Organization-Oriented Measurement and Evaluation Framework for Software and Web Engineering Projects

Luis Olsina, Fernanda Papa, and Hernán Molina

GIDIS_Web, Department of Informatics,
Engineering School at Universidad Nacional de La Pampa,
Calle 9 y 110, (6360) General Pico, La Pampa, Argentina
{olsinal,pmfer,hmolina}@ing.unlpam.edu.ar

Abstract. A common challenge faced by many software and Web organizations is to have sound specifications of metrics and indicators metadata, and a clear establishment of measurement and evaluation frameworks and programs. In addition, organizations can succeed in this endeavour if resulting measurements and evaluations are tailored to their information needs for specific purposes, contexts, and user viewpoints. In previous works an ontology for software metrics and indicators was specified based as much as possible on the concepts of specific ISO standards. In this paper, we discuss a measurement and evaluation framework so-called INCAMI (*Information Need*, *Concept model*, *Attribute*, *Metric* and *Indicator*), which is based on that ontology. We argue this framework can be more robust and well-established than the GQM (Goal-Question-Metric) paradigm for measurement and evaluation purposes. Finally, an example is presented and the strengths and weaknesses of this framework compared with others are analysed as well.

1 Introduction

Without sound specifications of metrics and indicators metadata, and a clear establishment of measurement and evaluation frameworks and programs, organization's projects are less repeatable and controllable, and therefore more prone to fail. While many useful approaches for and successful practical examples of software measurement programs exist, the inability to specify clearly and consistently about measurement and evaluation concepts (i.e. the metadata) hampers unfortunately the progress of the software and Web engineering as a whole, and hinders their widespread adoption. For instance, the GQM paradigm is a useful, simple, purpose-oriented measurement approach that has been used in different measurement projects and organizations [1]. However, as Kitchenham *et al* pointed out [8], GQM is not intended to define metrics at a level of detail suitable to ensure that they are trustworthy, in particular, whether or not they are repeatable. Nor is GQM a robust framework for evaluation purposes as we will discuss later on.

Software and Web organizations introducing a measurement and evaluation program need to establish a set of activities and procedures to specify, collect, store, and use trustworthy metrics and indicators metadata and data. Without appropiate definitions of metrics and indicators it is difficult to ensure measure and indicator values are repeatable and comparable among projects.

D. Lowe and M. Gaedke (Eds.): ICWE 2005, LNCS 3579, pp. 42–52, 2005.

In order to bridge this gap, we have built a sound and explicit specification of metrics and indicators metadata, i.e., an ontology for this domain. Thus, the main domain concepts, properties, relationships, and axioms were explicitally specified [9, 12]. The sources of knowledge for the ontology stemmed from our own experience backed up by previous works about metrics and evaluation processes and methods [11], from different software-related ISO standards [4, 5, 7], and recognized research articles and books [2, 8, 14], among others.

In this paper, we discuss an organization-oriented measurement and evaluation framework so-called INCAMI – that stands for *Information Need*, *Concept model*, *Attribute*, *Metric* and *Indicator.* The metrics and indicators ontology and the cataloging system [10] are the rationale for our INCAMI framework. It is made up of four main components, namely: The measurement and evaluation project definition itself; the nonfunctional requirements definition and specification; the measurement design and execution, and; the evaluation design and execution. In addition, the present work also aims to bring the attention about the usefulness of this framework and strategy as well as to introduce why INCAMI can be a more robust and engineered framework than others. INCAMI_Tool, which is a prototype tool to support this framework, allows saving consistently not only metadata of metrics and indicators but also measure and indicator values for specific measurement and evaluation projects. Inter and intra-project analyses and comparisons can be now performed in a consistent way.

The rest of this article proceeds as follows. In Section 2, we discuss the main components of the INCAMI framework. In Section 3, a quality in use example for an e-learning application as proof of concepts is presented. In Section 4, related works and the strengths and weaknesses of the INCAMI framework compared with others are analysed as well. Finally, concluding remarks are drawn.

2 Framework for Measuring and Evaluating Information Needs

The approach behind the INCAMI framework is based upon the assumption that for an organization to measure and evaluate in a purpose-oriented way it must first specify the information need for a measurement and evaluation project, then it must design and select the specific set of useful metrics for measurement purposes, and lastly interpret the metrics values by means of contextual indicators with the aim of evaluating the degree the stated information need has been achieved. The strength of INCAMI resides in which not only permit recording the metrics and indicators values but also the metrics and indicators metadata (and related project metadata) in order to allow drawing consistent and traceable analyses, conclusions, and recommendations.

The conceptual framework is made up of four main components, namely: The nonfunctional requirements definition and specification; the measurement design and execution; the evaluation design and execution, and; the measurement and evaluation project definition itself. Each component is supported by many of the ontological concepts, properties, and relationships defined in [9, 12]. For instance, in the aforementioned requirement definition component, concepts such as *Information Need*, *Calculable Concept*, *Concept Model*, *Entity*, *Entity Category,* and *Attribute* intervene. Some other concepts were added to the INCAMI_Tool for design and implementation reasons. In the sequel, the first three components are illustrated.

2.1 Information Need, Concept Model, and Attributes

For the non-functional requirement specification component (i.e., the *INCAMI.requirement* package), key concepts such as *Information Need*, *Calculable Concept*, *Concept Model*, and *Attribute,* among others, intervene as shown in Fig. 1.

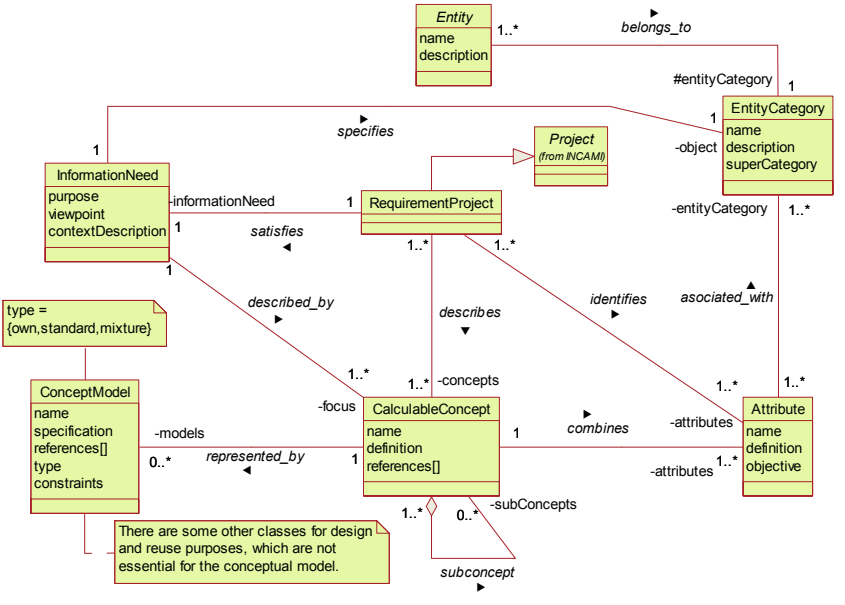

Fig. 1. Key concepts and relationships that intervene in the INCAMI.requirement package for the definition and specification of non-functional requirements

First of all, the *Information Need* for a measurement and evaluation *Project* must be agreed. Information need is defined as the insight necessary to manage objectives, goals, risks, and problems [9]. Usually, information needs come from two organizational/project sources: goals that decision-makers seek to achieve, or obstacles that hinder reaching the goals – e.g. obstacles involve basically risks and problems. The *InformationNeed* class has three properties (i.e. the *purpose*, the user *viewpoint*, and the *contextDescription*), and two main relationships with the *CalcuableConcept* and the *EntityCategory* classes respectively (as seen in Fig. 1).

A calculable concept can be defined as an abstract relationship between attributes of entities' categories and information needs [9]; in fact, quality, quality in use, cost, etc. are instances of a calculable concept. For instance, a common practice is to assess quality by means of the quantification of lower abstraction concepts such as attributes of entities' categories. The attribute can be shortly defined as a measurable property of an entity category (e.g. categories of entities of interest to Software and Web Engineering are resource, process, product, product in use, service, and project as a whole). An entity category may have many attributes, though only some of them may be useful to a given measurement and evaluation project's purpose.

To illustrate the above concepts let us consider the following example that will be expanded in section 3. A basic information need for an organization's project within a

quality assurance program may be "*understand the quality in use of the X e-learning application that supports courses tasks for pre-enrolled students*". Therefore, given the *entity category* (i.e., an e-learning application, which its *superCategory* is a product) it allows evaluators to specify an *information need*, that is to say, the *purpose* (i.e. understand), the *user viewpoint* (i.e. a novice student), in a given *context* of use (e.g. as support to a math preparatory course for pre-enrolled students, the software is installed in the Engineering School server with known bandwidth constraints), with the *focus* on a *calculable concept* (quality in use) and *subconcepts* (effectiveness, productivity, and satisfaction), which can be *represented by* a *concept model* (e.g. the ISO quality-in-use model [5]) and associated *attributes* as shown in Fig. 2.

1.Quality in Use
- 1.1. Effectiveness
  - 1.1.1. *Task Effectiveness* (TE)
  - 1.1.2. *Task Completeness* (TC)
- 1.2. Productivity
  - 1.2.1. *Efficiency related to Task Effectiveness* (ETE)
  - 1.2.2. *Efficiency related to Task Completeness* (ETC)
- 1.3. Satisfaction

Fig. 2. Specifying an instance of the Quality in Use model.

In summary, the *INCAMI.requirement* package allows the definition and specification of non-functional requirements in a sound and well-established way. Its underlying strategy is organization and purpose-oriented by information needs; evaluator-driven by domain experts, and; concept model-centred where the concept model type can be whether a standard-based model, an organization own-defined model, or a mixture of both. The INCAMI_Tool currently implements concept models in the form of requirement trees. In addition to save the measurement and evaluation project data and metadata, it also allows importing partially or totally a previously-edited requirement tree for a new project.

2.2 Metrics and Measurement

For the measurement design and execution component (i.e., the *INCAMI.measurement* package in the framework) purposeful metrics should be selected. In general, each attribute can be quantified by one or more metrics, but in practice just one metric should be selected for each attribute of the requirement tree, given a specific measurement project.

The *Metric* contains the definition of the selected *Measurement* and/or *Calculation Method* and *Scale*. For instance, the measurement method is defined as the particular logical sequence of operations and possible heuristics specified for allowing the realisation of a metric description by a measurement; while the scale is defined as a set of values with defined properties [9]. Therefore, the metric *m* represents a mapping *m*: *A*-> *X*, where *A* is an empirical attribute of an entity category (the empirical world), *X* the variable to which categorical or numerical values can be assigned (the formal world), and the arrow denotes a mapping.

In order to perform this mapping a sound and precise measurement activity definition is needed by specifying explicitly the metric's method and scale (see Fig. 3). We

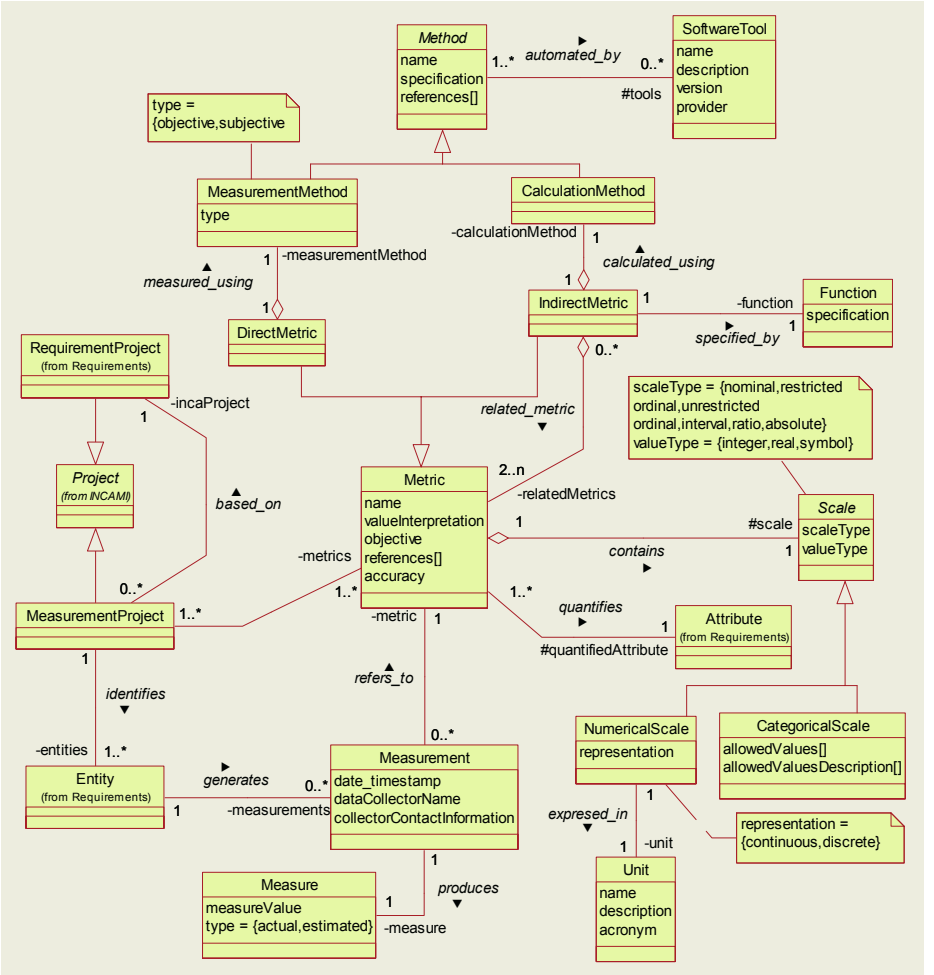

Fig. 3. Key terms and relationships that intervene in the INCAMI.measurement package for the definition of metric and measurement concepts

can apply an *objective* or *subjective* measurement method for *Direct Metrics*; conversely, we can perform a calculation method for *Indirect Metrics*, that is, when a formula intervenes. A direct metric is defined as a metric of an attribute that does not depend upon a metric of any other attribute [9].

To illustrate the above concepts, let's consider an attribute of the effectiveness characteristic from Fig. 2. Effectiveness assesses whether the tasks performed by users achieve specified goals with accuracy and completeness in a specified context of use [5]. Particularly, for the *Task Completeness* attribute, we can design a metric that specifies what proportion of the tasks is completed by a given user. The indirect metric's name is *Task Completeness Ratio*; the formula is TCR= #CT / #PT, where #CT is the number of completed tasks, and #PT is the number of proposed tasks [6]. The scale type of the indirect metric is ratio represented by a numerical scale with a

real value type. The unit description is completed tasks per proposed tasks by a user. In the formula intervenes two direct metrics with objective measurement methods (we further can specify thoroughly the metadata for each direct metric).

Once the metric was defined and selected, we can perform the measurement process, i.e., the activity that uses a metric definition in order to produce a measure's value. *Measurement* class allows to record the date/time stamp, the owner information in charge of the measurement activity, and the actual or estimated yielded value.

However, because the value of a particular metric will not represent the elementary requirement's satisfaction level, we need to define a new mapping that will produce an elementary indicator value. One fact worth mentioning is that the selected metrics are useful for a measurement process as long as the selected indicators are useful for an evaluation process.

2.3 Indicators and Evaluation

For the evaluation design and execution component (i.e., the *INCAMI.evaluation* package) contextual indicators should be selected. Indicators are ultimately the foundation for interpretation of information needs and decision-making. There are two types of indicators: *Elementary* and *Global Indicators* (see Fig. 4).

In [9] the indicator term is stated as "*the defined calculation method and scale in addition to the model and decision criteria in order to provide an estimate or evaluation of a calculable concept with respect to defined information needs*". Particularly, we define an elementary indicator as that which does not depend upon other indicators to evaluate or estimate a concept at lower level of abstraction (i.e., for associated attributes to a concept model). On the other side, we define a partial or global indicator as that which is derived from other indicators to evaluate or estimate a concept at higher level of abstraction (i.e., for subconcepts and concepts). Therefore, the elementary indicator represents a new mapping coming from the interpretation of the metric's value of an attribute (the formal world) into the new variable to which categorical or numerical values can be assigned (the new formal world). In order to perform this mapping, an *Elementary* and *Global Model* and *Decision Criteria* for a specific user information need must be considered.

Thus, an elementary indicator for each attribute of the concept model can be defined. For instance, to the 1.1.2 attribute in Fig. 2, the name of the elementary indicator can be *Task Completeness Performance Level* (TC_PL). The *specification* of the elementary model can look like this:

TC_PL = 100% if TCR = 1; TC_PL = 0% if TCR <= $X_{min;}$ where X_{min} is some agreed lower threshold as 0.45; otherwise TC_PL = TCR * 100 if X_{min} < TCR < 1

The decision criteria that a model of an indicator may have are the agreed acceptability levels in a given scale; for instance, it is *unsatisfactory* if the range is 0 to 45 percent; *marginal*, if it is greater than 45 and less or equal than 70; otherwise, *satisfactory*. Notice that a score within a marginal range indicates a need for improvement actions. An unsatisfactory rating means change actions must take high priority.

Regarding partial and global indicators, an aggregation and scoring model and decision criteria must be selected. The quantitative aggregation and scoring models aim at making the evaluation process well structured, objective, and comprehensible to evaluators. For example, if our procedure is based on a linear additive scoring model,

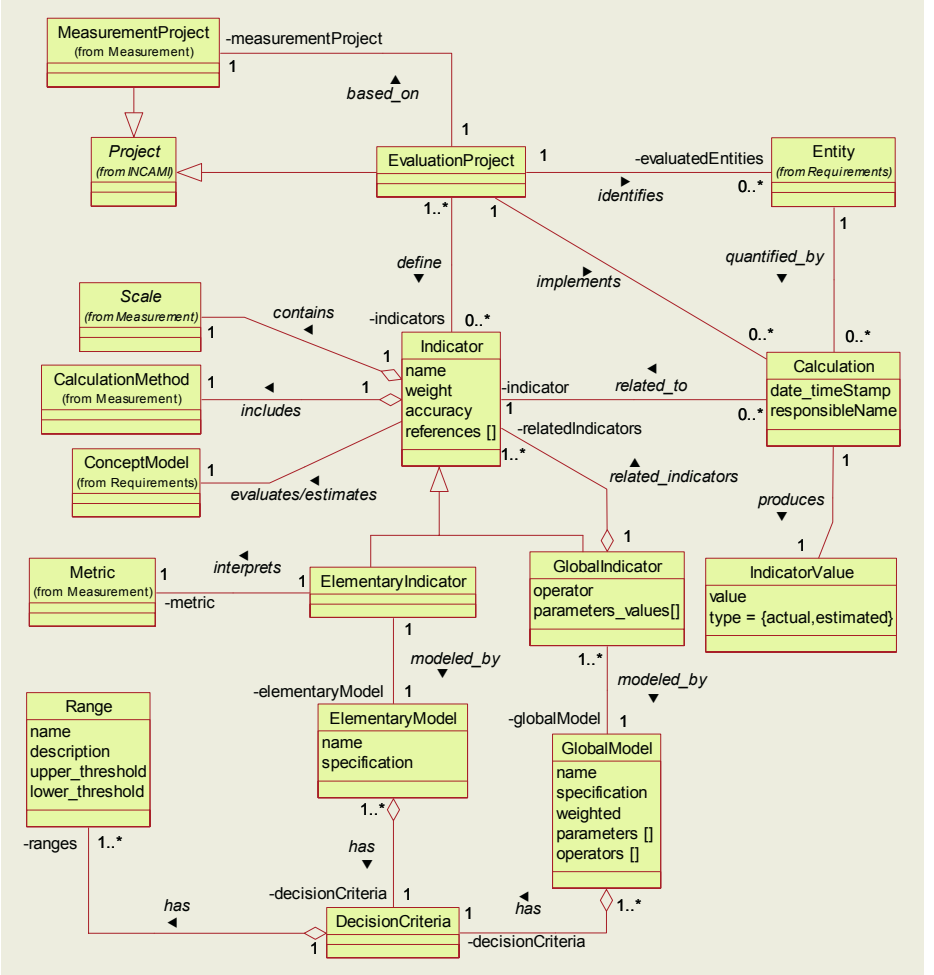

Fig. 4. Key terms and relationships that intervene in the INCAMI.evaluation package for the definition of indicators and evaluation concepts

the aggregation and computing of partial/global indicators (P/GI), considering relatives weights (W) is based on the following formula:

$$P/GI = (W_1\ EI_1 + W_2\ EI_2 + \ldots + W_m\ EI_m); \quad (1)$$

such that if the elementary indicator (EI) is in the percentage scale the following holds:

$$0 <= EI_i <= 100; \quad \text{and the sum of weights for an aggregation block must fulfil,}$$
$$(W_1 + W_2 + \ldots + W_m) = 1; \quad \text{if } W_i > 0; \text{ to } i = 1 \ldots m;$$

where m is the number of subconcepts at the same level in the tree aggregation block.

The basic arithmetic aggregation operator for inputs is the plus (+) connector. We can not use Eq. 1 to model input simultaneity or replaceability, among other limitations (see [11, 13] for a broader discussion).

Therefore, once we have selected a scoring model, the aggregation process follows the hierarchical structure as defined in the quality in use requirement tree (Fig 2), from bottom to top. Applying a stepwise aggregation mechanism, we obtain a global schema. This model let's compute partial and global indicators in the execution stage. The quality-in-use global indicator's value ultimately represents the global degree of satisfaction in meeting the stated requirements for a user viewpoint.

3 A Quality in Use Example for an e-Learning Application

Quality in use is the combined effect of the internal and external quality subconcepts (e.g., usability, functionality, reliability, and efficiency characteristics [5]) for the end user. It can be measured and evaluated by the extent to which specified users can achieve specified goals with *effectiveness*, *productivity*, *safety*, and *satisfaction* in specified contexts of use. When designing and documenting quality in use requirement, measurement and evaluation processes, at least the following information is needed:

a) Descriptions of the components of the context of use including user type, equipment, environment, and application tasks (i.e., tasks are the steps or sub-goals undertaken to reach an intended goal by a user group type), and;
b) Quality in use metrics and indicators for the intended purpose and information need.

The INCAMI_Tool allows recording all this information.

Notice that one important difference between evaluating external quality and evaluating quality in use is that the former generally involves no real users but rather experts as long as the latter always involves real end users. It is unthinkable to conduct a task testing in a real context of use without the end user participation [13].

Recently, we have conducted an e-learning case study from the quality in use perspective. The information need was established in section 2.1 in addition to the requirement tree shown in Fig. 2. Given the "*Qplus Virtual Campus*" Web application (www.qplus.com.ar/productos.htm), which since 2003 is being employed as support to a math preparatory course in the Engineering School, four tasks and six pre-enrolled students were chosen for testing purpose. Experimental design issues were considered as well, such as randomisation of the user list, among other issues.

On the other hand, in the design of the measurement process, for each attribute of the requirement tree a metric was selected. In section 2.2, the *Task Completeness Ratio* metric for the *Task Completeness* attribute was illustrated. This indirect metric specifies what proportion of the proposed tasks is fully completed by a given user. The final metric we used is the average for the six selected users; similar considerations were taken into account for designing almost all the other metrics. The exception was to the *Satisfaction* concept. For this, we designed a closed questionnaire with subcategories (as for example content, navigation, aesthetics, functions, etc), where items and scales were considered. The questionnaire's items were designed to represent attributes. Thus, we considered a direct metric for each item, and then we specified a formula to compute the indirect metric value; this formula computes the whole score for a questionnaire complete response (this procedure will be discussed in a follow-up article).

Lastly, in the design of the evaluation process, for each leaf of the requirement tree an elementary indicator is selected; it interprets the metric's value of the attribute.

In section 2.3, we illustrated the specification of the *Task Completeness Performance Level* elementary indicator (Table 1 shows the final indicators values both to elementary and partial/global ones). Likewise in other case studies [11, 13], we used a weighted, non-linear, multi-criteria scoring model for the aggregation and enactment processes.

Table 1. Global, partial and elementary indicators' values to the quality in use case study

Code	Global/Partial Indicator Name	Elementary Indicator Name	Weight	Actual Value
1.	Quality in Use Level			57.43
1.1	Effectiveness Level		0.33	59.67
1.1.1		*Task Effectiveness Performance Level*	0.5	54.17
1.1.2		*Task Completeness Performance Level*	0.5	65.58
1.2	Productivity Level		0.33	51.87
1.2.1		*Efficiency Level related to Task Effectiveness*	0.5	49.76
1.2.2		*Efficiency Level related to Task Completeness*	0.5	54.04
1.3	Satisfaction Level		0.33	87.08
1.3.1		*Calculated Satisfaction Level*	1	87.08

As a final remark, when the execution of the measurement and evaluation activities were performed for a given project, decision-makers can analyse the results and draw conclusions and recommendations with regard to the established information need. For instance, the global indicator value for the quality-in-use study (see Table 1) fell into the marginal acceptability level (i.e., a 57.43 percent of the whole requirements has been reached); this outcome means that improvement actions need to be planned. Lastly, the INCAMI_Tool allows saving consistently not only metadata for requirements, metrics and indicators but also actual or estimated values for specific projects.

4 Discussion and Related Works

To make quality assurance a useful support process to software and Web development and maintenance projects, organizations must have sound specifications of metrics and indicators metadata associated consistently to data sets, as well as a clear establishment of measurement and evaluation frameworks and programs. In addition, organizations will not willingly waste their resources if resulting measurements and evaluations are not tailored to their information needs for specific purposes, contexts, and user viewpoints.

The proposed INCAMI approach is based upon the assumption that, for an organization to measure and evaluate in a purpose-oriented way it must first specify non-functional requirements starting from information needs, then it must design and select the specific set of useful metrics for measurement purpose, and lastly interpret the metrics values by means of contextual indicators with the aim of evaluating or estimating the degree the stated requirements have been met. The INCAMI's strength resides in which not only allow recording actual or estimated values for metrics and indicators but also recording associated metadata. In this way, consistent and traceable analyses, conclusions, and recommendations can be drawn.

However, contrary to our approach that is based on an ontological conceptualisation of metrics and indicators, GQM [1] lacks this conceptual base so that it could not assure that measure values (and the associated metadata like scale, unit, measurement method, and so forth) are trustworthy and consistent for ulterior analysis among projects. Besides, GQM lacks specific concepts for evaluation in order to interpret measures. For instance, elementary and global indicators and related terms are essential for evaluation as shown above. The interpretation of measures is a weak point in GQM. Conversely, GQM is more flexible that INCAMI in the sense that it is not always necessary to have a concept model specification in order to perform a measurement project.

On the other hand, it is worthy of mention the efforts carried out by Kitchenham *et al.* [8] in the definition of a conceptual framework and infrastructure (based on the ER model) to specify entities, attributes and relationships for measuring and instantiating projects, with the purpose of analysing metrics and datasets in a consistent way. This last framework is the closest one to our research, which we tried to strengthen not only from the conceptual modeling point of view (using O-O approaches), but also from the ontological point of view including a broader set of concepts. Particularly, we deal with evaluation concepts that Kitchenham *et al.* did not.

Finally, it is also worthy of mention that there are a pair of useful ISO standards related to software measurement [7], and evaluation processes [4]. The primary aim of these standards was to reach a consensus about the issued models and processes; however, they do not constitute themselves a formal nor a semiformal ontology. Moreover, in [12] we highlighted some lack of consensus about the used terminology among these ISO documents. Despite this, in [7] there is a basic measurement information model that was also useful as a source of knowledge for our approach.

5 Concluding Remarks

Developing successful Web sites and applications with economic and quality issues in mind requires broader perspectives and the incorporation of a number of principles, models and methods from diverse disciplines. Web Engineering has a very short history compared with other engineering disciplines, but is rapidly evolving [3]. Like any other engineering science, Web Engineering is concerned with the establishment and use of sound scientific, engineering and management principles, and disciplined and systematic approaches to the successful development, deployment, maintenance and evolution of Web sites and applications within budgetary, calendar, and quality constraints. The measurement and evaluation framework discussed here can contribute by making a humble progress to the state of the art of Software and Web Engineering quality assurance processes and tools.

As a matter of fact, the INCAMI framework has also its roots in our previous researches such as the Web Quality Evaluation Methodology (WebQEM) [12], and the ontology of metrics and indicators. The underlying conceptual ground of WebQEM is now materialized by the INCAMI framework in a systematic and rigorous way. WebQEM has been used in different case studies [12, 13], and in some industrial Web quality evaluation processes.

Due to the importance of managing the acquired organizational knowledge during quality assurance projects, a semantic infrastructure that embraces organizational

memory is being considered. This can be integrated to the INAMI_Tool and framework, regarding that ontologies and Semantic Web are enabling technologies for our current research aim.

Finally, architectural and navigational design aspects of the Web-based INCAMI_Tool will be illustrated in a follow-up manuscript.

Acknowledgments

This research is supported by the UNLPam-09/F022, and the PICT 11-13623 research projects. Thanks to Guillermo Covella who was in charge of conducting the e-learning case study.

References

1. Basili V., Rombach H.D. (1989) "The TAME Project: Towards Improvement-Oriented Software Environments", *IEEE Trans. on Software Engineering*, 14(6), pp. 758-773.
2. Briand L., Morasca S. and Basili V. (2002) "An Operational Process for Goal-driven Definition of Measures", *IEEE Trans. on Software Engineering*, 28(12), pp. 1106-1125.
3. Deshpande Y., Murugesan S., Ginige A., Hansen S., Schwabe D., Gaedke M., White B (2002) "Web Engineering", *Journal of Web Engineering*, *Rinton Press* US, 1(1), pp. 61-73.
4. ISO/IEC 14598-1:1999 "International Standard, Information technology - Software product evaluation - Part 1: General Overview".
5. ISO/IEC 9126-1 (2001) "International Standard, Software Engineering - Product Quality - Part 1: Quality Model".
6. ISO/IEC DTR 9126-4 (2001) "Software Engineering - Software Product Quality - Part 4: Quality in Use Metrics".
7. ISO/IEC 15939 (2002) "Software Engineering - Software Measurement Process".
8. Kitchenham B.A., Hughes R.T., Linkman S.G. (2001) "Modelling Software Measurement Data", *IEEE Transactions on Software Engineering*, 27(9), pp. 788-804.
9. Martín M., Olsina L. (2003) "Towards an Ontology for Software Metrics and Indicators as the Foundation for a Cataloging Web System", *In proceed. of IEEE Computer Society (1st Latin American Web Congress),* Santiago de Chile, pp 103-113, ISBN 0-7695-2058-8.
10. Molina H., Papa F., Martín M. de los A., Olsina L. (2004) "Semantic Capabilities for the Metrics and Indicators Cataloging Web System". In: *Engineering Advanced Web Applications*, Matera M. Comai S. (Eds.), Rinton Press Inc., US, pp. 97-109, ISBN 1-58949-046-0.
11. Olsina L., Rossi G. (2002) "Measuring Web Application Quality with WebQEM", *IEEE Multimedia*, 9(4), pp. 20-29.
12. Olsina L., Martín M. (2004) Ontology for Software Metrics and Indicators, *Journal of Web Engineering, Rinton Press*, US, Vol 2 Nº 4, pp. 262-281, ISSN 1540-9589.
13. Olsina L., Covella G., Rossi G. (2005) "Web Quality" Chapter to appear in a Springer Book titled *Web Engineering: Theory and Practice of Metrics and Measurement for Web Development,* Emilia Mendes and Nile Mosley Eds.
14. Zuse H. (1998) *A Framework of Software Measurement*, Walter de Gruyter, Berlín-NY.