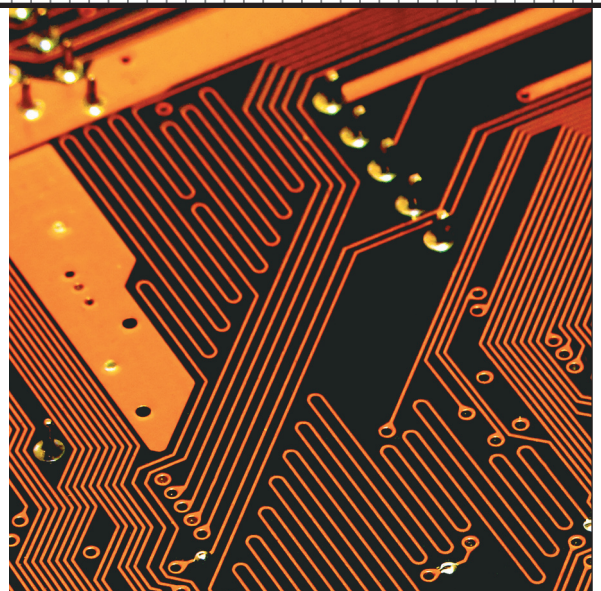


Linking Software Development and Business Strategy through Measurement



- ➔ **Victor R. Basili, Mikael Lindvall, Myrna Regardie, and Carolyn Seaman,**
Fraunhofer Center for Experimental Software Engineering
- ➔ **Jens Heidrich, Jürgen Münch, Dieter Rombach, and Adam Trendowicz,**
Fraunhofer Institute for Experimental Software Engineering

The GQM*Strategies approach extends the goal/question/metric paradigm for measuring the success or failure of goals and strategies, adding enterprise-wide support for determining action on the basis of measurement results. An organization can thus integrate its measurement program across all levels.

Most organizations that develop software try to retain their competitive edge by reducing software-related risks. Effective risk management requires aligning business goals with development strategies and translating the results into a quantitative project management plan. In addition, organizations must also justify cost and resources for software and system development and other IT services. Often, such justification demands a concrete demonstration of how such development will contribute to the organization's overall business goals—the top-level goals the organization desires to accomplish.

Working against both alignment and justification is the lack of methods for linking business goals and software-related efforts. Without a deep understanding of how software development fits into their organization's business objectives, decision makers can neither plan development nor evaluate the organization-wide success of development strategies.

To address this need, we have developed GQM*Strategies, which uses measurement to link goals and strategies on all organizational levels. The approach is based on rationales for deciding when and how to transform goals into operations and how to evaluate the success of strategies with respect to those goals. The “Why Business Alignment?” sidebar describes our motivation for developing this approach.

As the name implies, GQM*Strategies is based on the goal/question/metric (GQM) paradigm¹ but adds the ability to create measurement programs that ensure alignment between goals and strategies from the highest strategic business levels to individual development projects and vice versa.² Although we derived the approach from software development experiences, it is not necessarily applicable solely to this domain.

GOAL-ORIENTED APPROACHES

Software development organizations often encounter problems when instituting measurement programs, such

WHY BUSINESS ALIGNMENT?

With the increasing dependence on software and other information technology (IT) has come a commensurate growth in software system size and complexity. This growth in turn has magnified the cost, schedule, and quality concerns that have always plagued software development. For decades, software engineering researchers and practitioners have attempted to reduce development cost, shorten development time, and increase quality. Although great strides have been made in all three areas, software growth along all dimensions (size, complexity, pervasiveness, criticality, and so on) has outpaced the ability to control all the development-related factors.

Efforts to define and understand these relationships have made one notion crystal clear: Issues related to software cost, schedule, and quality are inextricably linked with the larger issues that face the businesses developing the software—regardless of the specific business objectives. Some businesses sell what they develop, either as contracted custom software or as commercial shrink-wrapped applications. Others sell a product or service, of which software is a significant component. Still others might develop software only to support their organizational IT infrastructure so that they can develop and market products more effectively. Finally, some have no commercial interest at all, such as nonprofit organizations, government entities, or educational institutions.

Although all these organizational configurations challenge their development projects in different ways, the point is that all have a larger business context, which encompasses larger business goals, strategies, and success measures. All software development projects must consider context factors—the variables that represent the organizational environment and affect the kind of models and data the project can use.

The problem is not that businesses fail to achieve their objectives, but rather that they do not always state these objectives explicitly or clearly enough to verify that they have indeed achieved those objectives. It is even less clear how a business translates its objectives into its lower organizational levels and into individual projects. At present, no methodology bridges the gap between business strategies and their project-level implementation.

Quantitative data is a prerequisite to understanding the relationships between the business and project-level goals and verifying the achievement of objectives—one reason that software development maturity requires the use of quantitative measures. However, popular software improvement strategies, such as Capability Maturity Model Integration and the IT Infrastructure Library, are not directly and explicitly linked to business value. Consequently, the investment in collecting data does not result in the expected benefits, and the contribution of project performance to the achievement of strategic goals remains unclear.

as collecting too much data, failing to gather useful or correct data, and not having a mechanism for analyzing the data in a way that contributes to sound strategic decisions. This leads to numerous problems, including decreased cost-effectiveness of the measurement program and disillusionment about metrics on the part of developers and managers. The end result is often the eventual failure of the measurement program as a whole.

To address these problems, researchers have developed *goal-oriented approaches*—so called because they use goals, objectives, strategies, or other mechanisms to guide the choice of data to be collected and systematically analyzed. The GQM approach, for example, provides a method for an organization or a project to define goals, refine those goals into specifications of the data to be collected, and then analyze and interpret the resulting data in light of the original goals. Implicit in the GQM approach is the use of interpretation models, which help practitioners interpret the resulting data in a specific context. The GQM approach has served the software industry for several decades in defining measurement programs, but it does not explicitly provide support for motivating and integrating the measurement of goals at various organizational levels, such as project goals, business objectives, and corporate strategies. It also does not encourage users to explicitly document *assumptions*, estimated unknowns that can affect data interpretation.

Another goal-oriented approach, Balanced Scorecard (BSC),³ links strategic objectives and measures through a scorecard, which consists of four perspectives: financial, customer, internal business processes, and learning and growth. BSC does not dictate a static set of measures, but rather serves as a framework for strategic measurement and management. As such, it is a tool for defining strategic goals from multiple perspectives beyond a purely financial focus, but it does not address lower-level goal setting.

Practical Software Measurement (PSM)⁴ offers detailed guidance on software measurement, including providing a catalog of specific measures and information on how an organization can apply those measures in projects. PSM also includes a process for choosing measures according to the issues and objectives relevant to a software development project. For the most part, however, approaches such as BSC and PSM do not support the alignment of objectives at different organizational levels, or this is simply not the focus of these approaches.

Researchers have also proposed various combinations of GQM, BSC, and PSM⁵⁻⁸ that recognize the need to link higher- and lower-level goals. However, because these approaches do not support different goals at explicitly linked organizational levels, it is difficult to feed analytical results and interpretations back up the chain. In contrast, GQM+Strategies creates mappings between the data related to goals at different levels so that insights gained relative to one goal at one level—whether project or departmental, or a business objective—can support and contribute to satisfying goals at other organizational levels without requiring that each level share the same goals.

Some application domains have specific requirements that emphasize the need to link business goals to lower-level properties. Organizations are becoming increasingly aware, for example, that the IT infrastructure itself im-

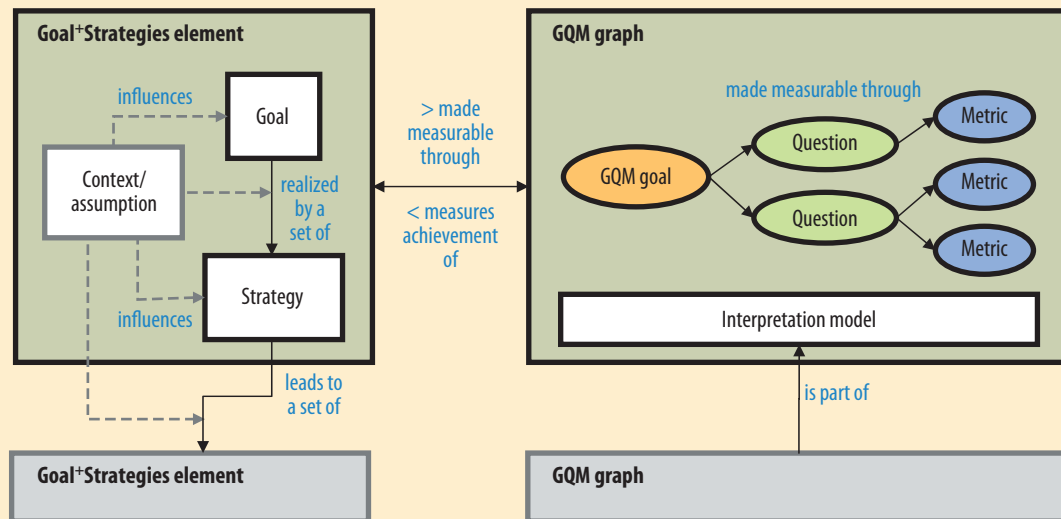


Figure 1. GQM+Strategies components. The primary components are the Goal+Strategies element and the GQM graph. The Goal+Strategies element includes a single goal and derived strategies, as well as all context information and assumptions that explain how goals and strategies link. The GQM graph reflects a single GQM goal, the corresponding questions and metrics, and an interpretation model.

poses significant risks, which has led to several regulatory constraints in both the IT governance and IT service domains, such as the Sarbanes-Oxley Act in the US.⁹

The solutions proposed by models in these domains offer connections between predefined sets of goals and attributes of the IT infrastructure. They are sharply focused on a particular domain, such as IT governance, and define a very detailed model that fits that domain quite well. To apply such an approach, however, the organization must strictly follow the prescribed model. Release 4 of the Control Objectives for Information and Related Technology (CoBIT) standard (www.isaca.org), for example, is based on a process model that subdivides IT into four domains and 34 processes.

Approaches such as CoBIT and the IT Infrastructure Library (ITIL) provide no mechanism for adapting and tailoring the solution, nor do they support addressing the specific context or documenting inherent assumptions. No clearly defined interpretation model indicates if an overall strategy is working or has to be changed to avoid business failure.

COMPONENTS OF AN INTEGRATED APPROACH

GQM+Strategies aims to address the weaknesses of existing goal-oriented approaches by providing both explicit links among organizational levels and the flexibility to tailor the approach to the organization's specific needs and objectives. Eight conceptual components form the basis for constructing a consistent model. Figure 1 illustrates how these components interrelate.

- **Business goals:** goals the organization wishes to accomplish in general to achieve its strategic objectives.
- **Context factors:** environmental variables that represent the organizational environment and affect the kind of models and data that can be used.
- **Assumptions:** estimated unknowns that can affect the interpretation of the data.
- **Strategies:** a set of possible approaches for achieving a goal that may be refined by a set of concrete activities.
- **Level *i* goals:** a set of lower-level goals inherited from the previous level's (*i* – 1) goals as part of the previous level's goal strategy—an example is a goal related to a project that is part of a selected software strategy.
- **Interpretation models:** models that help interpret data to determine if the goals at all levels have been achieved.
- **Goal+Strategies element:** a single goal and derived strategies (including a set of concrete activities), as well as all context information and assumptions that explain how the goal and corresponding strategies are linked.
- **GQM graph:** a single GQM goal that measures a Goal+Strategies element and its corresponding questions, metrics, and interpretation models.

All organizations have business goals, such as improving customer satisfaction, garnering market share, and reducing production costs, and they devise strategies to deal with them, taking into account the context and making explicit any assumptions. Strategies specify how to achieve the goal and should, in turn, define lower-level

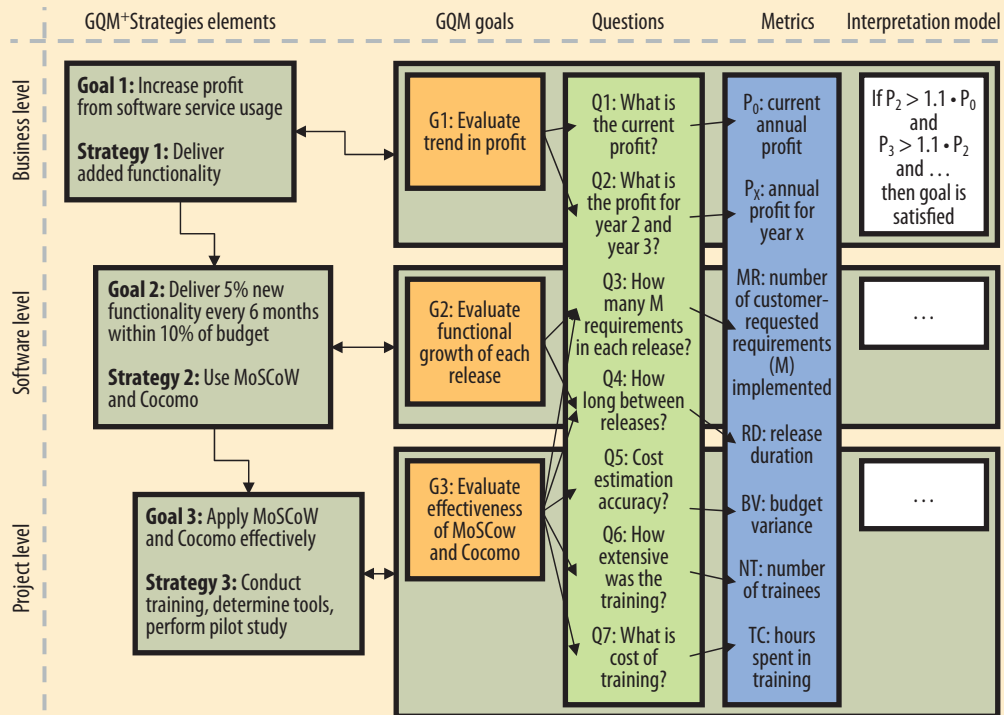


Figure 2. A GQM*Strategies model for ABC, a company that provides its customers with information services through the Web. GQM*Strategies enforces the explicit documentation of the relevant context factors and assumptions that are necessary for understanding and evaluating each goal. A Goal*Strategies element consists of a goal and an associated strategy (bottom of each goal box). Each element, in turn, is associated with a GQM graph (green rectangle to the right of the goal) representing questions and metrics as well as an interpretation model that evaluates if the goal was achieved.

goals for various parts of the organization, such as software, hardware, and marketing. An organization might devise a separate set of strategies to deal with these lower-level goals. The number of goal and strategy levels depends on the organization's internal structure.

Assumptions and context information about the organization strongly influence the definition of goals and strategies. Measuring the effectiveness and the accomplishment of goals and the effectiveness of strategies at all levels is critical for business success. And making goals connected and transparent at all levels helps in communicating and achieving them. What most organizations need is support for these concepts.

The components in Figure 1 provide this support by allowing multiple goal levels and multiple strategies for each goal (a GQM*Strategies element), as well as a measurement plan (GQM graph). The GQM*Strategies element provides the basic components for deriving the goals and strategies as influenced by the context; the basic known facts about the market, the organization, the product, and so on; and the assumptions—factors that are assumed or guessed but not known for sure. A key element is making the context and assumptions explicit so that they can be verified. The relevant context and assumptions aid in defining the rationale for choosing specific goals and strategies. A set

of predefined goals and strategies can become part of an organization-specific experience base and allow different parts of the organization to select and adapt predefined goals and strategies.

Associated with each GQM*Strategies element is a measurement plan that uses the GQM approach measurement and evaluation framework to specify how to evaluate the goal, what data to collect, and how to interpret that data. The nodes of each GQM graph consist of a measurement goal, which describes what knowledge needs to be gained from the measurement activity; a set of questions to be answered; the metrics and data items required to answer the questions; and an interpretation model that specifies how the data items are to be combined and what the criteria are for determining the goal's success.

The nodes are related semihierarchically. A goal can have several associated measurement goals, each of which is the basis for an entire GQM graph. However, different GQM structures are likely to use some of the same questions and metrics, and interpretation models might combine data from different GQM structures, thus optimizing metrics collection. The results of the lower-level interpretation models feed into the higher-level ones to provide feedback on lower-level goal achievement.

With the complete model, an organization can not only define measurement consistent with larger, upper-level organizational concerns, but also interpret and roll up the resulting measurement data at each level.

AN APPLICATION

To illustrate the features of our approach, we consider a fictitious but representative type of organization, ABC, which provides information services to its customers through the Web. Customers do not buy the software, but rather pay a service fee for access to information and to software that searches, analyzes, and presents that information. Thus, the business model implies that the number of customer accesses to ABC's system determines the company's revenue. Figure 2 represents a model of this application.

Business goals

Although the approach is flexible enough to accommodate several different entry points, the starting point of the GQM*Strategies process in this example is a business goal. As Goal 1 shows, one of ABC's business goals is to increase profit from software service use. The GQM*Strategies approach enforces the explicit documentation of the relevant context factors and assumptions necessary for understanding and evaluating each goal. For Goal 1, one such context factor is that how often customers access ABC software products determines the amount of revenue generated at ABC.

The GQM*Strategies goal template documents other details. Each template includes the desired magnitude of improvement, the time frame for achieving the goal, the scope of responsibility for achieving the goal, and any constraints or conflicting goals as well as relationships to other goals. Figure 3 shows a partial template for Goal 1.

The GQM graph in Figure 2 is based on the measurement goal, G1. In full GQM notation, G1 is

Analyze the trend in profit for the purpose of evaluation with respect to a 10 percent increase in annual income per year from the point of view of ABC's management in the context of the ABC organization.

This goal leads to questions Q1 and Q2: What is the current profit (measured by P_0), and what is the profit for each succeeding year (as measured by P_x)? The decision maker analyzes the results using the criteria incorporated in the interpretation model (far right), which says that

Starting in year 2, if the profit for the current year (P_2) is at least 10 percent (1.1 times) higher than the profit for the initial preceding year (P_1), then the goal has been satisfied.

Activity:	Increase
Focus:	Net income
Object:	ABC Web services
Magnitude:	10% per year
Time frame:	Annually, beginning in 2 years
Scope:	Development groups assessed at CMMI level 2 or higher
Constraints:	Available resources, ability to sustain CMMI levels
Relations:	CMMI-related goals

Figure 3. Business goal "Increase profit from software service usage" expressed in the GQM*Strategies goal template. Business goals are at the top level of the GQM*Strategies model. They become measurable through the derivation of one or more GQM graphs, which are then associated with other GQM graphs at other organizational levels.

The full interpretation model, not shown for simplicity, includes an "else" part related to the effectiveness of the chosen strategies.

The strategies associated with each goal come from a list of potential strategies that the GQM*Strategies user has enumerated, taking into account influencing context factors. Possible strategies for meeting Goal 1 are to deliver added capabilities to encourage heavier system use, increase rates charged to customers, or reduce development costs. The user opted for the strategy of delivering added functionality in the product releases at regular and frequent intervals. At this point, the user must also make explicit the assumption that this added functionality will lead to increased customer satisfaction, which will in turn lead to heavier use.

The assumption is that ABC has enough projects with a Capability Maturity Model Integration maturity level greater than 1 that, if just those projects provide a 15 percent improvement, ABC will see a 10 percent improvement overall. In general, all context information and assumptions are attached to goals, strategies, and their relations (as Figure 1 shows) and are documented in a corresponding graph (not shown).

Software-development goals

At the next level down in the model is a goal derived from the strategy (or strategies) chosen at the top level. Goal 2 is to deliver a new release of the software every six months that incorporates at least 5 percent more functionality than the previous release and to keep the cost of this increased functionality to within 10 percent of the current budget.

These are really two interrelated but separate software goals, so the user defines them in separate GQM*Strategies goal templates. Figure 4 elaborates the first of these goals (increasing functionality by 5 percent). At this level, the goal is specific to software development, so it is treated as

Activity:	Deliver
Focus:	More usable functionality, for example, M (must) type requirements from the backlog of customer-requested requirements
Object:	Each release of ABC Web services software
Magnitude:	5% more functionality than the prior release
Time frame:	Every 6 months, beginning in 2 years
Scope:	Web services development projects with CMMI level 2 or higher
Constraints:	Available resources, ability to sustain CMMI levels, ability to estimate cost, and schedule for a release
Relations:	Achievement of cost and schedule estimate accuracy, ability to improve CMMI levels of development groups

Figure 4. Software goal template for the goal “Deliver 5 percent new functionality every six months.” A separate but related software goal, which would have its own template, would address the 10 percent cost variance. Both these goals are derived from a higher-level strategy that addresses the business goal of increasing revenue from increased use of ABC’s software products.

```

for x = 2, 3, ...
  if  $P_x \geq 1.1 * P_{x-1}$  then
    the goal has been satisfied,
  else if functionality was increased appropriately, then
    either some assumption is incorrect or we have chosen the wrong level of strategy.

```

Figure 5. Refining the interpretation model from Figure 2 by adding an “else” part. Full interpretation—the exact reason that a particular goal was not satisfied—depends on goals at lower levels. This iterative refinement ensures that goals at all levels are linked and makes goal interrelationships explicit.

a goal at the software level, as opposed to one at the business level. In general, the name accorded the lower levels depends on the organization applying GQM*Strategies and the number of levels to be modeled.

The software goal template asks for the same categories of information as the business goal template in Figure 3. The user defines the measurement and interpretation model for this goal and refines the interpretation model for the business goal. As Figure 5 shows, the refinement of Figure 2’s interpretation model involves adding an “else” part that recognizes that full interpretation depends on the lower-level goals. For example, if the functionality was not increased by 5 percent, perhaps the strategy associated with Goal 2 was not effective. The interpretation model becomes increasingly detailed, with more conditional logic, at each lower level in the overall GQM*Strategies model.

The organization must decide to develop and carry out a strategy for accomplishing a software goal. The strategy for Goal 2 in Figure 2 is to adopt an approach such as rating the importance of different requirements, as in the MoSCoW prioritization approach¹⁰ for requirements and release planning, and to adopt the Constructive Cost Model (Cocomo)¹¹ for cost estimation. The user must explicitly document an

important context factor relevant to this strategy: Someone with expertise in the MoSCoW approach (a consultant who recommends it) is available, but ABC has no staff with such experience.

Three assumptions relevant to this strategy, which the user must also document, follow:

- The organization can estimate the percentage of functionality delivered, for example, by using a proxy such as additional lines of code delivered, number of function points delivered, or a formula based on an actual requirements count.
- The difficulty and importance of requirements are weighted in some way (hard, medium, easy) to provide input to the cost model.
- The backlog of customer-requested requirements continues to grow.

As with the business goal, the software goal has an associated set of GQM structures that define how to evaluate the goal (G2 in Figure 2). In full GQM notation, G2 is

Analyze each six-month release for the purpose of evaluation with respect to incorporation of 5 percent new functionality as compared to the previous release from the point of view of the Web services project manager in the context of the ABC organization.

This goal leads to questions Q3 and Q4: How many new requirements of must (M) importance were included in each release? How much time elapsed between releases? The interpretation of this software goal’s achievement is

If, at each six-month milestone, the growth in functionality of a release is 5 percent, then the level 2 goal is satisfied for this release; else, assumptions about MoSCoW are incorrect or we have not chosen the correct strategy.

At this point, the user can also further refine the business goal’s interpretation. If the business goal is satisfied (ABC’s profit increased by 10 percent), but Goal 2 is not, then the assumptions are wrong. The delivery of a particular requirement alone might have caused the gain, for example.

Project-specific goals

The last goal level in Figure 2 consists of goals derived from the previous level’s strategy that apply to a particular software development group or project. In the sample application, Goal 3 is to apply the MoSCoW and Cocomo approaches effectively. The relevant development group

has developed a strategy for this goal that involves training personnel, acquiring tools that will assist in applying these methods, and piloting these methods on a single project. A relevant assumption is that training for these approaches can be targeted to a few specific individuals, so the impact of the training on cost and schedule is reasonable.

The GQM graph at this level, only part of which is shown in Figure 2, involves evaluating the effectiveness of using MoSCoW and Cocomo and of the training and other tools. The GQM+Strategies user then uses these results to revisit the interpretation of higher-level goals. Some of the questions and metrics defined at the second level are reusable at subsequent levels—a benefit of basing GQM+Strategies on the GQM approach.

SUPPORTING STRATEGIC MEASUREMENT

In all cases, specifying a strategy for goal achievement creates explicit links among goals at the various levels, from business objectives to project operations, which is critical to strategic measurement. In the example, the highest-level goal was to increase profit, the next-level goal was to provide new functionality in short releases, and the lowest-level goal was to use specific application methods to achieve the previous level goals. The linking strategy specified that ABC will achieve its increased profit goal by providing customers with more functionality. Often such links are implicit, but making them explicit has many benefits.

Templates

Templates also support strategic measurement by guiding users in defining all goal types at the necessary detail level. In the example, the full template for the business goal partially described in Figure 3 includes more extensive information about the target increase in income, the time frame, and any constraints or conflicting goals. GQM+Strategies includes templates for all goal types in the measurement model.

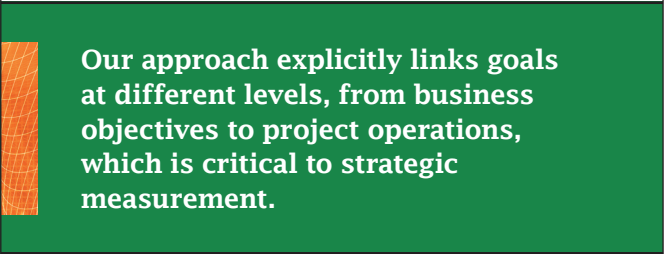
Tracking assumptions

Although the ABC example did not show it, our approach also builds in the ability to track context factors and assumptions at each level (through the requirement to document them along with all strategy documentation) so that users can more easily assess the effects of changes in either context or assumptions. In the example, the approach required that users document the assumption about the training required for MoSCoW and Cocomo so that, if the assumption turns out to be false, the model will indicate what elements the assumption affects and, consequently, what requires reevaluating.

Interpretation models

Interpretation models tie together measurement goals, context factors, assumptions, and data, thus making it

easier to correctly and usefully interpret measurement results. This idea stems from the original GQM approach, but we have broadened it to allow interpretation models at each level to inform not only that level, but also higher levels as the data is aggregated and rolled up. In the ABC example, the results of applying the interpretation model at the lowest level will yield information about how the piloting of MoSCoW and Cocomo went, as well as information about the training and tools. The information from the lower level can help diagnose any problems encountered at the next level up. Further, the results of the interpretation at the second level can inform the analysis at the top level. At the top level, if profit does not increase as expected, the analytical results from the second level will help determine if the problem is due to higher costs, inadequate functionality delivered, late releases, or some other cause.



Our approach explicitly links goals at different levels, from business objectives to project operations, which is critical to strategic measurement.

Flexibility

Our approach is flexible enough to accommodate other software measurement approaches and business strategy approaches. ABC might have used BSC to define its business goals and strategies and then used that description as a starting point for applying GQM+Strategies. PSM might have been useful later in applying our approach, when ABC needed to define the metrics.

Shared measurement planning and experience

Because all goal levels are linked, measurement planning and results are organization-wide rather than limited to a single project or department. Often the results are lower costs and better return on the investment in a measurement program, greater likelihood that the program will succeed, more effective risk identification and management, and greater compliance with software process improvement models such as CMMI.

A substantial benefit is that sharing makes it more feasible to build a measurement experience base that can become a valuable corporate asset. Such a base can make project measurement and planning easier over time and consequently decrease project costs. It might begin with the set of generic experiences already available in GQM+Strategies and add value as the organization populates it with organization-specific models.

Table 1. Ongoing applications of the GQM+Strategies approach.

Business	Domain	Application
European telecommunications company	Telecommunications	Drive strategic improvement programs, support paradigm shift toward purpose-driven metrics
European automotive supplier	Automotive	Support CMMI's Measurement and Analysis process area
European network testing company	Telecommunications	Evaluate cost, benefit, and schedule for modernizing existing product suite
International software company	Embedded systems used in telecommunications	Increase the visibility at all organizational levels of how strategic decisions impact operations
Asian insurance company	Information systems	Align strategies and goals for new business domain
Asian systems engineering organization	Safety-critical software for aerospace application	Increase visibility of goals and strategies and derived measurement goals to enhance supplier collaboration
Joint research project to develop a common software platform	Support of complex, dynamic business processes in a variety of domains, including logistics, retail, and customized industrial facilities	Align project objectives and business objectives of involved research and industry partners

Any part of the ABC model in Figure 2 is reusable. The assumptions and context factors are particularly important because they capture the properties that might not hold when the organization reuses the model for a different situation. However, because the model requires the explicit capture of assumptions and context factors and links these to particular goals and strategies, the organization can clearly see which parts of the model need to be reevaluated when an assumption or context factor changes. The result is the flexibility to adapt goals and strategies to particular market needs and to analyze the consequences to the organization.

APPLICATION SCENARIOS

We have already taken steps to apply the GQM+Strategies approach in various industrial settings. Although our approach focuses on software development, it is finding equal utility in broader contexts, as the applications in Table 1 illustrate.

A few studies are ongoing to get more insights into the deployment of GQM+Strategies. So far, most of the studies have reached the phase of initially setting up a GQM+Strategies grid, but they have not yet progressed to the point of using and maintaining it.

The GQM+Strategies approach provides many features for organizations that want to create a software measurement program that is consistent with and contributes to the achievement of goals at all organizational levels. Our sample application illustrates the use of GQM+Strategies in a single context with three goal levels. Other situations might call for additional levels, but generally the process of capturing multilevel goals includes a business goal level, a set of lower-level goals, and a project-specific goal level. Each of these levels, in turn, can have multiple peer goals. Regardless of how many levels the organization

requires, all levels must be based on well-defined goals, each with an associated strategy, documented context factors and assumptions, and a measurement and evaluation framework.

The most important benefit of applying GQM+Strategies is the resulting transparency of measurement motivations and goals at different organizational levels, which makes it easier to identify goal relationships and conflicts and facilitates communication for organizational segments, such as marketing and software development. In the ABC example, without the GQM+Strategies model, project personnel might have misdirected their training efforts or chosen the wrong tools. The model helped project personnel understand why they were being asked to implement MoSCoW and Cocomo, which was likely to produce more focused training or tailoring. This is one of the many benefits of using our approach in a short development cycle. **E**

Acknowledgments

GQM+Strategies is registered trademark No. 302008021763 at the German Patent and Trade Mark Office; international registration number IR992843.

References

1. V. Basili, G. Caldiera, and D. Rombach, "Goal, Question, Metric Paradigm," *Encyclopedia of Software Engineering*, vol. 1, J.J. Marciniak, ed., John Wiley & Sons, 1994, pp. 528-532.
2. V. Basili et al., "Bridging the Gap between Business Strategy and Software Development," *Proc. Int'l Conf. Information Systems*, Association for Information Systems Electronic Library; <http://aisel.aisnet.org/icis2007/25>.
3. R. Kaplan and D. Norton, "The Balanced Scorecard—Measures That Drive Performance," *Harvard Business Rev.*, Jan.-Feb. 1992, p. 71.
4. US Dept. of Defense and US Army, *Practical Software and Systems Measurement: A Foundation for Objective Project Management*, v. 4.0c, Mar. 2003; www.psmsc.com.

5. S.A. Becker and M.L. Bostelman, "Aligning Strategic and Project Measurement Systems," *IEEE Software*, May/June 1999, pp. 46-51.
6. L. Buglione and A. Abran, "Balanced Scorecards and GQM: What Are the Differences?" *Proc. 3rd European Software Measurement Conf.*, Federation of European Software Measurement Assoc., 2000, pp. 18-20.
7. A.J. Bianchi, "Management Indicators Model to Evaluate Performance of IT Organizations," *Proc. Int'l Conf. Management of Engineering and Technology*, vol. 2, IEEE Press, 2001, pp. 217-229.
8. D. Card, "Integrating Practical Software Measurement and the Balanced Scorecard," *Proc. 27th Ann. Int'l Computer Software and Applications Conf.*, IEEE CS Press, 2003, p. 362.
9. Sarbanes-Oxley Act of 2002, Public Law No. 107-204, 116 Stat. 745, codified in sections of 11, 15, 18, 28, and 29 in US Code, 30 July 2002.
10. D. Clegg and R. Barker, *Case Method Fast-Track: A RAD Approach*, Addison-Wesley Professional, 1994.
11. B.W. Boehm et al., *Software Cost Estimation with COCOMO II*, Prentice Hall, 2000.

Victor R. Basili is a professor in the Department of Computer Science at the University of Maryland and chief scientist of the Fraunhofer Center for Experimental Software Engineering. His research interests include measuring, evaluating, and improving the software process and product via empirical studies. Basili received a PhD in computer science from the University of Texas. Contact him at basili@fc-md.umd.edu.

Mikael Lindvall is a senior scientist and division director at the Fraunhofer Center for Experimental Software Engineering. His research interests include agile methods, software process improvement, software architectures, and experience and knowledge management. Lindvall received a PhD in computer science from Linköping University, Sweden. Contact him at mikli@fc-md.umd.edu.

Myrna Regardie is a senior engineer at the Fraunhofer Center for Experimental Software Engineering. Her research interests include software process improvement, measurement, and project and knowledge management. Regardie received a BS in mathematics from Juniata College, Huntingdon, Pennsylvania. Contact her at mregardie@fc-md.umd.edu.

Carolyn Seaman is an associate professor of information systems at the University of Maryland, Baltimore County, and a scientist at the Fraunhofer Center for Experimental Software Engineering. Her research interests include software evolution, management of software projects, and qualitative research methods in software engineering. Seaman received a PhD in computer science from the University of Maryland. Contact her at cseaman@fc-md.umd.edu.

Jens Heidrich is a department head at the Fraunhofer Institute for Experimental Software Engineering. His research interests include project management, quality assurance, and measurement. Heidrich received a PhD in computer science from the University of Kaiserslautern, Germany. Contact him at jens.heidrich@iese.fraunhofer.de.

Jürgen Münch is division manager for quality management at the Fraunhofer Institute for Experimental Software Engineering. His research interests include quality assurance, process management, and measurement. Münch received a PhD in computer science from the University of Kaiserslautern. Contact him at juergen.muench@iese.fraunhofer.de.

Dieter Rombach is a professor and chair of the Software Engineering Department at the University of Kaiserslautern and executive director of the Fraunhofer Institute for Experimental Software Engineering. His research interests include software methodologies, modeling and measurement of the software process and resulting products, software reuse, and distributed systems. Rombach received a PhD in computer science from the University of Kaiserslautern. Contact him at dieter.rombach@iese.fraunhofer.de.

Adam Trendowicz is a researcher at the Fraunhofer Institute for Experimental Software Engineering. His research interests include software cost modeling, measurement, and process improvement. Trendowicz received a PhD in computer science from the University of Kaiserslautern. Contact him at adam.trendowicz@iese.fraunhofer.de.



Selected CS articles and columns are available for free at <http://ComputingNow.computer.org>.

Join the
IEEE
Computer Society

www.computer.org