7th International Conference on Empirical Assessment in Software Engineering (EASE), 2003, Keele, UK

Software Measurement for Small and Medium Enterprises

-A Brazilian - German view on extending the GQM method -

Christiane Gresse von Wangenheim[1], Teade Punter[2], Alessandra Anacleto[3]

[1] Universidade do Vale do Itajaí/Centro de Eduação São José, Rod. SC 407/Km 04, 88122-000 São José/SC, Brazil, gresse@sj.univali.br
[2] Fraunhofer IESE, Sauerwiesen 6, 67661 Kaiserslautern, Germany, punter@iese.fhg.de
[3] Universidade Federal de Santa Catarina, Campus Universitário - Trindade, 88049-200 Florianópolis/SC, Brazil, ale@inf.ufsc.br

Abstract

Today many large, medium and small software companies experience difficulties in establishing quality improvement initiatives, although we can observe that unlike large companies, small and medium sized enterprises (SMEs) find it particularly difficult to adapt these quality initiatives. This is further complicated since most software quality initiatives, such as, e.g., CMM and SPICE, primarily address the needs of large software organizations. Thus, in order to provide a basis for the improvement of software quality and productivity also for small software companies, we propose a customized approach to measurement as an essential infrastructure for software quality improvement, which takes into account the specific characteristics and limitations of small enterprises. Our approach basically consists of the integration of the systematic reuse of context-specific quality and resource models in the planning of measurement programs, and of a compaction of the measurement process to a lightweight GQM method in order to reduce the measurement overhead. The approach is based on our experiences in applying measurement in software SMEs, and first results are presented.

Keywords: Software Measurement, Goal Question Metric (GQM), Reuse of Quality Models, GQM Lightweight, Assessment, Evaluation.

1 Introduction

Software engineering is not only done by large companies like Microsoft, Nokia or Siemens, that belong to the world's largest software development organizations. A considerable amount of software is produced world-wide by small and medium-sized enterprises (SMEs) ranging from 1 to about 50 employees. On the German software market, for example, small companies largely characterize the primary sector in software development and maintenance (77% in 2000), whereas medium-sized and larger companies dominate the secondary sector that is related to, e.g., mechanical engineering, telecommunications, and financial services [GfK00]. A similar situation is observed in Brazil, where the primary software sector is

basically composed of small and medium-sized software companies (69% in 2001) [MCT01].
In general, many software companies – regardless size – have difficulties in establishing quality improvement initiatives. For example, the number of software companies that have adopted reference models like CMM/CMMI, SPICE (ISO 15504) or ISO 9001 is still quite low. Till August 2002, only 21 official CMM assessments were reported for Germany (about 17,315 companies were related to German software production in 2000) and only 16 CMM assessments were reported for Brazil [SEI01].

Therefore, we observe that unlike large companies, software SMEs have little awareness of and/or encounter specific difficulties in adapting these quality initiatives, due to limited financial and human resources. In general, SMEs are frequently characterized by software processes that are hardly specified. Often, they only have a small number of employees with multiple role assignments and do not have resources for allocating dedicated staff and knowledge to start quality initiatives. For example, in 2001, only 25% of the Brazilian small companies had established a quality system (e.g., a total quality program or similar). In addition, small software companies are frequently start-up companies, which have an additional disadvantage compared to mature organizations regarding their lack of experience and infrastructure. These problems are further complicated as most software quality initiatives, such as, e.g., CMM or ISO9000, are primarily addressed at large software organizations and require considerable resources, infrastructure and experience.

In the context of any of those quality initiatives, measurement has proved to be an essential necessity for controlling software projects and improving quality as much as possible. Measurement is one of the Key Process Areas (CMM(i)) or Base Practices (SPICE) of the aforementioned reference models. Although measurement is applied in various areas and sciences, it has been shown to be a complex and difficult undertaking in the software domain and especially in the context of SMEs, due to their specific characteristics and limitations. For example, in Brazil, more than 80 % of the SMEs do not use any kind of size measurement [MCT01]. A survey among German software engineering companies [LVC02] shows that more than 50% of the respondents do not collect metrics. The data are either basically or systematically analyzed (75%), although 18% of the respondents state that they are not analyzed at all.

Thus, in order to provide a basis for the improvement of software quality and productivity, we propose a customized approach for measurement. Our approach

is based on the Goal/Question/Metric (GQM) Method as motivated in the next section. We propose a more lightweight GQM method to enable measurement in Small and Medium-sized Enterprises as described in Section 3. The adaptation of our approach basically consists of the integration of the systematic reuse of quality and resource models as described in Section 4. In Section 5, first experiences in applying this customized approach in practice are presented.

2 Software Measurement and Quality & Resource Models

It is generally accepted that in order to manage software projects effectively and improve software quality in a continuous way, context-specific quality and resource models have to be built that are based on quantitative and qualitative data collected through organization-specific measurement programs [BCR94]. Such quality and resource models can model a variety of relevant quality or productivity aspects (e.g., reliability, usability, maintainability, effort) by operationally defining the entities and attributes to be measured, the set of measures used and the relationships among several measures. For building such models, we need to determine how to model the aspects adequately, what to measure, and how to analyze and interpret the collected data and their relationships.

Two approaches to measurement have been distinguished. On the one hand, universal quality frameworks have been proposed, such as [McC77], which suggest pre-defined ways to decompose software quality into a set of components and further into a set of metrics. However, these universal models assume that all important quality factors in any context are a subset of those of the model. Several authors have already stated that more research is needed to confirm that internal quality assures external quality [KP96]. In addition, these frameworks do not provide guidance on how to select relevant factors and metrics and, if necessary, on how to adapt them to a specific environment.

On the other hand, measurement methods have been developed, for deciding what to measure and how to interpret the collected data. Basically, there exist two types of measurement approaches [PR94]: top-down and bottom-up. Bottom-up approaches start with measurable observations and build up to management objectives and goals, whereas top-down approaches help to derive useful measures from goals and interpret the collected data in the context of the goals of interest. However, in comparison to bottom-up approaches, top-down approaches have been shown to support the adequacy, consistency, and completeness of the measurement plan and to help manage the complexity of measurement programs [Rom91]. Different top-down measurement approaches have been developed as a

basis for the definition and implementation of tailored measurable and operational software improvement goals in a specific context, such as Quality Function Development (QFD) [KA83], Software Quality Metrics Approach (SQM) [Mur80], and Goal/Question/Metric Approach (GQM) [BCR94b, BW84].

Those top-down approaches focus on the relationship of the measures to the measurement goals, representing generic methods, instead of defining a pre-defined set of measures, and, e.g., like the GQM approach, guide the top-down definition of a customized set of relevant measures and the bottom-up analysis and interpretation of the collected data with respect to the measurement goal tailored to the company-specific context. Yet, the approaches differ significantly in terms of the scope of supported measurement goals, guidance for the identification of the relevant measures and their potential uses. In that sense, the GQM approach has been shown to be the most flexible of the three approaches [Rom91].

However, as these approaches, in general, start each new measurement program from scratch, measurement remains an intellectually complex process, which requires a considerable amount of effort, time and expertise [CEM96]. This turns out to be a problem especially for small and medium-sized enterprises due to their limited resources for investments in improving software quality and productivity compared to large companies. Here, we observe that software SMEs encounter specific difficulties as they generally do not have resources for allocating dedicated and experienced staff to measurement initiatives. In addition, small software companies are frequently start-up companies, which are often characterized by informal processes, the lack of systematic project management, their primary focus on getting the product out as well as their lack of experience.

In order to have a trade-off between the application of universal and validated models and the need for measurement customized to a specific environment, we aim at an integration of both approaches, especially taking into account the characteristics and limitations of SMEs. Therefore, we propose an enhancement of the GQM method, by enabling the systematic reuse of context-dependent quality and resource models that have been developed in past measurement programs, on the other hand, by compacting the measurement process to a lightweight GQM method in order to reduce measurement overhead.

3 A GQM Lightweight Process

The Goal/Question/Metric (GQM) paradigm [BCR94b,BR88,BW84] has been proposed as a goal-oriented approach for the measurement of products and

processes in software engineering. It is a mechanism for defining and evaluating a set of operational goals, using measurement. GQM represents a systematic approach for tailoring and integrating goals with models of the software processes, products and quality perspectives of interest, based upon the specific needs of the project and the organization.

The GQM paradigm was first developed in 1984 at the University of Maryland [BW84] in cooperation with the NASA Goddard Space Flight Center [BCG+92] and has been extended as part of the TAME project [BR88]. A first GQM process was defined by Basili in 1992 [Bas92], which was extended by the definition of a structure for GQM plans [Rom91] and the explicit representation of assumptions [BMB96]. Based on the results of the ESPRIT/ESSI project CEMP, the GQM process was refined [vSB99, GHW95]. In 1996, SEI defined a handbook where the goal-driven software measurement is presented [PGF96]. Recent enhancements of the GQM process are mostly directed at formalizing the development of the GQM plan based on explicit quality models, as done, e.g., by [CD99, MBB+98, OJ97, GM97, Gre02a], or on the integration of reuse [Gre02a,Dif01]. However, there does not exist any approach directly addressing small software companies and considering their specific requirements and limitations.

Based upon these existing GQM process models, especially [vSB99, PGF96, Gre02a, GHW95] and our experiences in applying GQM-based measurement in small software companies, we describe a customized GQM process model that combines the existing models adapted to the specific characteristics and limitations of small software companies. Table 1 shows an overview of the principal phases and activities of the GQM "lightweight" process, referred to in the following as *GQM Lightweight*.

In general, GQM Lightweight is based on [Gre02b, GHW95] regarding its principal phases and activities. However, each of the activities has been revised, shortcut or adapted when possible, especially due to the predominant informality, the small number of employees, and the limited resources in small software companies.

The *planning phase* prepares the establishment of software measurement in the organization. The organization and its projects are characterized in order to get a better understanding of the context. Normally, a pilot project is selected to introduce measurement and all people involved in the measurement program are motivated and trained. We think that less attention is needed for GQM-planning in

an SME. For example, in the context of an SME, no separate measurement team will be established due to the small number of employees and informal structures of the organizations. Here, one person of the organization will rather be allocated part-time to measurement responsibilities. In addition, much less effort has to be spent on promotion and training, as, in general, there are less people involved in the pilot project and shorter communication channels in such organizations make planning less intensive. However, we also observed that measurement programs are mainly supported by one or two persons who are convinced of their added value. For example, in a fault measurement program in a small development department (8 engineers), it was assumed that the direct communication within the organization would facilitate feedback of the data collection forms. However, developers had to be asked again and again to submit their data collection forms, although they originally expressed their interest and although they were involved in the development of the forms, which were quite simple and easy to use.

Phases	**Approach for SMEs** ***GQM Lightweigh***	**GQM method [vSB99]**	**Goal-driven measurement [PGF96]**
Planning	Introduce measurement program	Establish GQM team, Create project plan, Training and promotion	
		Select improvement areas, Select application project & establish project team	Identify business goals, Identify what to know or learn
Definition	Define measurement goals, Goal formalization	Define measurement goals, Conduct GQM interview, Review or produce software process models	Identify subgoals, Identify entities and attributes, Formalize measurement goals
	Define questions	Define questions & hypothesis and Review	Identify quantifiable questions
		Produce analysis plan	Identify indicators & data elements
Reuse of Quality & Resource models	Define metrics	Define measures and Review	Define measures
	Produce GQM plan, Define data collection procedures, Define data instruments	Produce GQM plan, Produce measurement plan	Identify the actions needed to implement measures, Prepare a plan
	Produce data collection plan Create metrics base	Trial period, Hold a kick-off session	
Data collection	Collect and validate data	Create metrics base	
	Store data collected	Collect and check data collection form, Store measurement data in metrics base	
Interpretation	Data analysis	Define analysis sheets and presentation slides	

	Data interpretation – Feedback session	Prepare feedback session, Organize and hold feedback session, Report measures resulting	
Packaging	Packaging results		

Table 1 Overview of GQM Lightweight and comparison with two existing GQM approaches

An explanation for this is that in the daily software development job, engineers have to prioritize their work load and cope with a multitude of information, which might result in lowering attention to the measurement program (this is true for both large and small companies). Therefore, the kick-off session whose objective is to get agreement of all people involved in the measurement program is essential but not enough for the success of the measurement program. Motivation for the measurement program should be reconfirmed during the process; we think that feedback sessions (see below) can support this.

The *definition phase* aims at defining the GQM-based measurement program. This includes the definition of the GQM goal(s) to be achieved by the measurement program, the development of the GQM plan, including questions and measures, and the development of the measurement plan defining data collection procedures and instruments. Here, these basic activities also have to be followed in an SME and a GQM and measurement plan have to be defined explicitly in order to have an acceptable foundation for the measurement program.

However, significant support can be provided by the reuse of quality and resource models during this step (see Section 4), resulting in a reduction of the definition effort and facilitating the definition. Our experience has further shown that the identification and selection of measurement goals is easier in SMEs partly due to the fact that less people are involved in the pilot project and, therefore, in the goal definition process, and often the goals defined in SMEs turn out to be more restricted.

Once the GQM plan and measurement are defined, they have to be reviewed. Here, in order to reduce the respective review effort, only project personnel review the Abstraction Sheets and data collection instruments, whereas the other parts of the GQM and the measurement plans are revised with respect to their consistencies by the person(s) responsible for the GQM program.

During the *data collection phase* the data is collected according to the procedures specified in the measurement plan. Effort reductions in the data collection phase can be achieved by developing suitable data collection instruments that are very well integrated into the software process of the specific SME. Wherever possible, data collection should be automated, e.g., through integration into existing tools at

the organization. However, due to the large degree of informality, the majority of measures often have to be collected manually. Therefore, data collection sheets have to be developed that can be used in paper form or online depending on the specific characteristics of the organization. For example, employees in one of the SMEs where we applied measurement used to work at different locations and did not necessarily have Internet access at all locations, and, therefore, preferred paper sheets. However, the collection using paper sheets requires an additional effort for the inclusion of the data into the measurement database.

The *interpretation phase* aims at the periodic analysis of the collected data and interpretation during feedback sessions involving project personnel following the GQM plan bottom-up. The intervals in which the collected data is analyzed and interpreted should not be too short in order to keep the effort low, but also not too long in order to provide feedback in time. On the one hand, intervals depend on the measurement goal (e.g., when the respective information has to be available) and, on the other hand, on the availability of the necessary data in order to be able to answer a sufficient number of questions of the GQM plan.

The measurement database used to store and process the collected data for analysis can simply be made with spreadsheet tools and database management systems (DBMS). Spreadsheets are interesting to use because of their flexibility in supporting goal adjustments during the execution of the measurement program. The advantages of using a DBMS are the availability of languages to define the data structure and the operations on the stored data and their support for consistency checks. Based on our experiences, depending on the total duration of the execution of the measurement program, it can be beneficial to automate the data analysis process as much as possible. Although, in general, this requires a higher effort in the beginning when developing the automated support, it results in effort reduction each time the data is analyzed. However, if depending on the specific measurement goal, the analysis is only performed a few times, this may not justify the higher initial effort.

We think that it is absolutely essential to perform the feedback sessions in order to obtain valid interpretations and to keep up the motivation in measurement. However, in small companies, such feedback sessions will not be as formal as in large companies, and should be prepared carefully in order to keep the time required for these meetings minimal.

Another more critical issue regarding the data analysis and interpretation in SMEs is the confidentiality of the data collected. Due to the fact that the person responsible for measurement in an SME often is a person with multiple role assignments (e.g., as project manager and developer), it may be difficult to

guarantee the anonymous analysis of the data. As a consequence, it becomes even more important not to use the collected data for the evaluation of people in the organization.

Once the measurement program is finished, in the *Packaging phase*, the measurement results including the collected data and its interpretation are analyzed, packaged and stored in a way suited to the organizational context so that this knowledge can be reused in future software projects and measurement programs.

The packaging should especially focus on documenting the GQM plan, the results of the feedback sessions, and cost and benefits reported.

4 Reusing Quality and Resource Models

Measurement aims at defining the software qualities to be analyzed operationally in relation to the particular environment. This is explicitly done in the form of quality or resource models, which operationally define a software quality or resource aspect of interest relative to an explicitly and precisely defined goal and context. *Quality models* describe quality attributes of all kinds of products or processes, for example, reliability, usability or maintainability, whereas *resource models* describe models regarding resource consumption in the context of software development and maintenance, e.g., on effort or duration. In general, those models are often related to abstract concepts. Thus, one of the major problems is to refine these aspects into operational models, leading to measurement that takes into account the characteristics of the particular environment. In order to develop a measurement program within a particular context oriented to a specific goal to be achieved by measurement, we need to define a customized and operational quality or resource model that defines relevant measures. Following the GQM approach [BDR96, BCR94b, GHW95, Gre02a], the elements of such a quality or resource model are:

- The *measurement goal* defining precisely the object of study, the purpose, the quality focus, the viewpoint and the context in which the respective model is valid.
- An *object model* describing the object of study of the measurement goal (e.g., the software process model).
- A high-level *quality focus model* refining the quality focus (e.g., documented in the form of an Abstraction Sheet).
- A set of *questions*, expressing the information of interest in order to achieve the goal.

- A *GQM model* operationalizing each of the questions by providing a means of to answer the questions.
- *Measures* for each of the attributes/entities to be measured in order to feed a GQM model.
- *Data collection procedures,* which define for each measure when, how, and by whom the data are to be collected, validated and stored.
- *Data collection instruments,* which implement the measurement plan, e.g., tools or questionnaires.
- *Analysis and interpretation procedures,* which define when and how to analyze and interpret which questions of the measurement program.
- *Packaging procedures*, which define how to capture and package the models in order to make them reusable in the future.

In the traditional GQM method, such models are generally developed from scratch. In practice, we have seen that such operationalization is not as easy as it seems to be. Especially the planning of measurement programs has been shown to be an intellectually complex and time-consuming process, difficult to be applied in small and medium-sized software companies [CEM96]. We, therefore, aim at the reuse of context-specific quality and resource models instead of starting each new measurement program completely from scratch.

However, instead of using "universal frameworks" that are supposed to be applicable in any context, we rather focus on the reuse of context-specific models that have been developed in past measurement programs within a similar context and are related to a similar measurement goal. In contrast to "universal frameworks", these quality and resource models present measurement knowledge that is gathered and valid only in a specific scope of context. Such organization-specific models can frequently be reused, as a company often develops various software systems with similar characteristics in the same domain. This approach balances the trade-off between relying upon universal quality models –that do not exist – and 'reinventing the wheel each time'. The benefits of such an integration are the improvement of the planning of customized measurement programs, which are more likely to address the specific needs than, e.g., universal frameworks, the reduction of measurement effort, and the provision of greater support for measurement planning.

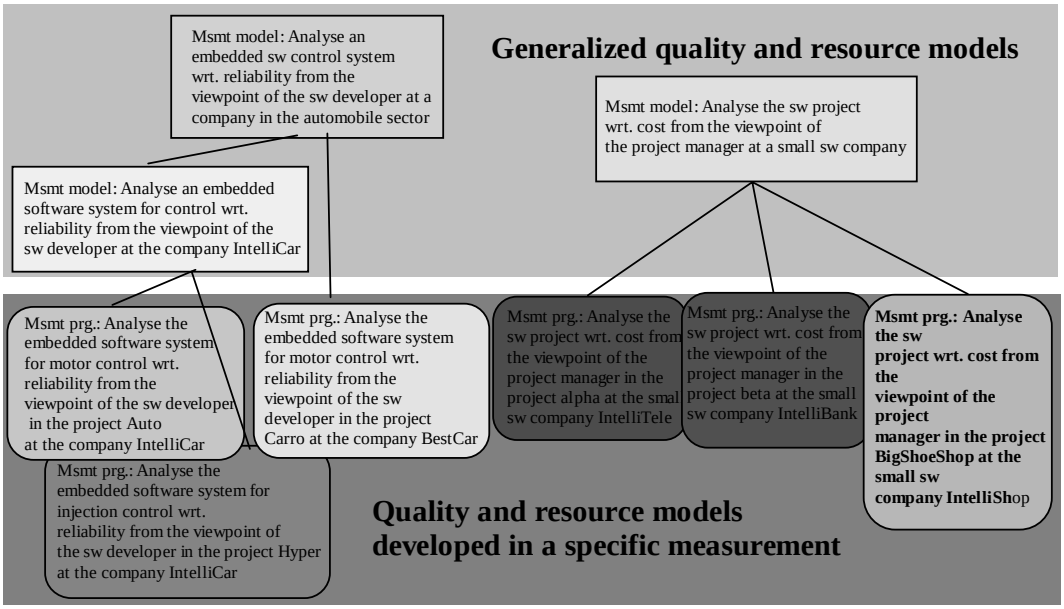

Figure 1 Example of generalization of quality and resource models

Besides reusing concrete experiences that have been gathered in a past measurement program, experiences gathered in similar types of contexts and related to similar measurement goals can be synthesized by developing generalized models. Such generalized quality or resource models are explicit abstract representations of measurement knowledge that capture key variables. The objective of these models is to summarize and communicate complex knowledge for a specific type of context (e.g., an organization, a market sector or a type of company) and does not intend to be "universally" valid. For example, as shown in Figure 2, based on the experiences gathered in various small software companies and standards wrt. the same measurement goal, a generalized resource model in the context of small software companies can be derived by unifying and abstracting relevant aspects from the individual plans.

Today, however, the reuse of quality and resource models often does not result in a satisfactory measurement plan, as, in most cases, reuse is done in an ad-hoc, informal manner, usually limited to personal experiences. To maximize productivity and quality gains, knowledge management and organizational learning with respect to the quality and resource models have to be systematically integrated into the measurement process.

An approach for continuous learning and reuse of experience in the software domain is the *Quality Improvement Paradigm* (QIP) [BCR94] supported by the *Experience Factory* approach (EF) [BCR94a, BR88]. Based on the QIP/EF approach, context-specific quality and resource models can be developed and managed in order to provide systematic support for measurement (see Figure 3) [AG02, Gre02a, PTK02].

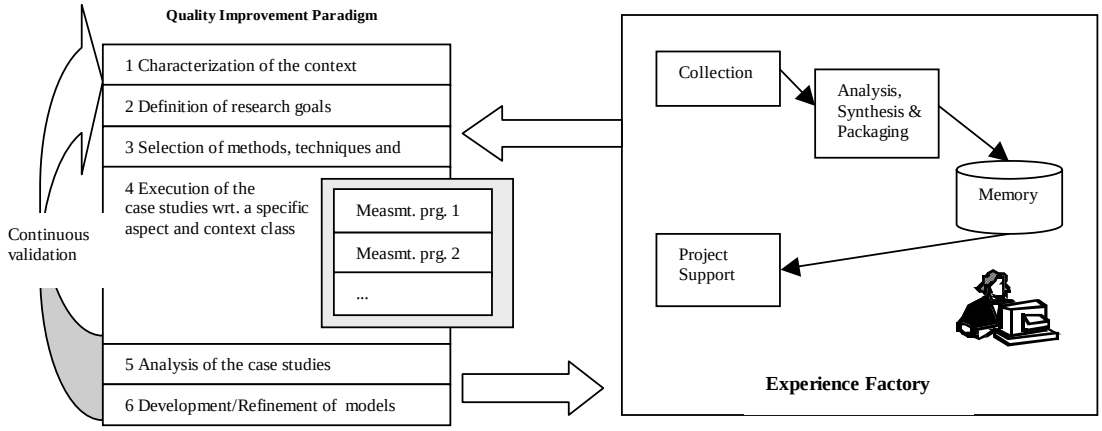

Figure 2 Management of context-specific quality and resource models

The EF organization provides support for the continuous collection of concrete quality and resource models developed in ongoing measurement programs. Those models are analyzed and packaged in order to create a repository of well-specified and organized experiences. This may include the storing of concrete quality and resource models, the cataloguing of a set of models wrt. their measurement goals and contexts or the development of generalized models by synthesizing experiences from individual measurement programs. These stored models then become available for reuse in new measurement programs, where useful quality and resource models are retrieved by identifying models that have been developed in similar contexts.

The usage of a technical infrastructure supporting the management of quality and resource models is meant to facilitate the reuse of useful models and will reduce overhead. However, so far only few first experiences with the application of tool support for this specific type of knowledge management exist (including ES-TAME [Oiv94,OB92] and REMEX [Gre02a]), and further experiences– especially looking at SMEs – are studied.

5 First Experiences in Applying the Approach

In this section, we describe our experiences with respect to reusing quality and resource models and the application of GQM Lightweight in practice.

What was done?
We applied (parts of) the GQM Lightweight approach in five small and medium-sized software companies. This included three start-up companies, each with 2 to 4 employees focusing on the development of information management systems for external commercialization. The other two cases were done at departments (each with 8 to 10 engineers) whose core task was software development. In all cases, the authors of this paper had a coaching/consultant role. They were the GQM experts that provided GQM knowledge and supported the people in the small enterprises in defining a suitable measurement plan.

The Definition phase of GQM Lightweight was conducted by all companies/departments. All three start-up companies focused on the analysis of management aspects and, therefore, the quality model of company A (on project management) was reused for the definition of the measurement programs at companies B and C. The questions and measures that were used to conduct Maintainability measurement for department D were reused from a past measurement program, done previously for another company (same measurement goal and a similar context). Only in Department E there was no explicit reuse of a quality or resource model. Instead, the plan was developed from scratch with the intention of future reuse.

Table 1 presents numbers of goals, questions and metrics per defined measurement plan. The size of those plans is much smaller compared to the measurement plans we generally experience at large companies, namely: ranging from 3 to 7 measurement goals, 10 to 50 questions, 22 to 70, sometimes even 160 measures.

	Quality focus	Number of goals, questions & metrics (G,Q,M)	Phases of GQM Lightweight process addressed[1]	Type of quality/ resource model[2]
Start-up company A	Project management	1, 9, 10	All phases	Quality focus model
Start-up company B	Project management	1, 10, 11	Definition	Quality focus model

[1] See also section 3
[2] See also section 4

Start-up company C	Project management	1, 10, 10	Definition	Quality focus model
Department D	Maintainability	1, 5, 12	Definition	Questions, Measures, Data collection procedures
Department E	Requirements elicitation and management	1, 6, 17	All phases	Questions, Measures

Table 2 Characterizing the application of GQM in SMEs

Criteria for analyzing the approach

The rest of this section reports on the experiences in applying GQM Lightweight and the reuse of quality- and resource models by looking at three criteria, namely:

1. Applicability – of GQM Lightweight and the reuse of quality/resource models in SME an context
2. Reduction in time and effort – to define a new measurement plan
3. Ability (of the people in the SMEs) to acquire measurement expertise

The first criterion is related to one of the problems of introducing software measurement successfully in SMEs, namely that SMEs often do not have structured processes. Instead, GQM Lightweight is a structured process. This criterion is also related to another characteristic of SMEs, multiple role-assignments. Thus the question is whether an extra task (such as measurement) and a structured process can be satisfactorily applied in SMEs. The second criterion is related to the motivation of having GQM Lightweight and reusing quality- and resource models, namely that time and effort should be as low as possible. The third criterion refers to the problems denoted in section 1 that SMEs often do not have resources to allocate dedicated staff and knowledge to start quality initiatives.

Applicability of the approach

For all participating companies and departments a measurement plan could be defined. In those cases where the aim was to reuse quality/resource models (in company B, C and department D), this was possible. The coaching by GQM experts was an enabling factor to start the application of the approach. We think that the companies/department that were consulted by us were not able to do it on their own. On the hand, the coach was needed for the 'tips and tricks' to define a measurement plan. On the other hand, the coach as an outsider is a soundboard for company staff. The coaches were only necessary to start up the measurement program. After a period of a few months the measurement program is lived by the company and people are able to conduct data collection and interpretation

themselves. So far, we did not notice that the defined measurement plan was improved/changed.

The data collection instruments that were applied in our cases were simple and non-automatic. In company A, a paper effort report sheet was used, which was filled out daily by all project personnel. In department D, a checklist was used that was filled in by engineers each time they scored a software module on its maintainability. The checklist was implemented by a spreadsheet, to provide engineers with direct feedback and to facilitate the data collection process. In department E, a mixture of automatic data collection (generated by a workflow tool) and expert assessments was planned. However, this measurement has not started yet because the choice of instruments was too complex to implement it in this department. The lesson learned is that the ambitions to define measurement instruments should not be too high and that starting with simple instruments (like data collection sheets) should be preferred.

The measurement programs were executed at company A and department D. The other companies/department did not yet implement the program, so no experiences on measurement results can be presented for these cases. In department D, a set of eight embedded code modules was assessed with the questionnaire resulting from the measurement program. This enabled the department to select modules that required further development.

At company A, the results provided a quantitative basis for the planning and control of the existing software projects and for planning the new projects. The measurement program also helped the company in getting a better understanding of the software process. The planning of the measurement program required the definition of the principal steps of the development process and resulted in a first description of the software process, which did not exist before. This process definition was validated and corrected adequately during the execution of the measurement program. Having this validated process description supported the company in defining improvement actions, e.g., by developing process guides and templates to facilitate execution.

Reduction of time/effort

The effort/cost for definition was reduced significantly by reusing the models. During the development of the resource model for company B, the model from company A was reused and during the development of the resource model for company C, both models from company A and B were reused by basically only adding or removing certain aspects and adapting them to the specific software

process in place. The effort in the second application was reduced by about 41% and in the third application by about 67%; see figure 4.

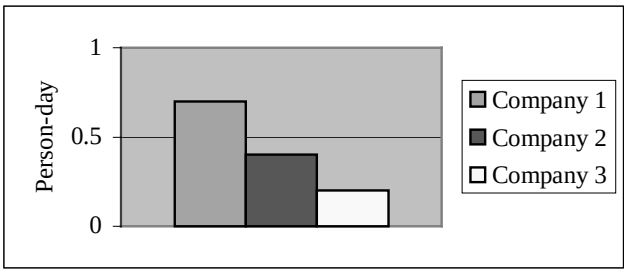

Figure 3 Planning effort reusing concrete models

This significant reduction might be partly due to the fact that the focus was on resource models. There might be less reduction when reusing quality models (e.g., on usability), as they might be more context-specific and, thus, may require more effort for their adaptation to completely fit into the particular context. However, in reusing a quality model for maintainability (at Department D), we observed a reduction in definition effort, too. The original definition cost 6 days; the tuning of this model in department D was done in 3 days.

When other GQM Lightweight phases are also taken into account, it is interesting to observe that the initial establishment of a measurement program focusing on resource aspects in one of the start-up companies using the approach required approximately 38 person-hours in the first 8 months of its application [AGH02]. These numbers are a first indication that GQM Lightweight can reduce the measurement effort to an acceptable amount in the context of small and medium-sized companies, especially in comparison with numbers from other GQM applications where the approximate total effort has been about 1 person-year [CEM96].

Contrary to these positive remarks on reducing effort/cost, we also noticed a great risk in SMEs as their frequent changes of focus also require a re-definition of the measurement program. For example, at company B and C, there was a sudden and complete change in the business focus of the start-up due to the for finding a market niche. As a consequence, the software product and application domain changed completely. Depending on the degree of change, this might require significant extra effort.

Ability to acquire measurement expertise

Measurement know-how was transferred by reusing the quality/resource models as well as by transferring expertise about the GQM Lightweight process. Concerning the models, it was observed that less expertise in measurement was required. However, in order to enable the reuse of these models, they have to be carefully revised in order to prevent the utilization of inappropriate models. Furthermore, a certain measurement experience was necessary in order to appropriately adapt the models to fully satisfy the present situation.

Concerning the process, the coaches provided expertise to one or two representatives of the company/department. Later, people were capable of continuing the measurement program after the coaching was finished. The successful transfer was due to fact that the people had a clear responsibility for this task and that they had time to consult the coaches and read about the GQM process. The fact that the knowledge transfer was directed to one (or two) person(s) also caused the GQM-knowledge to be restricted to these people. Despite the existence of shorter communication channels (which is often regarded as being a characteristic of SMEs), we observed that the knowledge on the measurement programs itself (the questions and metrics) was transferred amongst the people in the departments, but that additional background information on GQM was not communicated explicitly. Also, discussions on, e.g., the type of data collection were not conducted within the company. To conclude, the role of measurement expertise was assigned in a way similar to that in large companies. However, in a large company it is often a GQM team or department that is involved, while in a small company only one person is responsible.

6 Conclusions

In this paper we have argued that small and medium-sized software companies have various problems in adopting quality initiatives due to their specific characteristics and limitations. Therefore, we have proposed a customized approach for software measurement, as one of the key technologies for software process improvement and as a basis for systematic project management. Considering that there exists a trade-off between the application of universal and validated models and the need for context-dependent measurement customized to a specific environment, the *GQM Lightweight* approach presented aims at integrating both approaches by compacting the measurement process based on the GQM approach and integrating the reuse of context-specific quality and resource models.

First results in applying *GQM Lightweight* in practice indicate that the method is suitable for establishing an effective measurement program in SMEs. We have observed that the method and especially the reuse of context-specific models can reduce the effort spent on the planning and execution of measurement programs, keeping it to a level that enables its application in SMEs. However, *GQM Lightweight* should be empirically validated further, especially concerning other quality models, in order to make more general statements about the impact of the approach on software measurement in SMEs.

Acknowledgements

This work has been realized with the support of the CNPq (an entity of the Brazilian Government directed to scientific and technologic development) and the DLR (an entity of the German Ministry of Science and Education).

References

[AG02] A. Anacleto, C. Gresse von Wangenheim. *Applying Measurement in Small Software Companies for Supporting Project management* (in Portuguese), in: Proceedings of Simpósio Brasileiro de Qualidade de Software, Brazil, 2002.

[AGH02] A. Anacleto, C. Gresse von Wangenheim, J. Hammes, *Measurement for Supporting Project Management in a Small Software Company* (in Portuguese), in: Proceedings of the XIII Conferência Internacional de Qualidade de Software, Brazil, 2002.

[Bas 88] V. Basili, D. Rombach, *The TAME Project: Towards improvement-oriented software environments*, in: IEEE Transactions on Software Engineering, vol. SE-14, no. 6, pp. 758-773, 1988.

[Bas92] V. Basili, *Software Modelling 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.

[BCR94] V. Basili, G. Caldiera, D. Rombach, *Experience Factory*, in: J. Marciniak (ed.), Encyclopedia of Software Engineering, vol.1. John Wiley & Sons, 1994.

[BDR96] L.Briand, C. Differding, D. Rombach, *Practical Guidelines for Measurement-Based Process Improvement*, in: Software Process, 2(4), December 1996.

[BMB96] L. Briand, S. Morasco, V. Basili. *Property-Based Software Engineering Measurement*. IEEE Transaction on Software Engineering, vol. 22, no. 1, January 1996.

[BR88] V. Basili, D. Rombach. *The TAME Project: Towards Improvement-Oriented Software Environments*. IEEE Transactions on Software Engineering, SE-14(6), 1988.

[BW84] V. Basili, D. Weiss, *A Methodology for Collecting Valid Software Engineering Data*, in: IEEE Transactions on Software Engineering, SE-10(6):728-738, 1984.

[CD99] G. Cantone, P. Donzelli. *Goal-oriented software measurement models*. Proceedings of the ESCOM, 1999.

[CEM96] The CEMP project. ESSI Project Number 10358. *Customized Establishment of Measurement Programs*, Final Report, July 1996.

[Dif01] C. Differding, *Adaptive Measurement Plans for Software Development*, PhD Thesis, Department of Computer Science, University of Kaiserslautern, 2001.

[GHW95] 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, Germany, 1995.

[GM97] A. Gray, S. MacDonell. *GQM++ A Full Life Cycle Framework for the Development and Implementation of Software Metric Programs*. In Proc. of the Fourth Australian Conference on Software Metrics, Australia, 1997.

[Gre02a] C. Gresse von Wangenheim, *Operationalizing the Reuse of Software Measurement Planning Knowledge*, PhD Thesis, University of Kaiserslautern, Germany. aka Verlagsgesellschaft/infix, DISKI 256, 2002.

[Gre02b] C. Gresse von Wangenheim. *Planning and Executing GQM-Based Software Measurement*. Technical Report LQPS001.01E, UNIVALI, São José, Brazil, 2002.

[KA83] M. Kogure, Y. Akao, *Quality Function Deployment and CWQC in Japan*, Quality Progress, October 1983.

[KP96] B. Kitchenham, S. Pfleeger, *Software quality: the elusive target*, in: IEEE Software, January, pp.12-21, 1996.

[LVC02] O. Laitenberger, S. Vegas, M. Ciolkowski, *The State of the Practice of Review and Inspection Technologies in Germany*, ViSEK Report 011.02, ViSEK Consortium / Fraunhofer IESE, Kaiserslautern, 2002.

[MBB+98] M. Mendonça, V. Basili, I. Bhandari, J. Dawson, *An approach to improving existing measurement frameworks*, IBM System Journal, vol 37 (4), 1998.

[McC 77] J. McCall, P. Richards, G. Walters, *Factors in Software Quality*, RADC TR-77-369, 1977. Vols I,II,III', US Rome Air Development Center Reports NTIS AD/A-049 014, 015, 055, 1977.

[MCT01] Ministério da Ciência e Tecnologia, *Quality and Productivity of the Brasilian Software Sector* (in Portuguese), Ministério da Ciência e Tecnologia, Brazil, 2001.

[Mur80] G. Murine, *Applying Software Quality Metrics in the Requirements Analysis Phase of a Distributive System*, in: Proceedings of the Minnowbrook Workshop, New York 1980.

[OJ97] R. Offen, R. Jeffery, *Establishing Software Measurement Programs*, IEEE Software, March/April 1997.

[PR94] S. Pfleeger, D. Rombach, *Measurement based Process Improvement*, in: IEEE Software 1994.

[PGF96] R. Park, W. Goethert, W. Florac, *Goal-driven Software measurement – A guidebook*, CMU/SE-96-HB-002.

[PTK02] T. Punter, A. Trendowicz, P. Kaiser, *Evaluating Evolutionary Software Systems*, in: Proceedings of PROFES 2002, Rovaniemi (F), December 9-11, Springer LCNS 2559, 2002

[Rom91] D. Rombach, *Practical Benefits of Goal-Oriented Measurement*, in: Software Reliability and Metrics, Elsevier Applied Science, 1991.

[vSB99] R. van Solingen, E. Berghout, *The Goal/Question/Metric method – a practical guide for quality improvement of software development*, London, McGraw-Hill, 1999.

[SEI01] Software Engineering Institute, http://www.sei.cmu.edu/sema/pdf/2002aug.pdf.