	(purpose)
with respect to **reliability**	(quality focus)
from the point of view of the **developer**	(viewpoint)
in the context of **Project Y**	(environment).

4.2 GQM abstraction sheet

A GQM abstraction sheet is a document, often a single sheet of paper, that helps elicit and structure information during an interview and assists in constructing, refining, and reviewing a single GQM plan. The design of the abstraction sheet reflects the issue that people may skip from one issue to another during an interview. An abstraction sheet aids in coping with this problem and it provides a reminder as to which generic categories issues must be addressed. An abstraction sheet can also be used as an abstract view of a GQM plan that helps to reveal the dependencies between the questions of that GQM plan. The suggested layout for the components of a GQM abstraction sheet is shown in Figure 1; see also [GHW95]. An abstraction sheet consists of four quadrants plus a section labeled "Feedback," as described next.

Quality Focus: Information that defines the quality focus is collected in this quadrant. This information is intended to capture one person's definition (i.e., model) of the quality focus as well as the subtleties that affect their definition. The information gathered here will be used to construct the quality model in the GQM plan.

For example, if the quality focus is "effort," the information in this quadrant must document what "effort" means to the interviewed persons and what additional details are relevant with for defining effort such as dividing the effort according to the phase of development or specific development activity. However, *no* specific effort data would be entered here.

Baseline Hypothesis: This quadrant documents concrete data concerning the quality focus, considered from the point of view of the people who are interviewed. In other words, the statements recorded here attempt to capture the people's thinking about the project's state of affairs at the beginning of the measurement program. If no concrete data exist, information in this quadrant represents a hypothesized baseline. However, if concrete data do exist, this quadrant holds information that represents an actual baseline for the quality focus of interest. The information gathered here is used to validate the quality model or to state target values.

Object	**Purpose**	**Quality focus**	**Viewpoint**	**Context**
(object)	(purpose)	(focus)	(viewpoint)	(context)

Quality Focus: *Describe the quality focus* (Quadrant 1)	**Variation Factors:** *Which factors have an impact on the quality focus?* (Quadrant 2)
Baseline Hypothesis: *What do you believe/know is the current state with respect to the quality focus?* (Quadrant 3)	**Impact on Baseline Hypothesis:** *How do the variation factors influence the baseline hypothesis?* (Quadrant 4)

Feeedback:

Figure 1: Components of a GQM abstraction sheet

If people are not able to give a baseline hypothesis (based on data or estimated), then this suggests that the selected quality model is either not suitable for this purpose, or is entirely fictitious.

Continuing with the example of a quality focus on "effort," the baseline hypothesis quadrant will include concrete data about the distribution of effort among activities (e.g., "20% of total effort is spent in the coding phase"). People must state either their assumptions or the relevant data about the current distribution. Specific effort data would be entered in this quadrant.

Variation Factors: The environmental factors and project factors that may have an impact on the quality focus are stated in this quadrant.

Returning to the example, people might state that the distribution of effort per phase depends on how well the developers understand the application domain. They would enter this aspect as a variation factor ("domain understanding: weak, solid, expert").

Impact on Baseline Hypothesis: The relationships between variation factors and the quality focus are stated here concretely. The information given in this quadrant must relate to the quality focus and it must be falsifiable; i.e., testable. This information is used to decide whether the variation factors listed in the second quadrant are valid. If people are not able to explain precisely the consequences of the variation factors on the quality focus, the respective variation factor is probably invalid with respect to the quality focus and should not be used in the GQM plan.

In the example of quality focus "effort," people might state that if the application domain is understood well, there will be fewer faults in the requirements, and this better understanding is expected to reduce the total effort spent by reducing the amount of rework. They use variations factors to state these impacts (e.g., "a solid domain understanding causes the requirements phase to require less effort than if developers have only a weak understanding").

Feedback: Concluding the example of quality focus "effort," the feedback section (a possible "fifth" quadrant) describes the feedback that can be provided to help improvement efforts.

4.3 Structure of GQM plans

A GQM plan documents the operational refinement of an analysis task. The task is precisely specified as a measurement goal that is refined via questions into metrics. The three layers of a GQM plan correspond to the following three levels. See also [BCR94b].

1. Conceptual level: A goal is defined for an object, for a single purpose, with respect to some model of quality, from a single point of view, relative to a particular environment.

2. Operational level: A set of questions is used to define in a quantitative way the goal and to characterize the way the data will be interpreted. Questions try to characterize the object of measurement with respect to a selected quality issue and to describe either this quality issue from the selected point of view or the factors that may affect the quality issue.

3. Quantitative level: A set of metrics is associated with every question in order to answer the question in a quantitative way. These metrics are the final piece of an operational definition of a goal.

4.3.1 Structure for product-oriented GQM plans

As mentioned above, GQM plans oriented towards analyzing products are distinguished from GQM plans oriented towards analyzing processes. Here we state guidelines for deriving questions starting from a product-oriented goal (i.e., to derive a product-oriented GQM plan). The structure of a product-oriented GQM plan is illustrated in Figure 2. See also [Rom91].

Three major categories of questions[1] need to be addressed for each product under study, namely the definition of the product, the definition of the quality focus, and feedback related to the quality focus.

Definition of the product. This first question category includes questions related to logical and physical attributes, development cost, changes during development, operational context, and other aspects that help characterize the product.

[1]These are sometimes labeled "subgoals," but are not expressed using the GQM goal template presented earlier.

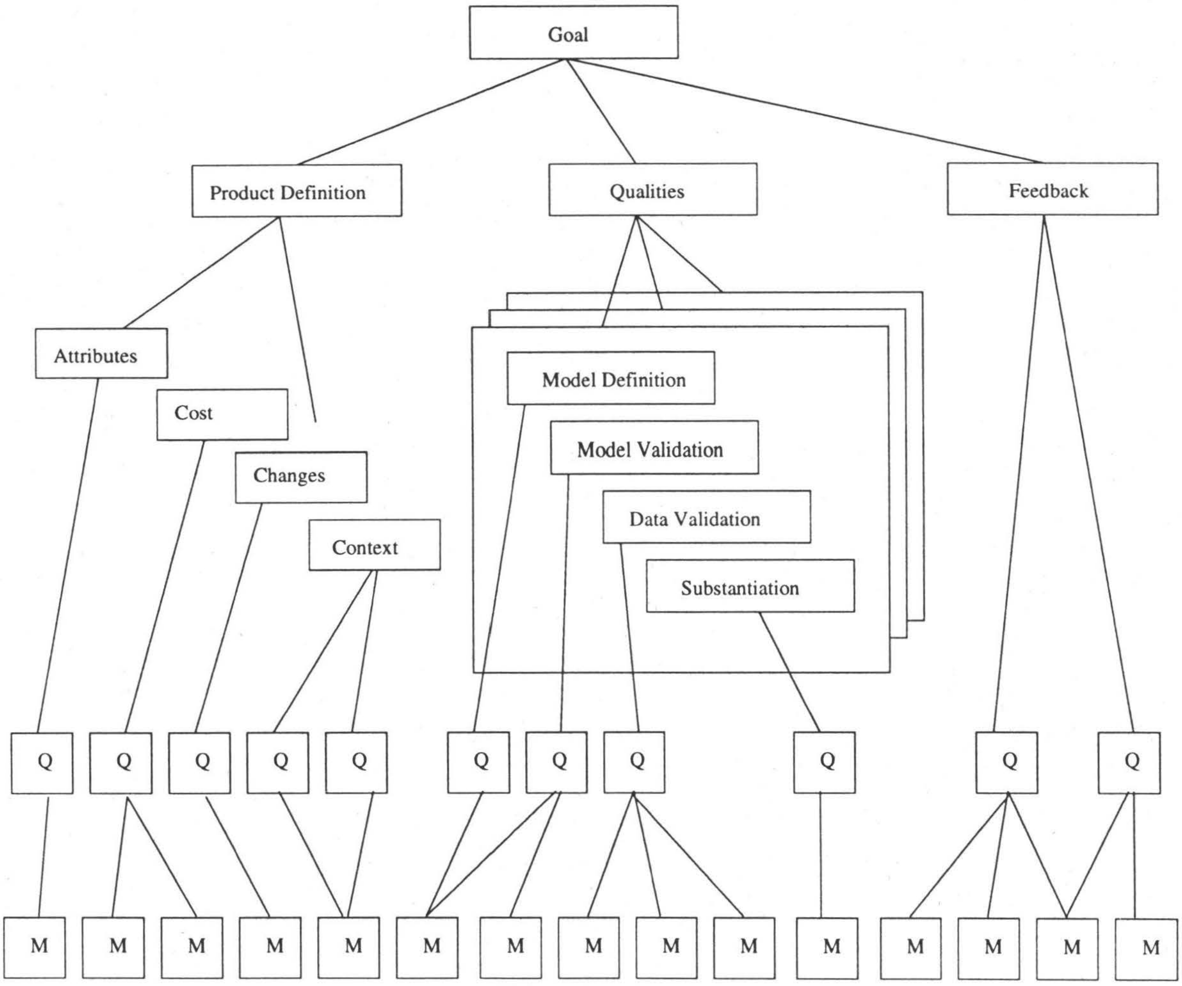

Figure 2: Structure of a product-oriented goal

Definition of the quality focus. This second category includes questions related to the major model of the quality focus that is used, the validity of the model for the particular environment, the validity of the data collected, and may also include a substantiation of the model. For example, a model of the quality focus "reliability" might be simply the number of critical and noncritical operational failures that were reporting during the acceptance test phase.

Feedback related to the quality focus. This third category includes questions, relative to the quality focus, that ask for information necessary when trying to improve the product.

4.3.2 Structure for process-oriented GQM plans

Next we state guidelines for deriving questions starting from a process-oriented goal. See also [Rom91]. The structure of a process-oriented GQM plan is illustrated in Figure 3.

Three major categories of questions[2] need to be addressed for each process under study, namely the definition of the process, the definition of the quality focus, and feedback relative to the quality focus(es) from using this process. A process-oriented GQM plan is the same as a product-oriented GQM plan with respect to the latter two categories of questions.

Definition of the process. The first category includes questions related to process conformance and domain conformance. Process conformance includes both a characterization of the process and an assessment of how well the process was followed. Domain conformance includes a characterization of the object to which the process is applied and an analysis of the process performer's knowledge concerning the object.

Definition of the quality focus and feedback. The purposes and structures of the second and third categories of questions for a process-oriented plan are essentially identical to their counterparts in the product-oriented plan; their contents are naturally very different (see Section 4.3.1).

[2] Again we do not name them "subgoals" in this document.

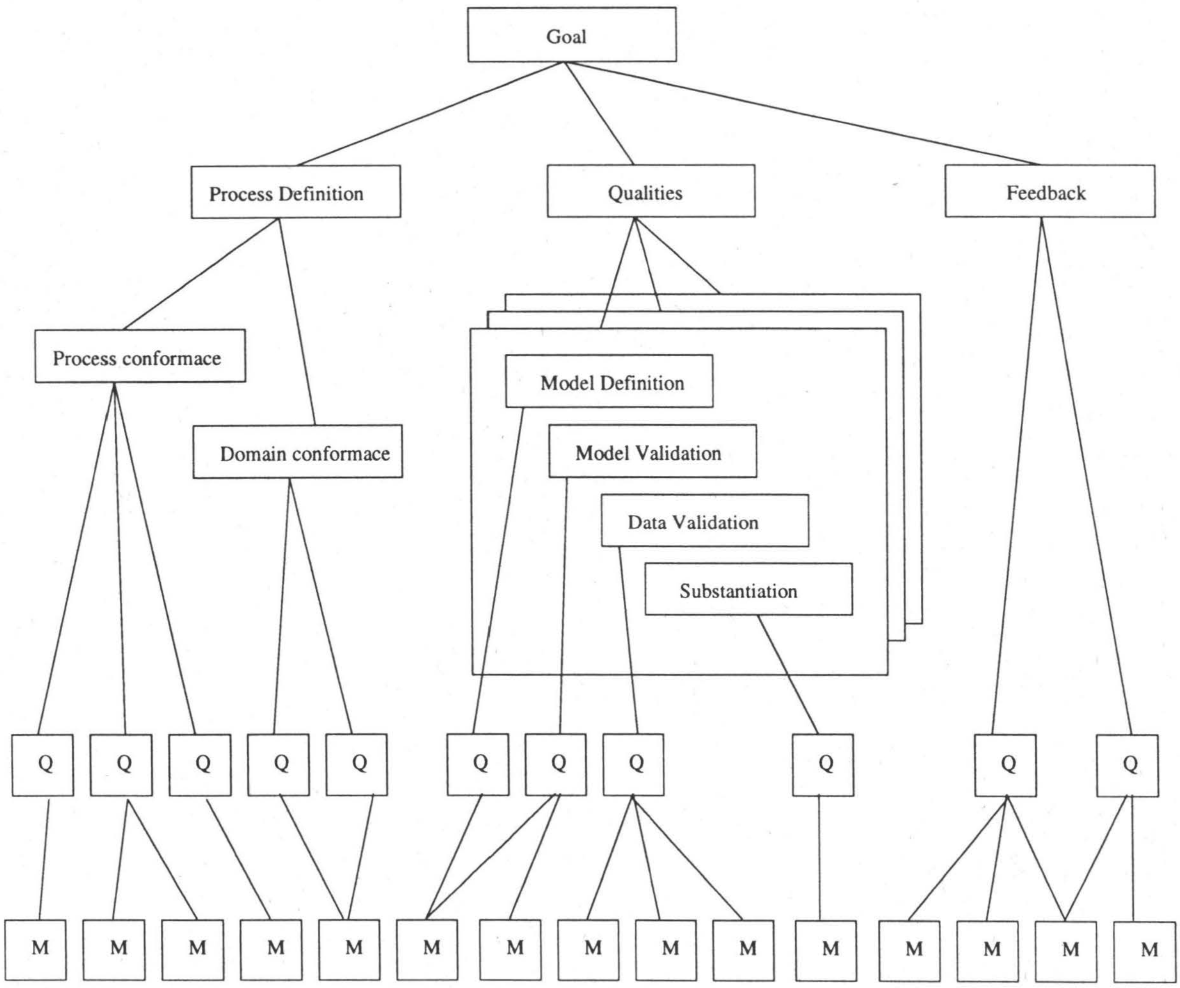

Figure 3: Structure of a process-oriented goal

4.3.3 Traceability between abstraction sheets and plan structure

Commonly a GQM abstraction sheet is created first, and then the information is transformed into the appropriate product-related or process-related GQM plan. This task can be eased by identifying the traceability between quadrants on the abstraction sheets and parts of the structure of a GQM plan.

Quadrant 1: The information in the first quadrant, as stated earlier, documents the quality focus. This information is commonly represented by the definition of the quality model category of questions (formalization of the quality focus), as defined by models, questions, metrics, or data representations.

Quadrant 2: The information in the second quadrant, as stated earlier, documents the variation factors that may affect the quality focus. This information is represented in the GQM plan by the group of questions etc. that defines the object of interest, i.e., the questions, metrics, or data representations.

Quadrant 3: The information in the third quadrant, as stated earlier, documents the baseline hypotheses. This information is treated similarly to the information in the second quadrant.

Quadrant 4: The information in the fourth quadrant, as stated earlier, documents the impact of the variation factors on the baseline hypotheses. This information adds hypotheses (e.g., possible relationship) concerning the information in the second quadrant.

Feedback section: The information in the feedback section, as stated earlier, documents the feedback for improving the product or process. This information is represented by the questions and metrics that are part of the "feedback" set of questions.

4.4 Measurement plans

A measurement plan describes *when*, *how*, and *by whom* the data required by the metrics in the GQM plan are collected. Essentially the measurement plan must determine the collection procedure (who, how, and when) for every metric in the GQM plan. If forms or tools will be used to collect data, these should either be included (forms) or be described in detail (tools) in the measurement plan. See [GHW95] for additional information. Also see [BDT96], in which Bröckers et al. describe how a process model helps with deriving the measurement plan from a GQM plan.

5 Methods for Applying the GQM Paradigm

Definition. A "method" as used here refers to some approach for performing a process that is part of planning and implementing a measurement program based on the GQM Paradigm. The focus is on performing processes. One or more of the techniques from the previous section will be used in these processes.

5.1 A process model for performing GQM-based measurement

Literature on GQM-based measurement primarily focuses on the description of the GQM Paradigm and GQM plans. However, the process of developing and using the GQM plans is not described in detail. Therefore a description of the GQM process, guidelines, and heuristics is needed to enable widespread use of GQM-based measurement. A first step towards this goal is the process model for planning GQM-based measurement. See also [GHW95].

The GQM process as modeled in [GHW95] starts from scratch and produces GQM plans and a measurement plan. The GQM plans contain the measurement goals as well as the models with respect to the measurement goals. To this end the GQM plans contain questions to be answered to achieve the measurement goals and metrics providing for data to answer the questions. The measurement plan describes exactly when and how and by whom which data will be collected. This process model was developed as part of the CEMP project and has been validated and revised in cooperation with several industrial partners.

6 Support tools

All of the tools discussed in this section are research prototypes. No commercial software is currently available that directly supports defining GQM plans in a top-down fashion, and interpreting data according to the plans.

6.1 ES-TAME

The ES-TAME system is a prototype of an expert system to support the design process for real-time software [OB92]. ES-TAME also offers a sophisticated, highly structured framework for building

GQM plans. The user can build a GQM plan in the form of a set of goals, questions, and metrics by using templates to write goals and then either selecting from a predefined set of questions and metrics or writing new questions and metrics. The expert system part of ES-TAME can use forward chaining to guess elements of the GQM plan under construction. When interpreting the plans, the system is intelligent enough to ask the user for data and to look in existing quality models to obtain those data when possible.

6.2 Formal language and object model for GQM

This project defined a formal language in which a GQM plan can be written, and a corresponding computer (internal) representation for storing plans encoded using the formal language. See also [Dif93]. The language was later extended to support the definition of hypotheses; see [vM95].

6.3 Syntax-directed editor for GQM plans

This project implemented an editor that supports the entry and modification of GQM plans that are encoded using the formal language defined by a previous project. See also [Fri94, vM93].

6.4 GQM-DIVA: Definition, Interpretation, and Validation

This project implemented a system that supports the definition of GQM plans via a WIMP interface, the validation of those plans in terms of missing or inconsistent elements, and the interpretation of collected data via user-defined procedures. See also [vM95].

7 Current research on GQM

As of this writing, ongoing research on the GQM Paradigm focuses on many different aspects. Some work is highly specific to a given application of the GQM Paradigm, while other work focuses on the GQM Paradigm itself. Several research projects are summarized briefly.

The ESPRIT project "Perfect" aims at assisting measurement-based improvement of software processes. Work is being conducted on augmenting the GQM technology based on the experiences of industrial application projects. The main contributions of the project "Perfect" are a handbook on GQM, a GQM process description, two tutorials on establishing GQM-based measurement programs, and a tool for supporting activities of the GQM process. The handbook provides an introduction and overview of GQM and describes the use of GQM within the "Perfect Improvement Approach" for the continuous improvement of software processes. The GQM process description involves a strategic (or "organizational") view as well as a software project's view on goal-oriented measurement. The two tutorials support the introduction of GQM into an organization and its projects. The tutorials are entitled "An introduction to goal-oriented measurement" and "Participation of project teams in measurement programs," respectively. Parts of the GQM-related results from project "Perfect" were made available in Spring 1996, namely the two tutorials. The entire set of results will be available by Fall 1996.

Other work on technology transfer was done in the context of the European Software and Systems Initiative (ESSI) project "Customized Establishment of Measurement Programs" (CEMP) at the University of Kaiserslautern. The CEMP project prepared a process model for GQM-based measurement to support introduction of the GQM approach into industry (see [GHW95]). The CEMP project also formulated experiences with the GQM Paradigm with respect to the costs and benefits of introducing GQM-based measurement in industry. The final report from the CEMP project should be available in Fall, 1996.

Work on supporting software engineering experiments using GQM is being conducted in the context of Special Research Area 501 (SFB) at the University of Kaiserslautern. The SFB is working on developing a process model for designing experiments according to the GQM Paradigm.

Work is in progress at the University of Maryland on formalizing the components of GQM plans and providing guidance for using the GQM Paradigm. This work is being done in the context of a doctoral dissertation by Manoel Mendonça.

Based on the strong relationship between measurement and process modeling, some work was done on investigating the integration of process modeling and GQM-based measurement. This work was done in the context of Christopher Lott's doctoral dissertation, in cooperation with the Software Engineering Research Group at the University of Kaiserslautern [Lot96]. The integration is based on the use of the Multi-View Process Modeling Language (MVP-L).

Other related work in progress at the University of Kaiserslautern includes formalizing GQM plans for better support of data interpretation and reusability. This work is being done in the context of a doctoral dissertation by Christiane Differding.

8 Conclusion

In this report we briefly introduced principles, techniques, methods, and tools that support the GQM Paradigm, surveyed current research on GQM, and identified key sources for further learning. This report can be considered a success if it succeeds in pulling many threads of research on the GQM Paradigm together in a single place.

A Glossary

GQM Paradigm: A collective term for the set of basic principles concerning goal-oriented measurement, the templates and guidelines that assist in defining goals, and methods of applying the basic principles.

GQM approach: Synonymous with **GQM Paradigm**.

GQM plan: A single goal plus the sets of questions and metrics that provide an operational definition of that goal. A GQM plan documents the refinement of a precisely specified measurement goal via a set of questions into a set of metrics. Thus, a GQM plan documents which metrics are used to achieve a measurement goal and why these are used - the questions provide the rationale underlying the selection of the metrics. On the other hand, the GQM plan is used to guide analysis tasks because it documents for which purpose the respective data were collected.

GQM model: Sometimes used as a synonym for **GQM plan**. Alternatively, this may refer to some type of model (e.g., a quality model, cost model, process model, etc.) that is comprised of nothing more than a set of questions within a GQM plan.

Measurement plan: When coupled with a GQM Plan, a measurement plan specifies who collects the data required by the GQM Plan, how the data is collected, and when the data must be collected. A measurement plan usually includes the data-collection forms as well as descriptions of tools that perform online data collection.

Metric: In measurement theory, this is a mapping used to assign a value to some attribute of an entity.

GQM metric: This describes something that we would like to measure. Opinions vary as to whether a GQM metric must be directly collectible (e.g., "time in days") or whether a GQM metric may be something as complex as "productivity." The argument for directly collectible metrics is that all necessary information for refining a goal into collectible metrics should be captured in the GQM plan. The argument for permitting complex metrics is that a metric need only state clearly what a person would like to know.

The use of complex metrics in a GQM plan implies that some refinement of GQM metrics into directly collectible data items may be necessary before beginning to use a GQM plan. Some hold the opinion that a GQM metric need not even be collectible at all; in other words, no refinement may be necessary or even possible. In the case of a noncollectible metric, stating the metric would merely serve to illustrate what is desired, and would thereby reveal the limitations of the metrics for which data are collectible.

Annotated Bibliography

[Bas89] Victor Basili. Software development: A paradigm for the future. Technical Report UMIACS-TR-89-57, Department of Computer Science, University of Maryland, June 1989.

Basili discusses treating software development as an experimental activity from which organizations can learn and improve. He introduces his paradigm for software development (later refined into the Quality Improvement Paradigm, QIP) and discusses the "Experience Factory" as a way of separating day-to-day project concerns from long-term organizational concerns such as learning. Explains how the GQM Paradigm is used in the context of his QIP.

[Bas92] Victor R. Basili. Software modeling and measurement: The Goal/Question/Metric paradigm. Technical Report CS-TR-2956, Department of Computer Science, University of Maryland, College Park, MD 20742, September 1992.

Basili discusses the GQM Paradigm in some detail, including a process for applying the paradigm (defining goals, questions, and metrics) as well as generating operational models of cost, reliability, etc. He also gives directions for presenting data that was collected, and offers detailed examples of process and product goals.

[Bas94] Victor R. Basili. GQM approach has evolved to include models. *IEEE Software*, 11(1):8, January 1994. Letter to the editor.

Basili addresses the criticisms offered by Card about the GQM Paradigm [Car93], explaining that more recent work on GQM solves many of the problems that Card identified.

[BCR94a] Victor R. Basili, Gianluigi Caldiera, and H. Dieter Rombach. Experience Factory. In John J. Marciniak, editor, *Encyclopedia of Software Engineering*, volume 1, pages 469–476. John Wiley & Sons, 1994.

Basili et al. give an overview of the Quality Improvement Paradigm and the Experience Factory, an organizational structure designed to support the systematic capture and reuse of all types of products, models, and experience within the context of a software development organization.

[BCR94b] Victor R. Basili, Gianluigi Caldiera, and H. Dieter Rombach. Goal Question Metric Paradigm. In John J. Marciniak, editor, *Encyclopedia of Software Engineering*, volume 1, pages 528–532. John Wiley & Sons, 1994.

Basili et al. briefly introduce the principles and refinement levels of the GQM Paradigm and sketches the process of developing a GQM plan.

[BCR94c] Victor R. Basili, Gianluigi Caldiera, and H. Dieter Rombach. Measurement. In John J. Marciniak, editor, *Encyclopedia of Software Engineering*, volume 1, pages 646–661. John Wiley & Sons, 1994.

Basili et al. review the history of measurement in software engineering, including models, measures of products including size, software science and cyclomatic complexity, measures of processes including effort and effort distribution, and additionally discuss the validation of measures.

[BMB93] Lionel C. Briand, Sandro Morasca, and Victor R. Basili. Assessing software maintainability at the end of high-level design. In *Proceedings of the International Conference on Software Maintenance*. IEEE Computer Society Press, September 1993.

Briand et al. apply the GQM Paradigm for measuring and assessing maintainability at the end of high-level design. A detailed example of a goal and its refinement into questions and metrics, as well as of interpreting the data in terms of the goal.

[BR88] Victor R. Basili and H. Dieter Rombach. The TAME Project: Towards improvement–oriented software environments. *IEEE Transactions on Software Engineering*, SE-14(6):758–773, June 1988.

Basili and Rombach describe the TAME project, including their holistic view of software engineering planning, project performance, measurement, and analysis. They use their views to develop a set of requirements for automated support for these activities (the TAME system) and discuss the GQM Paradigm in the context of their Quality Improvement Paradigm (QIP).

[BW84] Victor R. Basili and David M. Weiss. A methodology for collecting valid software engineering data. *IEEE Transactions on Software Engineering*, SE-10(6):728–738, November 1984.

Basili and Weiss present an early version of GQM in the context of collecting and validating data about changes made to software systems.

[BDT96] Alfred Bröckers, Christiane Differding, and Günter Threin. The role of software process modeling in planning industrial measurement programs. In *Proceedings of the Third International Software Metrics Symposium*, Berlin, March 1996. IEEE Computer Society Press.

Bröckers et al. present guidelines for developing a measurement plan for an industrial software development project using GQM plans and detailed process models.

[Car93] David N. Card. What makes for effective measurement? *IEEE Software*, 10(6):94–95, November 1993.

Card offers a brief critique of the GQM paradigm, and states that an organization will need either significant experience or assistance when using the GQM Paradigm to set up a measurement program.

[Das92] Michael K. Daskalantonakis. A practical view of software measurement and implementation experiences within Motorola. *IEEE Transactions on Software Engineering*, 18(11):998–1010, November 1992.

Daskalantonakis describes the Motorola company's effort at defining and implementing a company-wide software metrics program based on the GQM Paradigm.

[Dif93] Christiane Differding. An object model for supporting the GQM paradigm (in German). Master's thesis, Department of Computer Science, University of Kaiserslautern, 67653 Kaiserslautern, Germany, June 1993.

Differding implemented a system for storing the information developed as part of a GQM plan.

[Fri94] Ralf Friedl. A tool for evaluating GQM plans (in German). Master's thesis, Department of Computer Science, University of Kaiserslautern, 67653 Kaiserslautern, Germany, October 1994.

Friedl implemented a system that supports the process of interpreting data collected for GQM metrics in the context of questions and ultimately goals.

[GHW95] Christiane Gresse, Barbara Hoisl, and Jürgen Wüst. A process model for planning GQM-based measurement. Technical Report STTI-95-04-E, Software Technology Transfer Initiative, University of Kaiserslautern, 67653 Kaiserslautern, Germany, October 1995.

Gresse et al. present a model for the complete GQM process, beginning with the characterization phase and ending with the packaging of experience. A detailed description of the process steps and related products is given for each phase of the GQM process Guidelines and heuristics for performing each step are also provided. In addition, a comprehensive example of a GQM-based measurement program is included. The model is based on experience gained in the ESSI project CEMP while conducting pilot projects at three sites throughout Europe. (This report supersedes [Hoi94].)

[Hoi94] Barbara Hoisl. A process model for planning GQM-based measurement. Technical Report STTI-94-06-E, Software-Technology-Transfer-Initiative Kaiserslautern, University of Kaiserslautern, 67653 Kaiserslautern, Germany, April 1994.

Hoisl describes a process model for planning GQM-based measurement and states the role of abstraction sheets in that process. This model was developed in the context of the European Software and Systems Initiative (ESSI) Project "Customized Establishment of Measurement Programs" (CEMP). (This report is the precursor to [GHW95].)

[Hum89] Watts S. Humphrey. *Managing the Software Process.* Addison Wesley, Reading, Massachusetts, 1989.

Humphrey discusses many issues in managing software projects, including the definition and use of metrics.

[Lot96] Christopher M. Lott. *Measurement-based feedback in a process-centered software engineering environment.* PhD thesis, Department of Computer Science, The University of Maryland, College Park, Maryland 20742, February 1996.

Chapter 3 of Lott's dissertation offers an integration of the process-modeling language MVP–L with the Goal Question Metric paradigm to establish a foundation for guiding developers in their work with quantitative quality models.

[OB92] Markku Oivo and Victor R. Basili. Representing software engineering models: The TAME goal oriented approach. *IEEE Transactions on Software Engineering*, 18(10):886–898, October 1992.

Oivo and Basili discuss ES-TAME, a prototype system that supports the representation of various software engineering models including GQM plans.

[PR94] Jenny Preece and H. Dieter Rombach. A taxonomy for combining software engineering and human-computer interaction measurement approaches: Towards a common framework. *International Journal of Human–Computer Studies*, 41:553–583, 1994.

Preece and Rombach use GQM to develop a taxonomy that helps characterize empirical studies done in both the software engineering and the human-computer interface domains. This taxonomy is used to identify commonalities and differences in empirical work on those domains.

[Rom91] H. Dieter Rombach. Practical benefits of goal-oriented measurement. In N. Fenton and B. Littlewood, editors, *Software Reliability and Metrics*, pages 217–235. Elsevier Applied Science, London, 1991.

Rombach compares three approaches to top-down planning of measurement (QFD, SQM, and GQM) and elaborates on GQM. Also discusses benefits of using the GQM Paradigm as to plan and perform a measurement program.

[RU89] H. Dieter Rombach and Bradford T. Ulery. Improving software maintenance through measurement. *Proceedings of the IEEE*, 77(4):581–595, April 1989.

Rombach and Ulery apply the GQM Paradigm in a maintenance improvement program. They also present a measurement model that is a prescription for applying the GQM Paradigm and survey three projects in which maintenance was measured.

[vM93] Marco van Maris. A syntax-directed editor for GQM plans (in German). Projektarbeit, Department of Computer Science, University of Kaiserslautern, 67653 Kaiserslautern, Germany, August 1993.

Van Maris implemented a syntax-directed editor for GQM plans in the context of a student research project.

[vM95] Marco van Maris. GQM-DIVA, a tool for defining, interpreting and validating GQM plans (in German). Master's Thesis, Department of Computer Science, University of Kaiserslautern, 67653 Kaiserslautern, Germany, 1995.

Van Maris implemented a graphical user interface that supports defining GQM plans as well as interpreting and validating the data that was collected for GQM plans.

[Wei94] Dave Weiss. GQM plus heuristics better than brainstorming. *IEEE Software*, 11(1):8–9, January 1994. Letter to the editor.

Weiss addresses the criticisms offered by Card about the GQM Paradigm [Car93] by offering some of the heuristics he uses when applying GQM.