Spago4Q and the QEST *n*D Model: an Open Source Solution for Software Performance Measurement

Claudio A. Ardagna[1], Ernesto Damiani[1], Fulvio Frati[1], Sergio Oltolina[2], Mauro Regoli[1], and Gabriele Ruffatti[2]

1 Dipartimento di Tecnologie dell'Informazione
Università degli Studi di Milano
via Bramante, 65 – 26013 Crema (CR)
{claudio.ardagna,ernesto.damiani,fulvio.frati}@unimi.it
mregoli@crema.unimi.it

2 Engineering Ingegneria Informatica
Via San Martino della Battaglia, 56 – 00185 Roma, Italy
(sergio.oltolina, gabriele.ruffatti}@eng.it

Abstract. Improving the software development process requires tools and model of increasing complexity, capable of satisfying project managers' and analyzers' needs. In that paper we present a solution integrating a formalized and established model for performance evaluation like QEST *n*D, and an open source Business Intelligence platform like Spago4Q. We obtain a new environment that can produce immediate snapshots of projects' status without any constraint on the number of projects and the type of development process.

Keywords: QEST *n*D, Spago4Q, Performance Indicator, process monitoring

1 Introduction

The availability of detailed and updated information on the software development process is of paramount importance for organizations to maintain their competitiveness level and operate in new and more challenging markets. Such a scenario of integrated information is known as *Business Intelligence* and encloses all the business processes and tools used by organizations for data acquisition.

Within this context, a number of structured process models are adopted by enterprises, depending on their application domain and size, to collect specific knowledge about their development processes, strengthening, at the same time, their *know-how* in terms of more efficiency and quality.

In this paper, we describe our experience in the deployment of an integrated and complete environment for software performance evaluation. To this aim, we exploit a formal model for process performance evaluation (QEST *n*D) and we connect it with

an open source Business Intelligence application (Spago4Q). In particular, the QEST *n*D model (**Q**uality factor + **E**conomic, **S**ocial, and **T**echnological **D**imension) is a multi-dimensional model, proposed in [1] and [3], for the performance evaluation of software processes. Its multi-dimensional nature is based on three important concepts:

1. A number of measurable concepts derived from different business areas (called *dimensions*) including economic, social, and technological ones.
2. The number of business areas interested by the analysis. This number may change for each single project, without any limit (from here comes the acronym *n*D – *n* Dimensions).
3. Organizations are allowed to choose the dimensions of each analysis with respect to their needs.

Such a philosophy makes QEST *n*D an open model, decoupled from any specific development process, allowing multi-process, multi-project performance analysis.

The final objective of the model is to express the overall process performance (*P*) as a combination of the performance of any considered dimension, calculated as the weighted sum of the applied metrics. The global performance value approach gives an immediate and accurate snapshot of the current state of the project, and allows a top-down analysis starting from the global value, which includes all the single measurements, to the analysis of the performance of the single dimension. The performance indicator is calculated by the integration of instrumental measurements (called *RP – Rough Productivity*) and the subjective perception of the overall quality (express as *QF – Quality Factor*).

A problem characterizing the QEST *n*D model and preventing its diffusion in the Business Intelligence context was the lack of reliable and flexible environments where the model could be implemented and distributed. As described in the following sections, we deliver QEST *n*D on an open source business intelligence platform, Spago4Q (SpagoBI for Quality) [2][5]. Spago4Q is a platform for maturity assessment, effectiveness of development software processes and application services, and quality inspection of the released software, achieved by the evaluation of data and measures collected from the process management and development tools with non-invasive techniques.

The tool is easily adaptable to different organizational contexts, independently from the development process adopted by the single projects (i.e. waterfall, XP, Scrum, etc.), meeting exactly the multi-process multi-project approach of QEST *n*D. Although the initial vision of Spago4Q was focused on the software development process, the implementation of the QEST *n*D model, and consequently of a global multi-dimensional performance value, could extend the performance and quality evaluation to services and business areas that are typical of software organizations.

The paper is organized as follows. Section 2 describes more in detail the two frameworks (QEST *n*D and Spago4Q). Section 3 shows the steps to build the implemented integration. Section 4 presents two real case studies. Finally, Section 5 contains our conclusions.

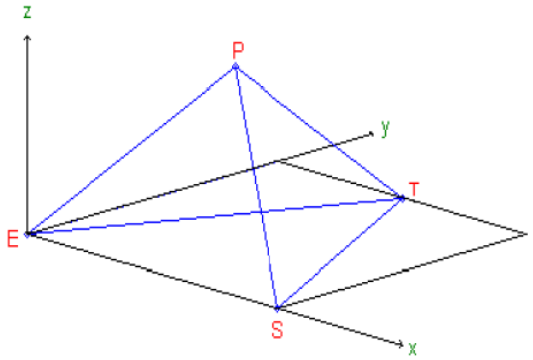

Fig. 1. Geometrical representation of QEST model using E, S, T dimensions and P value as edges of the figure [1].

2 The Context: QEST nD and Spago4Q

In the following sections, we give an overview of the two frameworks describing the mathematical formalization of QEST *n*D, and the main characteristics of Spago4Q.

2.1 The QEST *n*D Model

In the software engineering context, several one-dimensional performance models are available which integrate individual measurements into a single performance index. By comparison, in more traditional domains such as Business Modeling, there exist a number of multi-dimensional models that take into account data derived directly from their accounting systems, which means that multiple viewpoints are, in fact, considered [1].

Furthermore, models currently available in the software engineering domains are too oversimplified to properly reflect the different aspects of performance when various perspectives, or viewpoints, must be taken into account at the same time. Therefore, to manage simultaneous multi-dimensional constraints in development projects, managers need to estimate the status of current projects based on their own interpretation of rough data, due to the lack of reliable measurement models.

In multi-dimensional analysis, complex viewpoints are taken into account simultaneously, each one analyzing a distinct aspect of the overall process performance. Therefore, an extension to the traditional single-dimension approach is needed, to consider both quality and performance of the development process.

The QEST *n*D model [3] is aimed at measuring software project performances addressing the aspects of multidimensionality and qualitative-quantitative assessment. With respect to the original QEST model that was initially designed for measurements to be done at the end of a project, QEST *n*D provides a dynamic extension to analyze software process data throughout all the development phases. In particular, in the

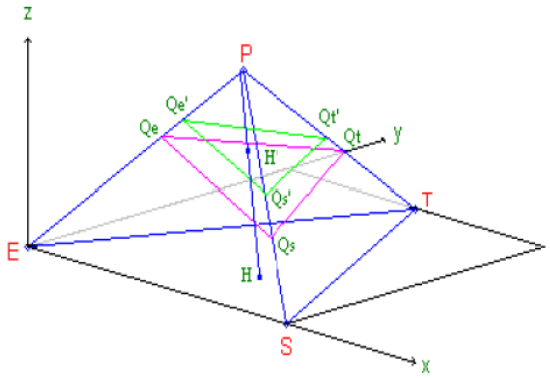

Fig. 2. (Q_e, Q_s, Q_t) and (Q'_e, Q'_s, Q'_t) plane sections [1].

QEST model the quality can be viewed as the integration of at least three different viewpoints.

- **Economical (E)**: expresses the viewpoint of *management*, interested in measurements focused on the overall quality level, rather than the quality of specific features or process areas.
- **Social (S)**: measures the perspective of the *user*, where the quality is intended as the characteristic of a product to satisfy present and future needs.
- **Technical (T)**: relative to *developers*, for whom software quality is achieved by conforming to specific, explicitly stated standards and requirements explicitly stated.

In the QEST model the measurement of performance (P) is given by combining a quantitative measurement, indicated by the component *RP-Rough Productivity*, and a qualitative measurement, calculated as a perception-based measurement of the overall product quality (*QF-Quality Factor*). A detailed explanation of the model, that has been formalized in [3] [4], is out of the scope of this paper. For the initial QEST model, the three-dimension geometrical representation of a regular tetrahedron (see Fig. 1) was selected and studied to help the model formalization. In particular:

- the three dimensions (E, S, T) in the space d to the corners of the pyramid's base, and the convergence of the edges to the P vertex describes the top performance level;
- the three sides are of equal length: the solid shape that represents this 3D concept is therefore a pyramid with a triangular base and sides of equal length (tetrahedron). The figure represents the ranges to which the dimensions performances will belong to.

The geometrical approach permits the representation of the measurement of performance in a simple and visual way, assisting the global performance computation. Thanks to this representation, it is possible to express the performance value in term of geometrical concepts like distance, surface, and volume. The value of each dimension is seen as the weighted sum of a list of *n* distinct measures, each one representing single measurable concepts for each perspective. Then, the values of the

three dimensions E, S, T are placed on the respective tetrahedron side, describing a sloped plane section and representing the three dimensional performance measurements [3]. Fig. 2 better explains such a geometrical aspect, indicating the three dimensions' values as Q_e, Q_s, and Q_t. Finally, the QF, with respect to each dimension, is added to the previous values describing an upward or downward translation of the plan (Q_e, Q_s, Q_t) finding the new plan (Q'_e, Q'_s, Q'_t).

On the other side, the value RP can simply be expressed as the distance between each single corner (E, S, T) with the specific point Q_e, Q_s, and Q_t. Please note that the maximum value of each edge is 1, consequently all the values that are placed on the tetrahedron have to be normalized. Then, the performance value is calculated as the distance between the center of gravity of the original tetrahedron and the center of the described plane section along the tetrahedron height (see Fig. 2).

The explanation above is valid for the QEST *n*D case, where more than 3 dimensions are taken into account. Through computational geometry, it is possible to develop a generic representation of it with a generalization of a tetrahedral region of space to *n* dimensions describing it with a *simplex* [4]. To conclude, the geometrical formalization of the model allows to describe it with a simple formula for the computation of the global performance value P:

$$P = 1 - \prod_{i=1}^{n} (1 - p_i) \qquad (1)$$

Where p_i represents the single dimension performance value added with the respective QF value. Section 3 and 4 deal with the definition of the environment and, in particular, of the metrics model that will be used to compute the value of the single dimensions.

2.2 Spago4Q

Spago4Q (SpagoBI for Quality) [2] is an open source platform for the continuous monitoring of software quality. Its most important characteristic is the total independence from the adopted development process and from the number of monitored projects. Spago4Q can then be described as a *multi-process multi-project* monitoring platform.

In Spago4Q, the evaluation of metrics and the collection of data are executed in a fully-transparent way, without any action due by programmers and designers and any change in their typical working tasks.

Spago4Q includes in its package a number of extractors for the main environments that are exploited during the software lifecycle (IDE, text editing tools, requirements management frameworks, and the like) that collect data directly from process work-products (e.g. java classes or logs).

Since Spago4Q relies exclusively on open frameworks and it is released under an open source license, its structure could be enriched with the implementation of additional extractors for particular work-product types. In any case, the extractors will

be executed at specified time intervals and store data directly in the application data warehouse.

Spago4Q is a vertical adaptation of SpagoBI [10], a more complex framework for Business Intelligence analysis, whose structure was enriched with the use of a complex meta-model (see Fig. 3) for the representation and description of the generic development process, the measurement framework, the extractors, and the assessment framework [5], that defines the entities that play a role in the monitoring process the relations between them. The *Development Process* meta-model has been designed to be as generic as possible, allowing the modeling of virtually all process models, from waterfall to XP. It is connected with the *Measurement Framework* meta-model, which defines a skeletal generic framework and is used to obtain measures from most development processes. Then, the *Assessment* meta-model allows to model a generic evaluation structure with a simple classification in terms of *Category, Target,* and *Practice*. Finally, the *Extractors* meta-model is used to formalize and define the extractors used to retrieve data from process module and supply it to the measurement module.

In particular, the inclusion of a specific meta-model for the assessment framework allows the tool to implement metrics that are specific to a particular assessment model. Originally Spago4Q was studied to fully support the CMMi framework [6] for

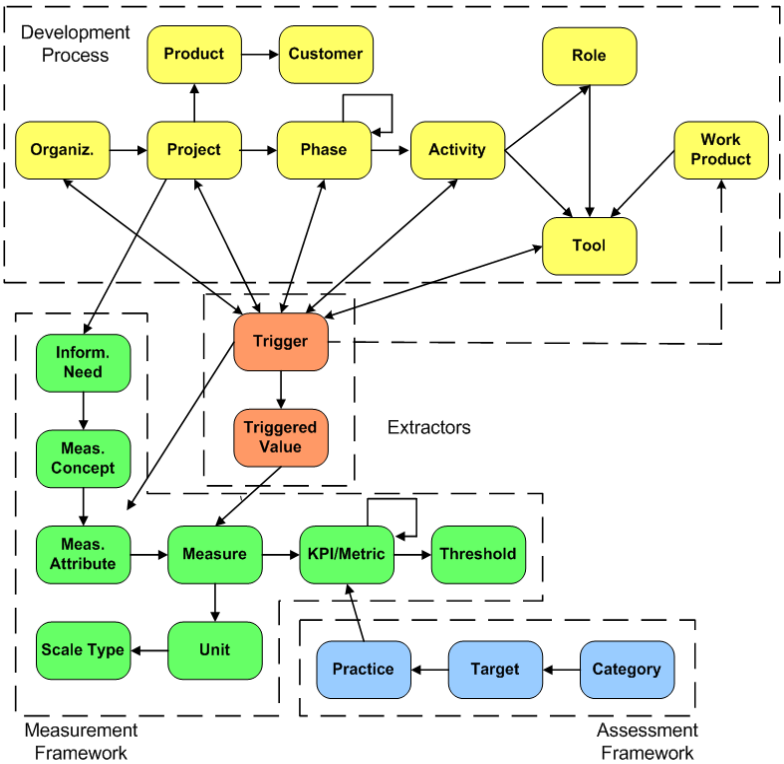

Fig. 3. The Spago4Q meta-model [5].

the maturity assessment of the development process. However, the intrinsic generality of the meta-model approach allows adapting it to any assessment model, for instance ISO 9000 or Balanced Scorecards. In fact, Spago4Q provides support for the definition and implementation of measurement frameworks based on the GQM (Goal-Question-Metric) approach [7], which categorizes the metrics in terms of *(i)* generic goals to be measured, *(ii)* the particular aspects that the metric has to measure, and *(iii)* the metrics that implement the actual measurement.

3 An integrated Environment

The below-described work has been the subject of a Master thesis carried out within the Department of Information Technology, Università degli Studi di Milano. The definition of a QEST *n*D model is a five-step procedure that will be described in the following paragraphs, without entering in the implementation details for the sake of conciseness.

The steps, all executable through the graphical interface of Spago4Q, are coherent with the PMAI (*Plan-Measure-Assess-Improve*) cycle [8], which is composed of four logical phases:

1. **PLAN,** which consists in defining a set of metrics basing on the GQM approach, defining the dimensions that characterize the analysis, the mathematic formalization of the metrics and the weight to assign to each metric.
2. **MEASURE,** which includes the collection of data, the computation of metrics values, the normalization of them (values must be ≤ 1), and the computation of global Performance value using the Eq. 1.
3. **ASSESS:** results are presented in dashboards and reports, and analyzed by the management and analyzers.
4. **IMPROVE:** every negative or low value is deeply analyzed to find problems in the processes and consequently find solutions to improve the overall quality.

3.1 Step 1: Metrics and Model Definition

The first step deals with the declaration of a complete GQM, with the definition of the analysis dimensions, the concepts to measure, and the metrics to apply to project work-products.

The GQM will be defined using the *Model Definition* interface, while metrics (or KPI – Key Performance Indicator) are defined through the *KPI Definition* section.

In particular, the definition of KPIs involves the specification of an algorithm for the computation of the metric. This algorithm will exploit the SQL mathematical library for simple computations, or call an *ad-hoc* Java class for more complex ones. The KPI will collect data directly from the Spago4Q data warehouse, which contains all the data that the extractors get from the project work-products.

3.2 Step 2: Weights and Threshold Definition

The QEST *n*D model requires each metric to be coupled with the respective *weight*, which indicates the importance that such a concept plays in the dimension it belongs to. A complete analysis of the GQM should be performed prior to define the weights for each KPI. Thus, for each metric it is important to define the specific thresholds, which allow to evaluate the value with respect to the organization policies. The thresholds have to take into account the normalization of metrics and are also important for the creation of complete and understandable reports. The *KPI Definition* interface helps to define such aspects.

Finally, although the use of the QF is optional and its absence does not preclude the entire Global Performance value, in this step it is possible to assign the QF to each specific dimension. The definition of the QF is subjected to the analysis of pool of experts that define the value that will be added to the respective RP. A complete guide for the definition of the QF of a software product can be found in [3].

3.3 Step 3: Measures Collection

Measures are taken directly from Spago4Q data warehouse, which in turn is filled by data collected by extractors accessing process work-products. The collection process is defined in the configuration phase, where a specific *dataset*, that contains the description of the metric itself, is defined for each KPI or group of KPIs.

Metrics are described in terms of *(i)* the name of the model to which the metric is assigned to, *(ii)* default value, *(iii)* minimum and maximum values (for normalization), and *(iv)* the algorithm for the metric computation.

In particular, the algorithm can be implemented using the common mathematic library of SQL, as a separate Web Services or, for computations that involve complex operations, as a Java package. Furthermore, the application supplies to users KPI-specific drivers to be used in the metric formula to help the definition of it in the selected programming language, supplying methods for the direct access to data warehouse fields. Finally, specific fields for the KPI results will be added to the data warehouse and supplied to other KPIs and components of Spago4Q.

3.4 Step 4: Global Performance Computation

In our approach, both global and dimension-wise performance indexes are computed as simple KPIs that take in input configuration data and results of the metrics at the bottom of the GQM tree. First of all, the performance value of each dimension is calculated as the sums of the product of each metric, belonging to the dimension, with its specific weight:

$$D = \sum_{i=1}^{n} (V_i * w_i) \qquad (2)$$

where *D* is the dimension performance value, *V* is the result of the *i*-th metric, and *w* is the assigned weight. If the QF is provided by the model, its value is added to the results:

$$D' = D + QF = \sum_{i=1}^{n} (V_i * w_i) + QF \quad (3)$$

Note that Eq. 2 and 3 are computed for each of the *n* dimensions that compose the QEST *n*D model that has been specified. Finally, the KPI that computes the global performance indicator could be defined using the Eq. 1, defined in Section 3.1 and it is valid in the case the QF is defined or not.

$$P = 1 - \prod_{j=1}^{n} (1 - D_j) \quad (4)$$

The global performance *P* value will be assigned to the root node of the created model.

3.5 Step 5: Reports

As last step of the process, a set of reports and dashboards could be defined and configured to satisfy any reporting and managerial need. One of the open source reporting tools we integrated with Spago4Q is Eclipse BIRT [9]. The generation of reports requires the creation of a specific dataset that includes all the data that will be described by the report.

Such data is collected from the application data warehouse and, in particular, from the fields that contain the metrics values. Several pre-defined templates and layouts are available.

Spago4Q provides methods and interfaces to directly configure and create a new report using all the facilities provided by BIRT.

4 Case Studies

The application of the QEST *n*D model in Spago4Q has been tested using two real case studies on data taken from real development projects realized by Engineering Ingegneria Informatica, a major player within the community of Spago4Q.

The two projects are of increasing complexity: the first one deals with a little project consisting of a single measurement of project data, while the second one measures the complex performance of three big projects with several measurements in a three-month time slot.

For the sake of conciseness, in this section we focus on the second case study only, providing information about the steps that were described in Section 3. The realized QEST *n*D model was called *Business-Service Model* and takes into consideration four specific analysis dimensions:

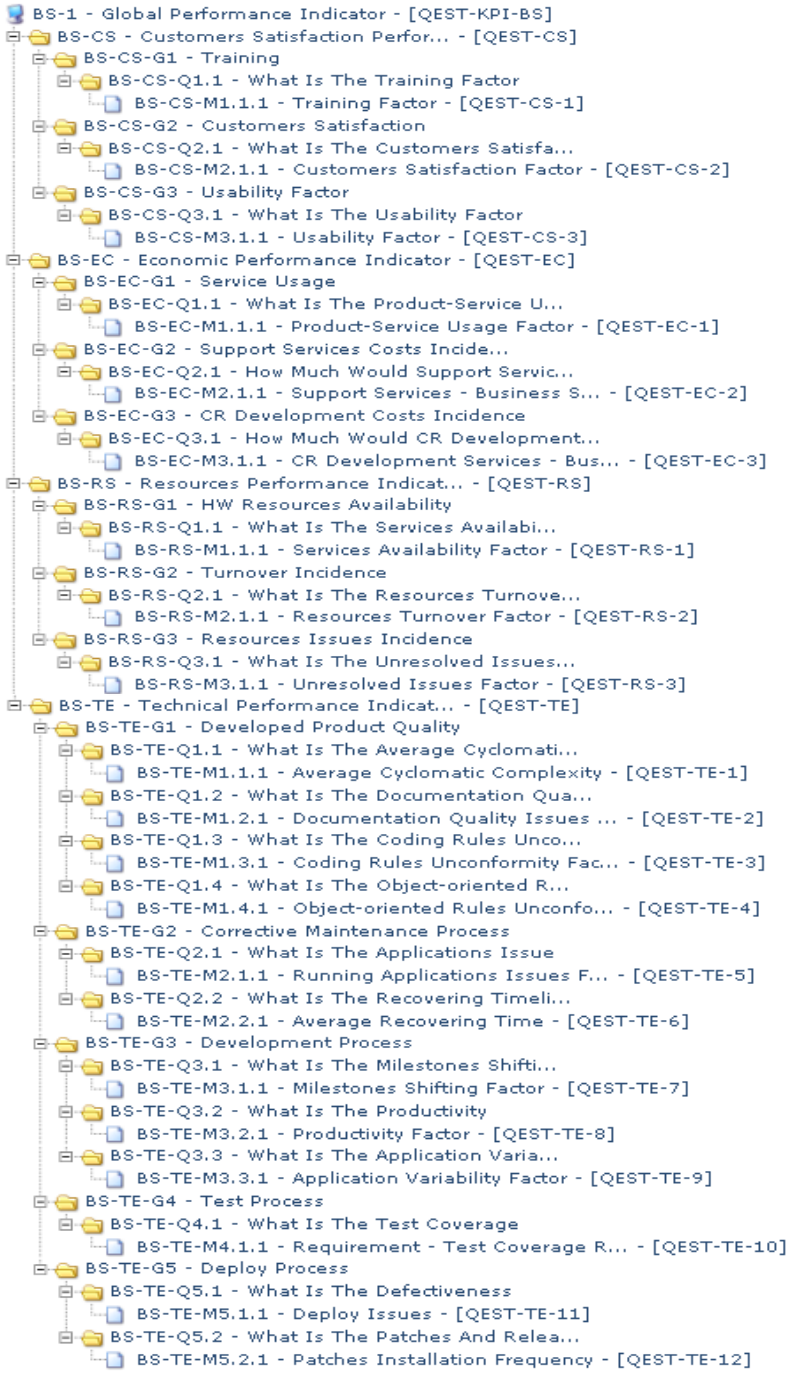

Fig. 4. The complete GQM model defined in the case studies.

1. QEST-EC: Economic performance indicator;

2. QEST-RS: Resource performance indicator;
3. QEST-TE: Technical performance indicator;
4. QEST-CS: Customer Satisfaction performance indicator.

Fig. 4 shows the complete model, highlighting the GQM structure of the metrics. The root node is the global performance indicator (QEST-BS), which includes the four **goals** describing the analysis dimensions. For each dimension, a set of **questions** (i.e., the concepts to measure) has been defined, which in turn includes the **metric**, or the metrics, which evaluates the concept. Table 1 summarizes the metrics, along with the respective weights, that compose the model.

Each metric is associated to specific SQL queries or Java classes that directly access to the data warehouse to collect the input for the computation. For each metric, a field in the data warehouse has been created to store its output, to be used by other KPIs or reports.

Table 1. List of KPIs defined for the case study.

KPI	Description	Weight
QEST-KPI-BS	Global Performance Indicator	
QEST-EC	Economic Performance Indicator	1.0
QEST-RS	Resources Performance Indicator	1.0
QEST-TE	Technical Performance Indicator	1.0
QEST-CS	Customers Satisfaction Performance Indicator	1.0
QEST-EC-1	Product/Service Usage Factor	0.2
QEST-EC-2	Support Services – Business Services Costs Ratio	0.4
QEST-EC-3	CR Development Services – Business Services Costs Ratio	0.4
QEST-RS-1	Services Availability Factor	0.7
QEST-RS-2	Resources Turnover Factor	0.15
QEST-RS-3	Unresolved Issues Factor	0.15
QEST-TE-1	Average Cyclomatic Complexity	0.1
QEST-TE-2	Documentation Quality Issues Factor	0.05
QEST-TE-3	Coding Rules Unconformity Factor	0.05
QEST-TE-4	Object-oriented Rules Unconformity Factor	0.1
QEST-TE-5	Running Applications Issues Factor	0.15
QEST-TE-6	Average Recovering Time	0.1
QEST-TE-7	Milestones Shifting Factor	0.1
QEST-TE-8	Productivity Factor	0.1
QEST-TE-9	Application Variability Factor	0.05
QEST-TE-10	Requirement - Test Coverage Ratio	0.08
QEST-TE-11	Deploy Issues	0.07
QEST-TE-12	Patches Installation Frequency	0.05
QEST-CS-1	Training Factor	0.1
QEST-CS-2	Customers Satisfaction Factor	0.6
QEST-CS-3	Usability Factor	0.3

It is important to remark that the definition of weights and thresholds has to be very careful and must involve skilled experts that have a solid background in the enterprise scenario. In fact, an overestimation, or underestimation, of metrics weights will result in a global value that does not reflect the process state, as well as, the definition of incorrect thresholds will imply an incorrect analysis of the organization status. In these case studies, for the sake of conciseness, the QF has not been taken into consideration; hence the performance computation takes into account only the weighted sums of the metric results. The experimentation covered three months of development. Raw data have been collected by the application from process work products and stored in the data warehouse. In particular, Table 2 shows a snapshot of values collected at the end of the three months for each project.

Finally, metric values were normalized and used as inputs to Eq. 2 and Eq. 4, for the computation of single dimension and global performance indicators. The values of indicators can be represented using the internal Spago4Q dashboards (Fig. 5) or the user can create *ad-hoc* reports using BIRT functionalities. In particular, dashboard gives an immediate snapshot of the situation, highlighting problems and suggesting to project managers the areas where the effort should be concentrated or where a quality improvement of the process is needed. By contrast, reports can give a more detailed analysis of the data, describing in details the results of indicators and better targeting the improvement actions.

Table 2. Metrics values collected for the three projects at the end of each month. Note that data are cumulative and have to be normalized before performance computation.

KPI	Project 1			Project 2			Project 3		
	M1	M2	M3	M1	M2	M3	M1	M2	M3
QEST-EC-1	80	78	82	50	68	72	25	49	70
QEST-EC-2	25	45	30	5	7	6	8	9	7
QEST-EC-3	50	65	55	2	4	5	14	11	10
QEST-RS-1	99	97	98	60	70	87	97	95	96
QEST-RS-2	91	94	95	96	93	94	98	97	97
QEST-RS-3	50	62	82	35	37	68	10	35	88
QEST-TE-1	20	18	15	30	32	25	50	39	35
QEST-TE-2	1.1	1.9	1.8	1.8	2.3	2.1	4.5	3.2	1.9
QEST-TE-3	95	93	94	96	92	93	82	91	92
QEST-TE-4	99	97	95	30	49	67	85	89	93
QEST-TE-5	2	1	2	1	2	1	1	2	2
QEST-TE-6	3,5	3.2	3.1	2,4	1.3	1.1	16	12	8
QEST-TE-7	87	78	83	88	92	93	50	80	85
QEST-TE-8	10	20	17	55	40	35	68	65	55
QEST-TE-9	2	122	9	25	15	12	29	22	18
QEST-TE-10	375	390	410	178	230	245	210	240	255
QEST-TE-11	1	2	2	2	1	1	3	2	2
QEST-TE-12	2	3	4	9	6	5	10	8	7
QEST-CS-1	60	75	89	95	91	92	94	92	91
QEST-CS-2	75	85	90	91	87	90	93	95	94
QEST-CS-3	91	83	92	96	93	92	80	83	93

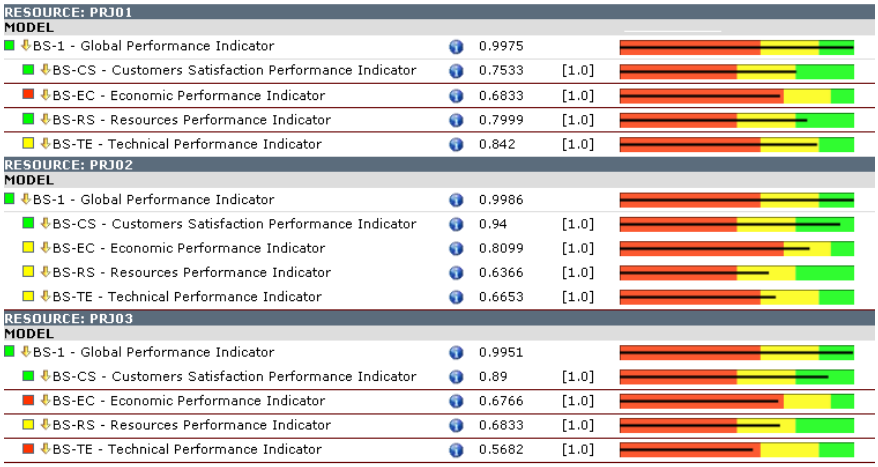

Fig. 5. Spago4Q dashboards for projects state at month 3.

Looking at the results of our case study, the graphs in Fig. 6 and 7 show that all projects were concluded with an excellent global performance (close to one), showing some issues in the process that is worth analyzing.

For instance, the Economic dimension of Project 1 is characterized by a red square, indicating that the value is within the bad area, hence a deep analysis of that area is needed for next implementations. In fact, project managers discovered that the financial resources assigned to the project were overestimated for the needed effort, suggesting an adjustment to the enterprise criteria for projects financing.

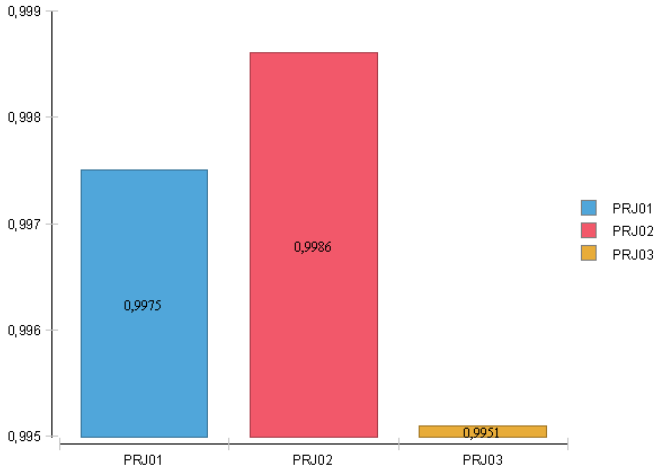

Fig. 6. Spago4Q graph for projects global performance comparison.

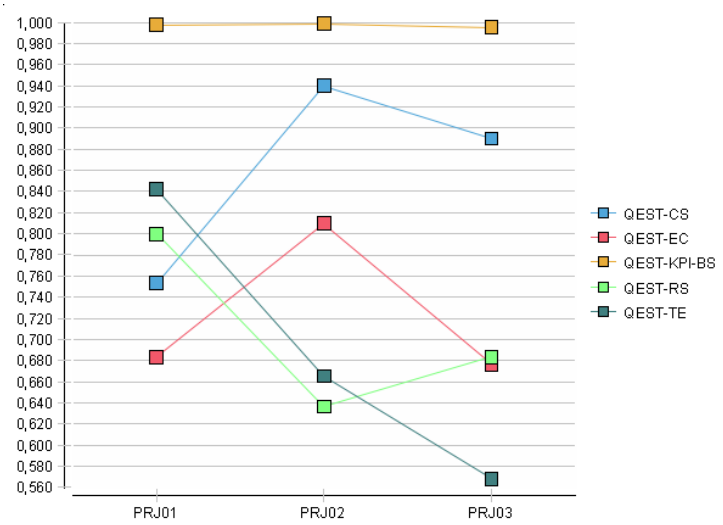

Fig. 7. Spago4Q graph for projects single dimensions performance comparison.

5 Conclusions

The integration of the QEST *n*D model and Spago4Q allows implementing a complete multi-project multi-process performance evaluation environment that combines the mathematical formalization of the QEST *n*D model and the facilities offered by Spago4Q. In this paper, we analyzed such an integration, and described a complete case study that shows the high configurability and reliability of the framework.

The contribution is twofold. First, we implemented and used a formal model for process performance evaluation (QEST *n*D) and we connected it with an open source Business Intelligence application (Spago4Q). Second, we developed a solution that derives global performance indicators of the enterprise developing work by analyzing its process raw data (e.g., java classes, logs).

The main benefit of the proposed solution lays in the fact that it gives the possibility to analyze the performance of the development process from different points of view and integrate semantically different metrics and KPIs in a single indicator. The QEST *n*D model described in this paper will be made available in the future version of Spago4Q.

Acknowledgments

This work was partly founded by the European Commission under the project SecureSCM (contract n. FP7-213531) and by the Italian Ministry of Research under FIRB TEKNE (contract n. RBNE05FKZ2_004).

References

1. Buglione L, Abran A (2002) QEST nD: n-Dimensional Extension and Generalisation. Advances in Engineering 33 (1): 1-7
2. SpagoWorld Solutions - Spago4Q (2009). Available at: www.spago4q.org
3. Buglione L, Abran A (1999) A quality factor for software. Proc. 3rd International Conf. on Quality and Reliability (QUALITA99): 335-344, Paris, France
4. Buglione L, Abran A (1999) Geometrical and statistical foundation of a three-dimensional model of performance. Int. J. Adv. Eng. Software 30 (12): 913-9
5. Damiani E, Colombo A, Frati F, Oltolina S, Reed K, Ruffatti G (2008) The use of a meta-model to support multi-project process measurement. Proc. 15th Asia-Pacific Software Engineering Conference (APSEC): 503-510, Beijing, China
6. Software Engineering Institute - Carnegie Mellon University (2009) CMMi - Capability Maturity Model integration. Available at: www.sei.cmu.edu/cmmi/
7. Basili V, Caldeira G, Rombach H D (1994) The Goal Question Metric approach. Encyclopedia of Software Engineering, Wiley
8. Deming W E (1986) Out of the Crisis. MIT Press
9. Eclipse BIRT Project (2009). Available at: www.eclipse.org/birt/phoenix/
10. SpagoWorld Solutions – SpagoBI (2009). Available at: www.spagobi.org