WebEv - a Collaborative Environment for Supporting Measurement Frameworks

Lerina Aversano, Thierry Bodhuin, Gerardo Canfora and Maria Tortorella
RCOST - Research Center On Software Technology
Department of Engineering, University of Sannio
Via Traiano, Palazzo ex-Poste – 82100, Benevento, Italy
(aversano/bodhuin/canfora/tortorella@unisannio.it)

Abstract

Assessment activities play a key role when an enterprise wants to evolve its business processes for adopting innovative information and communication technologies. The assessment regards business processes and supporting software systems and, to be effective, it requires the support of adequate software environments that facilitate the collaboration among the involved assessors. The definition and implementation of such a kind of environment implies the previous definition of a measurement framework that may guide the specification and design of the environment.

This paper presents the collaborative software environment WebEv to support any kind of assessment framework whose definition is based on the Goal Question Metric paradigm. The environment is flexible and easily customizable to any context and arising needs. It can represent a practical mean for the collection of the available quantitative information regarding the processes and/or software systems of an enterprise.

1. Introduction

Success in business innovation requires an analysis of the current processes and supporting systems of an enterprise and their assessment [10]. Thus, systematic approaches and tools have to be defined in order to guide the assessors in the collection of important data in a correct and reproducible manner, and decrease the risk of erroneous interpretation and/or superficial analysis. It is necessary to provide useful guidelines to identify: the objects to be assessed and evaluated; the needed measurements; and the manner in which the collected information can be synthesized to achieve useful suggestions. Moreover, the assessment activities entail the interaction of different actors including software engineers and process owners, aiming at achieving a deeper knowledge of the analyzed objects. The involved actors do not necessarily work in the same organization. Thus, the availability of collaborative environments becomes important for facilitating their reciprocal interaction during data gathering and analysis.

In this paper, a collaborative software environment, named *WebEv*, *Web for the Evaluation*, is proposed to support assessment activities. *WebEv* has been designed for supporting any kind of assessment framework whose definition is based on the Goal Question Metric (GQM) paradigm [3, 5]. The environment is flexible and easily customizable to any context and arising need. In addition, it is distributed, so that any possible user through the Internet can access it. *WebEv* can be used in different assessment contexts. In particular, it has been defined to support the assessment of evolving business processes [2]. In particular the assessment of the supporting software systems has a central role for evaluating the costs and risks of renovating and integrating them into the new process view. *WebEv* has been used in a real assessment case study within an on-going research project, named Legacy to e-Legacy Re-engineering System, aiming at developing a framework of methodologies, techniques and tools to update traditional software toward net centric environments.

The next section describes the improvement design guiding the characterization, evaluation and evolution of the processes and systems. Section 3 presents in greater detail the *WebEv* environment, while the following section describes how *WebEv* can be used and, bearing this in mind, introduces a part of an assessment framework. Section 5 presents some previous research. Concluding remarks and future work are outlined in section 6.

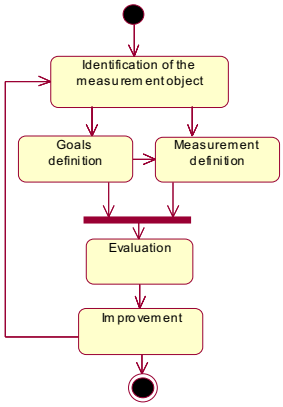

Figure 1 - Measurement and Improvement Approach

2. Approach to measurement

The need to improve the internal business processes of an enterprise implies the continuous monitoring of the enterprise's internals [7]. Key problems concern the identification of the items to be analyzed, such as the processes, supporting systems and results they produce [4], and the definition of assessment methods and tools. In real contexts, it is impossible to measure all the processes' elements and it is, therefore, necessary to identify a limited number of significant aspects to be monitored. Furthermore, the data measurement mechanisms have to be defined, together with the approaches for abstracting meaningful information.

Different actors are involved in a monitoring process, such as, for example, Business Process Analysts, Domain Experts and Managers. The Business Process Analyst analyses the processes and, together with the Domain Expert, collects the data necessary for measuring them. The Manager defines the improvement goals, evaluates the analyzed processes, and, on the basis of the results obtained and the Domain Expert's suggestions, determines the guidelines to be adopted in the process improvement. In addition, on the basis of the Manager's suggestions, the Business Process Analyst actually implements the improvement.

Figure 1 illustrates the macro-activities to be performed and their flow. These activities entail the continuous monitoring of the enterprise's internal and external components. The external components are related to the identification of innovative technologies, new laws and users' needs, and so on. The internal components are the business processes and involved software systems. As a consequence of this monitoring and on the basis of the suggestions of the enterprise's management, the goals to be achieved are identified. Consequently, a generic measurement framework with related acceptance criteria can be defined. It has to be complete in terms of what is expected from a measurement framework, i.e. what to measure, how to measure, and how to use the collected data. A good support to define an effective framework is offered by the GQM paradigm [5]. It deals with the problem of a measurement framework definition by considering [3]: a *Conceptual* level, referred to the definition of the GOALs, an *Operational* level, consisting of a set of QUESTIONs regarding the specific goals, and a *Quantitative* level, identifying a set of METRICs to be associated to each QUESTION.

Figure 2 shows the definition and application of a measurement framework based on the GQM paradigm in the form of a use case diagram. On the basis of the guidelines of the Manager, the Software Engineer has to define the goals, questions, metrics and measure abstraction mechanisms of an initial generic framework. The next measurement activity consists of assigning measures to the framework metrics. This activity must be as objective as possible, and when needed it is useful to define the values the metrics can assume. The first version of the framework can be generally defined. Its customization to the specific domain can be performed only when the processes and software systems to be analyzed are selected and it is possible to consider the specific requirements of the data collection activity. In this context, the software engineer has to refine the framework by verifying if the measures considered as generic can be applied to the specific context, and introducing specific sections to adapt the framework to the particular chosen objects to be assessed. The identified parameters of the measurement framework can be defined also in terms of acceptance criteria indicating the target values they can assume.

Once the measurement framework is fully defined, the Assessor, that is the owner of the analyzed objects, can perform the analysis and measurement activities with the help of the Software Engineer. He has to store all the collected information and notify eventual anomalies to the Software Engineer. The measurement framework can be applied each time it is needed. At the beginning of the improvement activities, it can be applied for measuring quality aspects of the objects to be improved. Once an evolution has been performed, the measurement framework can be re-applied. In this case, the framework can be used to identify if the planned improvement has been satisfied by comparing the quality results obtained by applying the framework before the evolution and those reached after.

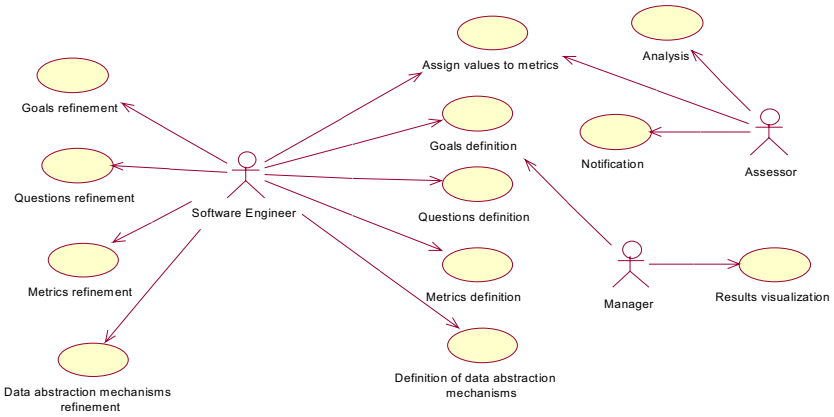

Figure 2 – Use case diagram of the assessment process and WebEv

Besides illustrating the interaction with the measurement framework, Figure 2 represents the use case diagram of a software environment supporting the identified activities. It shows that the environment has to provide a support for all the actors of the measurement activities in the terms above described. It has to be flexible in the sense that it has to support not only the definition of the framework but also its refinement, by keeping track of its evolution. The assessor has to be able to store all the information collected in the analysis and measurement activities in such a way that the traceability between the old stored and new information is maintained. From the manager's point of view, an effective instrument is needed to query the collected data and to interact with the Software Engineer for helping her/him to define the goals. Moreover, as already stated, it is important that the environment is based on Web-technologies, in order to facilitate the actors' cooperation in an effective manner. The next section presents *WebEv* in greater details.

3. The WebEv Environment

WebEv is the environment implemented to support all the activities related to the definition, instantiation and characterization of the measurement framework. It can represent practical means for collecting all the available quantitative information regarding the assessment task and permitting its diffusion through out the enterprise. The recovered data can be used to identify the quality of the assessed object and provide useful information to be considered during its evolution.

WebEv can be used both in the planning and enacting stages of the assessment. In the first stage, *WebEv* permits the definition of the measurement framework and its customization to the specific context. Subsequently, it supports the configuration and management of the items under assessment and the actual collection of metrics. Using a Web-based interface, software engineers can collaboratively define the measurement framework, by identifying the Goals of the analysis, defining all the Questions and Metrics required, and then collecting the information of interest. The structured framework can be stored with the identified information and mutual interactions, by using the environment functionalities in the application database. The assessors can access via the Internet the framework questionnaire and use the environment facilities for answering the questions through the Web-based user interface. Finally, the manager can use *WebEv* to abstract significant data from all the collected information.

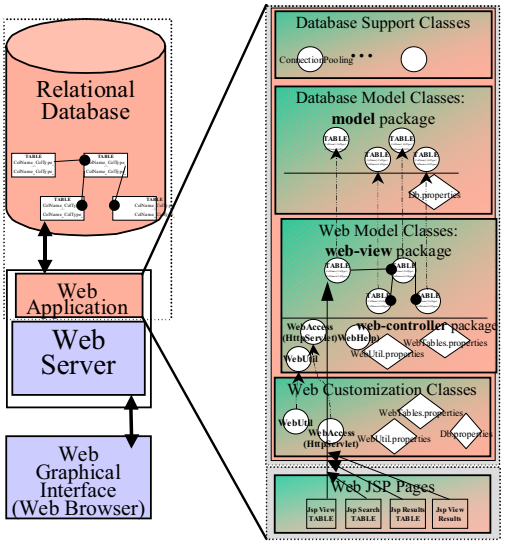

Figure 3 – *WebEv* application structure

One of the main requirements of *WebEv* is the possibility to store a large amount of information that can be assessed by different users acting in different sites. Thus, it is necessary to provide an environment based on Web technologies. The technologies chosen to accomplish these requirements are:

- The Java programming language, chosen for its portability and independence from the hardware platform and operating system environment.
- The Java JDBC (Java Driver Database Connection) API, chosen for the connection with the relational database.
- The Java Servlet API, chosen for implementing the interface between the Web server and the application.

WebEv is based on the Model-View-Controller (MVC) design pattern chosen to achieve maximum reciprocal independence and proper localization of the functionalities. This is reached by maintaining the separation among the generated classes implementing the interface with the database (Model), the ones implementing the Web/HTML graphical interface (View), and those implementing the connections between the previous ones (Controller).

Figure 3 shows the *WebEv* architecture and the various packages complying with the MVC design pattern. The *model package* interacts directly with the database table and contains a class for each of them. This class contains the attributes mapping the columns of the table, and the methods aiming to perform set and get accesses to them, and database queries (i.e. search, insert, update, delete operations). The classes implement the operations for retrieving, updating, inserting, and deleting the objects from/into the database. The APIs of these objects are compliant with the Java Beans specifications including set and get methods for all the attributes. A file named *Db.properties* contains the standard parameters of the database connection (e.g.: JDBC driver, JDBC URL, Login and Password of the user). The *model package* eases the expandability of *WebEv*. If new functionalities which need to access a database table are required, the direct database access does not need to be implemented. In fact, it is enough to invoke the methods of the *model package* class modeling the generic operations of the table at model level.

Also the *web-view package* in Figure 3 contains a Java class for each relational database table. A web-view package class contains the methods generating, upon HTTP request, the HTML, XML or CSV pages implementing at the graphical interface level the retrieval, visualization, modification and deletion operations of one object mapping one database row. Other methods, specific to the database table columns, are added for managing the attributes of the corresponding class in the *web-view package*.

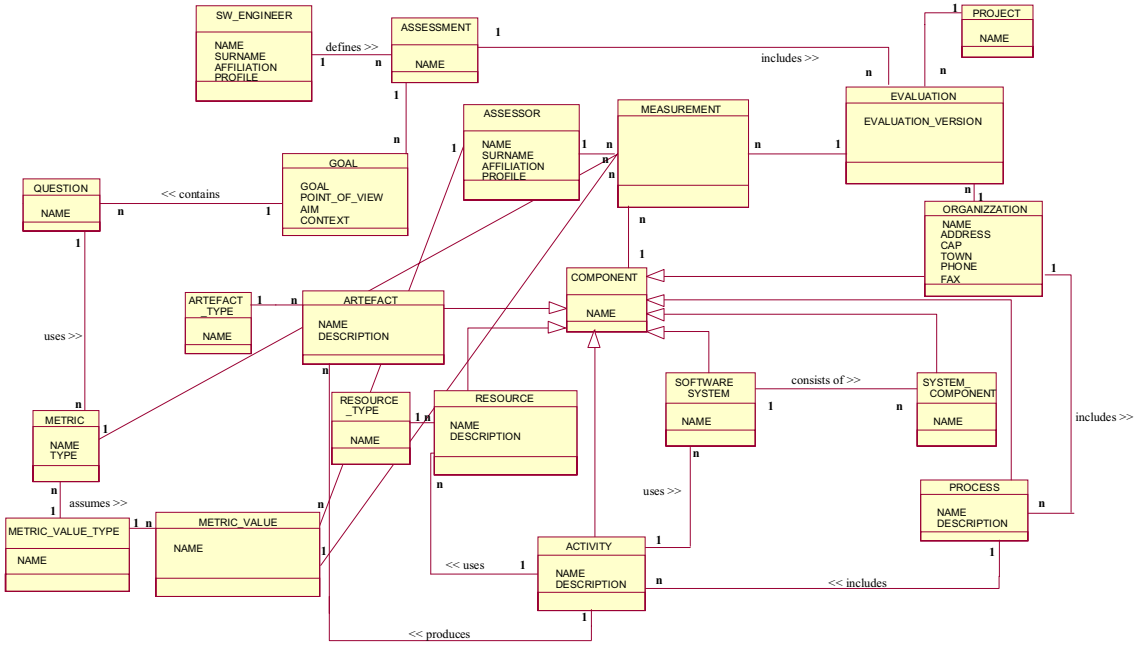

Figure 4 – WebEv Database Model

The *web-controller package* in Figure 3 contains the *WebAccess* and *WebHelp* classes to allow the integration of the model package classes with the web-view package ones. These two classes respond directly to the HTTP request and are implemented as Java Servlet. The *WebAccess* class interprets the HTTP request as an action to be applied to a specific object of the model package class and treat it by calling a specific method of the object class. The *WebHelp* class allows the display of a HTML Help window indicating to the user how to create and/or update the information of an object presented in a HTML form. In addition, the *web-controller package* contains the *WebUtilInterface* interface and the *WebUtil* class, implementing it. The *WebUtil* class contains the utility methods permitting the definition of the HTML layout of the environment for each data type. It can be extended for customizing the HTML layout of the environment. For generating the HTML pages, the web-view package classes use some parameters stored in two files: *WebUtil.properties* and *WebTables.properties*. They permit the personalization and internationalization of the text displayed by a Web application.

Figure 3 shows that *WebEv* uses Java Server Pages (JSP) including HTML search form pages, HTML/XML or CSV results pages and HTML view item pages for each related table in the database model. They permit the formulation of the data abstraction mechanisms and the insertion of the formulas to be applied for combining the values of the metrics and answering the questions.

All the information to be managed by the measurement framework is stored in a relational database whose partial data model is shown in Figure 4. It permits the storage of information regarding the definition of the measurement framework, in terms of Goals, Questions, and Metrics and its characterization in terms of the values the metrics assume. It takes in consideration the fact that the metrics can refers to various kinds of components interacting with each other and having different granularity. For example, it is possible to measure aspects related to both processes and supporting systems. Likewise, if the evaluation refers to processes, the assessment of their activities, resources, and supporting software systems may be required. The assessment of some quality attributes of a software system implies the evaluation of each composing subsystem. For these reasons the entity COMPONENT related to the entity MEASUREMENT is a generalization of all the objects that can be considered for being assessed. The database has been designed also to keep trace of the various framework versions, by storing the designer and all his characterizing information. On the other hand, it considers all the data the assessor collects connecting them to the measurement edition, evaluated components and metrics. In this way, by querying the database, it is possible to combine the gathered data by implementing the formulas defined to answer the framework questions and evaluate the goals.

Figure 5 shows the generic screen dump of the interface of *WebEv*. It is divided in three frames. The left frame contains the list of the tables that can be edited and/or searched. They are grouped on the basis of their usage and the user role. Three kinds of users can be defined: the *Administrator* can access all the tables and functionalities and is the only user that can define new users by interacting with the links in the Role Menu; the *Software Engineer* can access the definition operations (see Figure 6); and the *Assessor* can access all the analysis and evaluation operations and can view the goals, defined questions and metrics (see Figure 7 and Figure 8).

During the definition activity, it is possible to interact with the tables listed in the Software Engineer section and store information related to the framework definition, the project, the kind of assessment to be performed and the measurement framework. During the Analysis activities, the Assessor section permits an interaction with the tables storing the data related to the objects to be evaluated and the reciprocal interactions. They can refers to the organization, organization's processes, its activities, all the used resources, human and not, and/or adopted software systems. The software system can be divided into software components that can be individually evaluated. Finally, the Assessor can access the Measurement link and view the entire framework. The top right interface frame in Figure 5 is used for querying the selected table, while the bottom right frame permits its editing and the visualization of the selected information and its connected tables.

4. Using WebEv

In order to verify the usefulness, efficacy and applicability of *WebEv* and validate its functionalities, a measurement framework for the assessment of software system has been used. It aims at assessing a software system in order to identify the requirements for its evolution in an *e*Business context [10]. The characteristics considered are defined with reference to the standard ISO/IEC 9126 [9].

Table 1 shows the part of the measurement framework related to the *Maintainability* characteristic and its sub-characteristics. This paper does not report the entire measurement framework, as its aim is not to present it, but to illustrate how *WebEv* can be instantiated when a framework has to be used. Greater detail regarding all the characteristics considered in the measurement framework can be found in [1].

Table 1– Part of the Measurement Framework

Code	Metric Identifier	Questions and Metrics for Maintainability
GOAL: *to evaluate the quality of the software system from the developer and maintainer's point of view*		
Q9		**What is the software system analyzability?**
M9.1	*t_rec*	Cyclomatic number
M9.2	*t_anal*	Number of statements
M9.3	*t_spec*	Comments rate
Q10		**What is the software system changeability?**
M10.1	*n_j*	Number of jump
M10.2	*n_nl*	Number of nested levels
M10.3	*a_vest*	Average size of statement
M10.4	*n_var*	Number of variables
Q11		**What is the software system stability?**
M11.1	*n_par*	Number of parameters referenced
M11.2	*n_gvar*	Number of global variables
M11.3	*n_par*	Number of changed parameters
M11.4	*n_rel*	Number of called relationships
Q12		**What is the software system testability?**
M12.1	*n_ncl*	Number of non-cyclic path
M12.2	*n_nl*	Number of nested levels
M12.3	*n_cyc*	Cyclomatic number
M12.4	*n_cp*	Number of call-paths
Q13		**What is the software system recovery time?**
M13.1	*t_r*	Time to system recovery
M13.2	*t_f*	Time to system failure
Q14		**What is the software system degradation rate?**
M14.1	*n_uf*	Number of unrecoverable functions
M14.2	*t_nf*	Total number of functions
Q15		**What is the time to switch?**
M15.1	*t_sw*	Time to stand-by system activated
M15.2	*t_f*	Time to system failure

Table 1 presents the goal concerning the evaluation of a software system to be assessed from the developer and maintainer's point of view. The characteristic considered is *Maintainability*. Table 1 contains questions and the list of the metrics usable in answering the questions and evaluating the goal. A code, an identifier and a description identify each metric.

Maintainability regards the effort needed to make modifications on the software system. It indicates how it is easy to understand, correct and adapt the software. The key sub-characteristics considered are: *Analyzability* concerning the effort needed for diagnosis of deficiencies or causes of failures, or for the identification of parts to be modified; *Changeability* regarding the effort needed for modification, fault removal or environment change; *Stability* which refers to the risk of unexpected effect of modifications; *Testability* concerning the effort needed for validating the modified software. The values to be assigned to the metrics are mainly numeric. Some of them are at a high level of abstraction, like *n_cyc*, and can be evaluated indirectly from the values of other metrics, which are simpler.

Figure 6 shows a screenshot of the *WebEv* main form, with parts of the framework defined for the assessment of the Maintainability. By using the proposed environment, the software engineer can define the measurement framework by specifying the goals, questions and metrics to be included. The definition is performed using the appropriate menu elements. Selecting the Project item from the Software Engineer Menu, a new frame is loaded and data regarding the measure project can be edited, (see region number 1). Then, the personal information of the software engineer who will manage the assessment can be edited through the frame in region number 2. If the Software Engineer is already in the database it is sufficient to select s/he from the list. Similarly, goals and related questions can be defined through the fulfillment of the forms in region 3. In the same way, it is possible to define metrics related to the defined questions and the values they can assume. Also in this case, when the software engineer identifies the metrics for each question, s/he can select from already defined ones or can insert a new one.

From both the Software Engineer and Assessor Menus, it is possible to achieve a synthesis of the data characterizing the measurement framework. In Figure 7, the framework view is shown when the user has the role of Assessor. An important element necessary for the effective management of the evaluation is needed to provide an useful support for the calculation of the metrics. With this in mind, a set of formulas is defined to combine the individual gathered measures. For example, with reference to Table 1, to evaluate the probability that the system can be restored within a given time after a failure, may be evaluated by calculating:

- *System Recovery Time*, *SRT*, indicating the time required to recover a system, and calculated as:
  $SRT = t\_r - t\_f$ *Unit: minute or hour or day*
- *System Degradation Rate*, *SDR*, indicating the percentage of unrecovered functionalities, and evaluated by:
  $SDR = (n\_uf / t\_nf) \times 100\%$ *Unit: normalized to 0-100%*
- *Time to Switch*, *TSw*, calculated as:
  $TSw = t\_sw - t\_f$ *Unit: minute or hour or day*

By using the *WebEv* facilities for the analysis, the software environment permits the insertion of the formulas and evaluates them automatically on the basis of the values assumed by the metrics. In particular, from the Measurement Menu in the left

frame, a manager can select the item of interest and achieve a view of the assessment with detailed data and/or their synthesis. In Figure 8, there is a synthesis of the data collected related to the maintainability characteristics of a software system and the synthesis achieved for the system degradation rate.

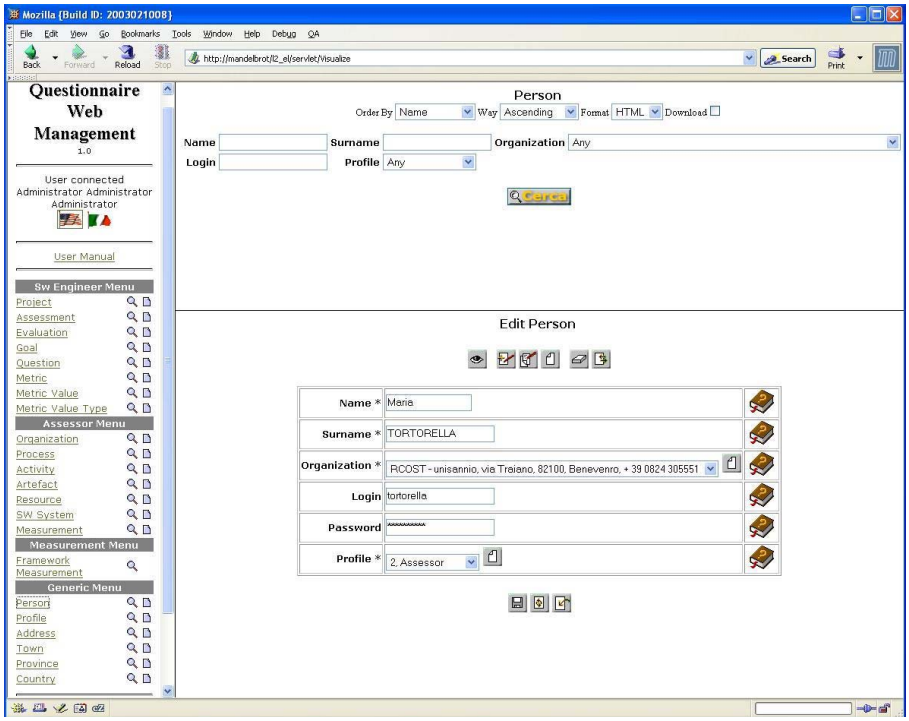

Figure 5 – Generic WebEv Screen Dump

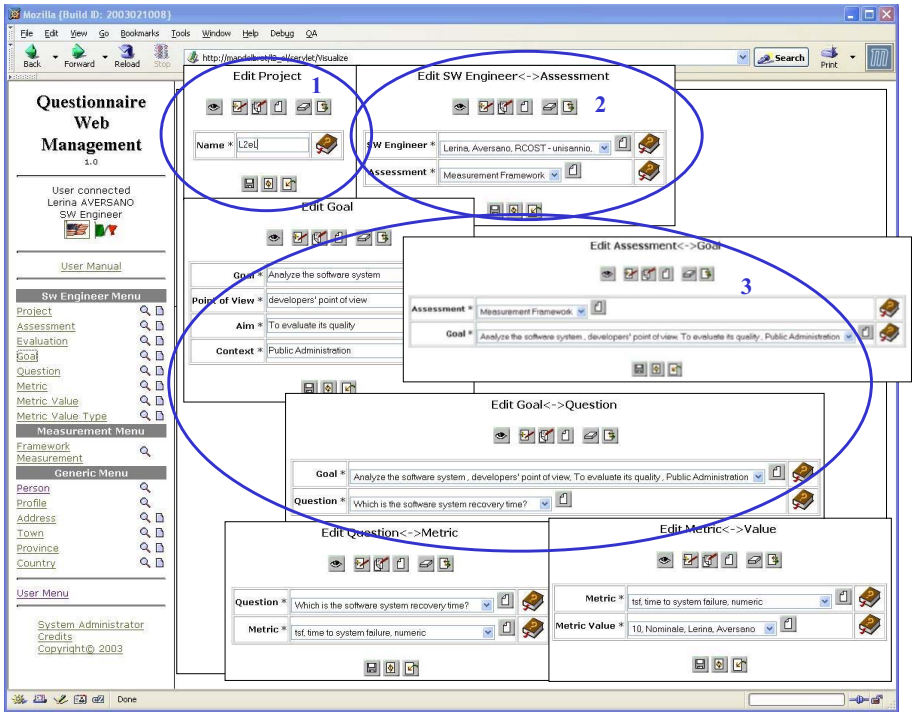

Figure 6 - Measurement Framework definition

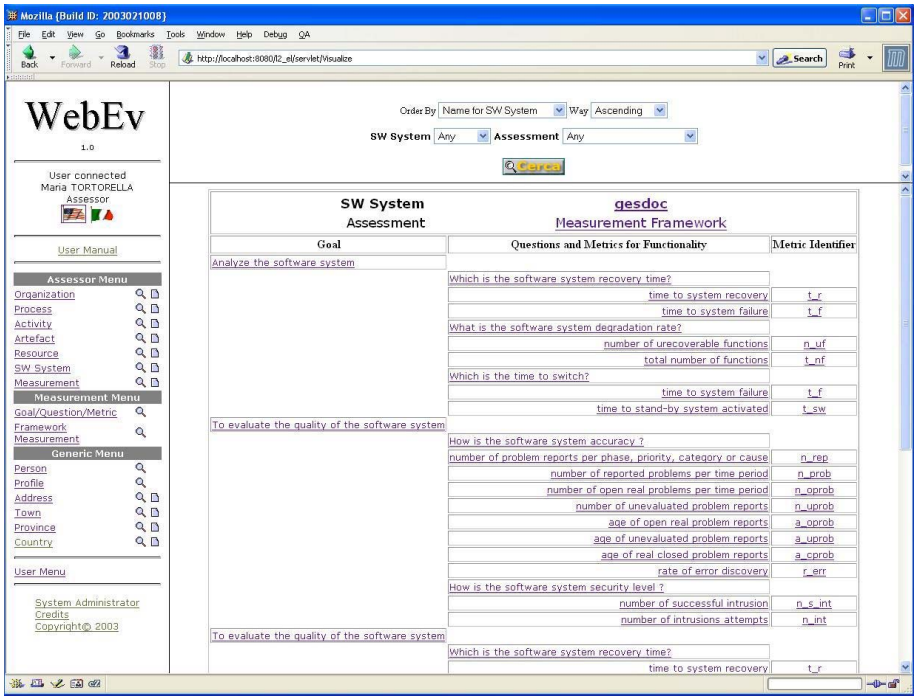

Figure 7 - Views of the Measurement Framework

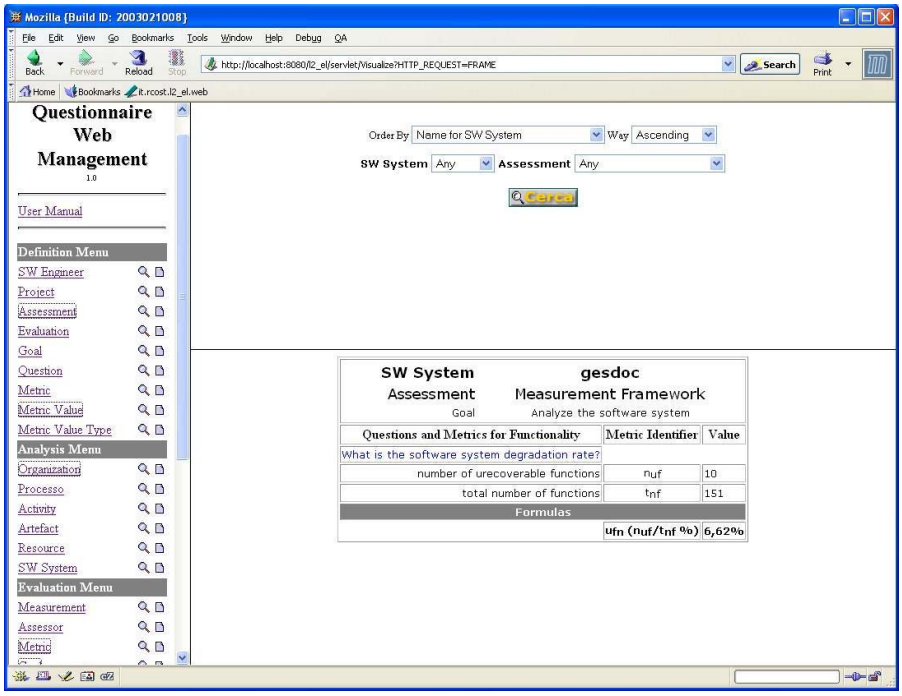

Figure 8 - Analysis of the information collected

5. Related Work

The challenge of assessing the existing software systems involves the collection of quantitative data useful to find the most adequate solution for evolving them in order to meet new requirements [1, 2]. This activity requires a decisional framework that takes into

account several factors including software value, risk analysis, and cost estimation.

Different frameworks aiming at supporting the assessment activities have been proposed in literature [12, 13]. All these approaches can be traced in the seminal paper of Verdugo [18] who defines four feasible alternatives for managing software assets: throwing away the system and redeveloping it, or reengineering, freezing, and maintaining it. Rugaber *et al*. [15] introduced a conceptual framework for selecting a strategy to move a COBOL system into an environment supporting distributed accesses. The framework includes a variety of options and a number of selection criteria that are expressed in terms of groups of factors related to the usage of the existing system, its structure and functionality, and expected evolution. Bennett *et al.* [6] proposed a decisional framework that assists organizations in making decisions about legacy systems. The model consists of exploring the purposes and objectives of the organization and producing different business scenarios for its future. Consequently, the business strategy is identified with respect to the organizational goals and on the basis of the analysis of the characteristics of the existing software systems, staff, and processes. This approach is very close to business process reengineering [8] which aims at changing the processes and information systems of an organization according to new business directions.

The construction of an effective measurement framework is based on the identification of a wide range of metrics and definition of consequent approaches to collect them [17]. The metrics can be used for all the stages of the assessment activities and formulated at various levels of abstraction. Additionally, the assessment needs of an organization can vary based on new goals and evaluation demands. [19]. For these reasons, an effective measurement framework cannot be predefined and always applicable, but it has to evolve together with the arising evolution needs. In this context, a software environment, supporting assessment activities, has to be able to manage the evolution of the measurement instruments and to be able to keep the history of both this evolution and gathered data, permitting their comparison, if required.

The assessment of a software system implies a continuous interaction among users, developers and maintainers. In addition, if the meetings are not well defined and planned, the time spent in interaction can rapidly increase. Mosvick and Nelson [14] determined that most of the time spent in meeting is wasted due to poor plan and confusing goals. The use of software environments, guiding the assessment activities, can decrease the problems connected to time management. Furthermore, collaborative Web-based technologies help to reduce also collaboration problems. The use of this kind of technologies is not new in supporting assessment processing. In [11], the authors describe how collaborative technologies can successfully support the continuous improvement effort associated with assessment activities.

WebEv exploits the collaborative technologies for facilitating interaction and collaboration during assessment activities.

6. Conclusions

The evolution towards *e*Business [10] implies the analysis of the processes together with supporting software systems. However, the assessment needs of an organization depend on the goals of the evaluation. Therefore, tools for the assessment, such as measurement framework, cannot be predefined and always applicable but have to evolve together with the arising evolution needs. The support of a software environment could be particularly useful in aiding the management of the measurement activities.

In this paper, the *WebEv* software environment has been proposed. The tool supports a measurement framework based on the GQM paradigm for assessing processes and software systems. *WebEv* facilitate cooperative work and is generally applicable to any context after its instantiation to the specific needs.

The research described addresses relevant issues for the evolution of legacy systems in the *e*Business context. Indeed, nowadays it is important to react to the central role that Internet is playing in the definition of new business relationships within and among enterprises, and, therefore, a careful evaluation of the impact on processes and legacy systems is required.

The previous experiences of the authors [1] highlighted the importance of using supporting instruments in the assessment activities. In fact, the assessment of a software system implies a continuous interaction with users, developers and maintainers. *WebEv* aims at facilitating the interaction and the data entry through the Internet.

For being really useful, it is necessary that a software environment, assisting assessment tasks, translates quantitative data in qualitative information, which can be directly interpreted by analysts and managers without considering details. To fulfill this objective, as practical activity, the authors are currently involved in the integration of *WebEv* with graphical tools to depict the calculated results. From

the theoretical point of view, the authors are studying how to integrate the measurement framework with critiquing techniques [16], in order to translate the unsatisfying parameter results into actions to be performed for evolving processes and software systems. In this context, *WebEv* will be integrated with modeling tools for annotating processes, activities and tools with the results of the critiquing process.

Finally, an instrument to be successfully used in a real context needs to be tested and refined on the basis of various needs and received feedbacks. For this reason, the measurement framework and *WebEv* have been experimented in both *vitro* and real contexts, and future work is related to the refinement of *WebEv* on the basis of the suggestions resulting from its application. With this in mind, the authors are interested in sharing information and experiences with anyone interested. Both the measurement framework and the *WebEv* environment will be transferable to researchers who want to apply them in various contexts. The *WebEv* environment is accessible through the link http://l2el.rcost.unisannio.it/.

6. Reference

[1] L. Aversano, R. Esposito, T. Mallardo and M. Tortorella, "Evolving Legacy Systems towards eBusiness", to appear in the book *Managing Corporate Information Systems Evolution and Maintenance*, Idea Group publishing.

[2] L. Aversano and M. Tortorella, "Evolving legacy system toward eLegacy system in eBusiness context", *Proc. of 7th European Conference on Software Maintenance and Reengineering*, IEEE Comp Soc. Press, Benevento, Italy, March 2003, pp. 201-210.

[3] V.R. Basili, G. Caldiera and H.D. Rombach, "The goal question metric approach", *Encyclopedia of Software Engineering*, Wiley, 1994.

[4] V.R. Basili and H.D. Rombach, "The TAME Project: Towards Improvement-Oriented Software Environments", *IEEE Transactions on Software Engineering*, vol. SE-14, n. 6, June 1988, pp. 758-773.

[5] V.R. Basili and D.M. Weiss," A Methodology for Collecting Valid Software Engineering Data", *IEEE Transactions on Software Engineering*, vol. 10, no. 6, November 1984, pp. 728-738.

[6] K.H. Bennett, M. Ramage and M. Munro, "Decision Model for Legacy Systems", *IEE Proceedings Software*, vol. 146, N° 3, June 1999, pp. 153-159.

[7] O. El Sawy, *Redesigning Enterprise Processes for e-Business*, Boston, MA, Irwin McGraw-Hill, 2001.

[8] M. Hammer and J. Champy, *Reengineering the Corporation: a Manifesto for Business Revolution*, NY: HarperCollins, 1993.

[9] ISO, *Information Technology - Software Product Quality*, ISO/IEC, 1999.

[10] R. Kalakota and M. Robinson, *e-Business 2.0: Roadmap for Success*, Boston, MA: Addison-Wesley, 2001.

[11] A. Lipp and C.Y. Carver, "Using Web Groupware and Cognitive Mapping in a CIS Department to Review and Revise the Assessment Process and Document Reasoning", *Proc. of the Thirty-third Hawai'i International Conference on Systems Sciences*, IEEE Comp. Soc. Press, Maui, Hawaii, January 2000.

[12] M.G. Mendonça and V.R. Basili, "Validation of an Approach for Improving Existing Measurement Frameworks", *IEEE Transactions on Software Engineering*, vol. 26, no. 6, IEEE Comp Soc. Press, 2000.

[13] M. Morisio, "Measurement Processes are Software Too", *Journal of Systems and Software*, vol. 49, IEEE Comp. Soc. Press, December 1999, pp. 17-31.

[14] R. Mosvick and R. Nelson, *We've Got to Start Meeting Like This!: A Guide to Successful Business Meeting Management*, Glenview, IL: Scott, Foresman, 1987.

[15] S. Rugaber and S. Doddapaneni, "The Transition of Application Programs from COBOL to a Fourth Generation Language", *Proc. of Conference on Software Maintenance CSM'93*, Montreal, Canada, IEEE CS Press, 1993, pp. 61-70.

[16] Silverman, B. G. and Mezher, "T. M. Expert critics in engineering design: lessons learned and research needs", *AI Magazine*, Spring 1992. pp. 45-62.

[17] R. van Solingen, E. Berghout, *The Goal/Question/ Metric Method: A Practical Guide for Quality Improvement of Software Development*, McGraw-Hill, 1999.

[18] G. Verdugo, "Portfolio Analysis- Managing Software as an Asset", *Proc. of International Conference Software Maintenance Management*, Software Maintenance assoc., IEEE Comp Soc. Press, New York, 1988, pp. 17- 24.

[19] D.M Weiss, D. Bennett, J.Y. Payseur, P. Tendick and P. Zhang, "Goal-oriented software assessment", *Proc. of the 24th International Conference on Software Engineering*, IEEE Comp Soc. Press, Orlando, Florida, 2002, pp. 221-231.

Acknowledgments

The authors would like to thank the anonymous referees, whose useful suggestions have enabled to improve the paper, and Ms. Antonietta Luongo for her precious support as English technical writer.