Journal of Web Engineering, Vol. 2, No.4 (2004) 262-281

ONTOLOGY FOR SOFTWARE METRICS AND INDICATORS

LUIS OLSINA, and M[a] de los ANGELES MARTÍN
GIDIS, Department of Informatics,
Engineering School at Universidad Nacional de La Pampa,
Calle 9 y 110, (6360) General Pico, La Pampa. Argentina.
{olsinal, martinma} @ing.unlpam.edu.ar

Revised October 24, 2004

Software and even more web measurement -as a younger discipline, are currently in a stage in which terminologies, models, and methods are still being defined and consolidated. It is a necessity to start reaching a common agreement between researchers and other stakeholders about primitive concepts such as attribute, metric, measure, measurement and calculation method, scale, elementary and global indicator, calculable concept, among others. There are various useful recently issued ISO standards related to software quality models, measurement, and evaluation processes; however, we observe sometimes a lack of a sound consensus among the same terms in different documents or, sometimes, absent terms. In this manuscript, we present an ontology for software metrics and indicators -based as much as possible on the concepts of those standards, which can be useful to support different assurance processes, methods and tools, in addition to be the foundation for our cataloging web system. In order to illustrate the ontology, we focus particularly on a set of intermediate representations for the domain (such as UML diagrams and tables), which were yielded during the conceptualisation step. In addition, a discussion about decisions that have been taken in choosing the terms is presented. Without sound and consensuated definition of terms, attributes, and relationships it is difficult to assure metadata consistency and, ultimately, data values are comparable on the same basis.

Key words: Metrics, Indicators, Ontology, Semantic Web, Cataloging Web System
Communicated by: R Baeza-Yates

1 Introduction

Because of increasing size, complexity, and changeable requirements, several problems have frequently been reported [4], such as unknown or bad product quality. The quality of web applications has often been assessed in an ad-hoc way, and has primarily been based on the common sense, intuition, and expertise of developers.

A common building block shared by many assessment and prediction methods that give support to assurance processes is the specification of nonfunctional requirements stemming from a sound definition and documentation of attributes and calculable concepts (e.g., quality, accessibility, productivity) and their metrics and indicators that quantify and evaluate them. In fact, great amounts of information about attributes, metrics, and indicators for different purposes and domains have been published in diverse fora and media. However, observing the rapid and chaotic growth and heterogeneity of information sources related to metrics and indicators, in addition to some lack of consensus in the terminology, it urges to provide mechanisms for agreeing, structuring, and retrieving

that information in order to be useful to diverse software and Web assurance activities, methods, and tools.

For that end, we are building a cataloging web system that basically provides different stakeholders with consultation, retrieval, and reuse mechanisms starting from a sound specification of the diverse metadata of software metrics and indicators. One key factor for the success of a such cataloging web system is the unambiguous and explicit specification of the conceptualisation for the software metrics and indicators domain formalized by an ontology. Software measurement and even more Web measurement -as a younger discipline [4], are currently in the stage where terminology and principles are still being defined and agreed on, nevertheless, the central role that software measurement and evaluation plays in the software and web engineering disciplines is actually out of discussion.

Since late 2001, we specifically started to construct a common conceptualisation for the software metrics and indicators domain, where concepts, attributes, and their relationships are explicitly specified. Hence, such explicit specification of a conceptualisation is one of the core steps for building an ontology.

The sources of knowledge for the proposed metrics and indicators ontology came from our own experience backed up by previous works in metrics, and evaluation processes and methods, from different software-related ISO standards, and also from recognized research articles. Taking into account some of his own previous works, Olsina [13] authored the *Web Quality Evaluation Method* (WebQEM), which is grounded on the design, selection, and implementation of metrics, elementary and global indicators and their associated decision criteria, starting from a calculable concept (e.g. quality), and its model. A further research was aimed to specify web metrics [14], and to develop a conceptual model just for metrics and its cataloging system [15]. Besides, other recent proposals such as the software measurement ontology documented by Genero *et al*. [6] inspired mainly in the terms of the ISO 15939 standard [10], has been a consultation source for our proposal –in addition, we are currently maintaining discussions with this and other research groups, where a final report will be drawn. Mainly, our ontology for software metrics and indicators has also been inspired on sources as ISO standards and other recognized research articles and books, namely:

- The terminology provided in the ISO/IEC 15939 standard [10], which deals with the software measurement process.
- The terminology provided in the ISO/IEC 9126-1 standard [9], which refers mainly to the terms defined in the ISO/IEC 14598-1 standard [8].
- The Zuse book [20], which deals with a validation framework of software measurement.
- The Kitchenham *et al*. [11] conceptual data model for modeling collections of software data sets.
- The Briand *et al* [2] proposal for goal-driven definition of measures.

Undoubtedly, there are various useful articles and recently-issued ISO standards related to software quality models, measurement, and evaluation processes; however, we observe very often a lack of a sound consensus among the same terms in different documents or sometimes absent terms.

In this manuscript, a discussion about decisions that have been taken during the conceptualisation step in choosing terms for the ontology is highlighted (notice this manuscript is an extended version of that published in [12]). Thus, in order to understand the real problems and the benefits of our proposal, we explain why the meaning from one proposal and not from the others was chosen, or why new terms were needed.

Our ontology contributes to the integration of the metrics and indicators related concepts that can be useful as a subontology to a measurement/evaluation process ontology, in addition to be a key requirement for our metrics and indicators cataloging web system. The ontology representation language we used for the conceptualisation step is a mixture of tables as suggested in the *Methontology* strategy [5], and the well-known UML language, which was already used for this end [18].

The rest of this article proceeds as follows. In Section 2, we represent the ontology for metrics and indicators using the *Methontology* strategy, and the UML language for the explicit specification of the conceptualisation. For ease the understanding of the selection process and the concepts, a thorough discussion about decisions taken in choosing the terms, in addition to a simple example illustrating the main terms are presented. In Section 3, we outline the cataloging web system architecture as a practical application, and we explain why the ontology is a key piece in this system; in addition, architectural and implementation issues using semantic web technologies and languages are analized as well. Related works in the area are highlighted in Section 4. Finally, concluding remarks are drawn.

2 Conceptualisation Step for the Metrics and Indicators Ontology

Let us introduce the next subsections with these recognized quotes:

"*An ontology is an explicit specification of a conceptualization*" [7]

"*An ontology may take a variety of forms, but necessarily it will include a vocabulary of terms, and some specification of their meaning. This includes definitions and an indication of how concepts are inter-related which collectively impose a structure on the domain and constrain the possible interpretations of terms. An ontology is virtually always the manifestation of a shared understanding of a domain that is agreed between a number of agents. Such agreement facilitates accurate and effective communication of meaning, which in turn leads to other benefits such as inter-operability, reuse, and sharing*" [17]

2.1 The Used Strategy for the Conceptualisation Step

A bunch of methodologies to build ontologies has been published in which different principles, design criteria, and stages for ontology development were reported. However, mainly due to the fact that Ontological Engineering is still a relatively young discipline, each work group has often employed its own methodology and formalism. So far, a lack of general consensus and standardisation about formalisms was observed (although the current impact on research about Semantic Web areas, particularly on ontologies and formalisms, will try likely to smooth things away).

One of the well-known methods for building ontologies has been the *Methontology* strategy. It proposes an effective, generally applicable method for domain knowledge model construction and validation as well. This methodology was developed by Fernandez *et al.* [5], and includes a set of stages and strategies, namely: identification of the ontology development process -where the main activities are represented going from requirement definition to maintenance of the finished product; a life cycle based on evolving prototypes; and the methodology itself.

The ontology development process distinguishes at least the following steps:

Step 1: Specification. The ontology specification´s goal is to put together a document that covers the ontology primary objective, user or application needs, granularity level, and scope. A requirements specification document is developed including the sources of knowledge as well.

Step 2: Conceptualisation. When most of the knowledge has been acquired, the ontologist has a lot of unstructured knowledge that must be organized. Conceptualisation helps to organize and structure the acquired knowledge using an external representation language that is independent of implementation languages and environments. Specifically, an informally perceived view of a domain into a semiformal specification is organized and structured using a set of intermediate representation languages and constructors (such as UML diagrams, tables, classification trees, etc.) that the domain expert and ontologist can understand.

Step 3: Implementation. It consists in implementing the conceptual model into a formal language like Ontolingua, RDF/S (*Resource Description Framework/Schema*) [19], or OWL (*Ontology Web Language*) [1]- that is a W3C initiative as well.

Step 4: Evaluation. Evaluation means to carry out a technical judgment of the ontology, its software environment, and the documentation with respect to a frame of reference (e.g. the requirements specification document).

In a general sense, a sound metrics and indicators specification, flexible documentation, consultation, and retrieval mechanisms are needed in order to contribute to the comprehension and selection process whether metrics and indicators can be useful, easy to collect, and understand. Regarding the aim of our research, we argued that a well-designed repository of metrics and indicators and a powerful cataloging system [14, 15] can be effectively used to support quality assurance processes such as nonfunctional requirement definition and specification, metrics and indicators understanding and selection, quality testing definition, amongst others. Therefore having an explicit metrics and indicators ontology was a key requirement for our cataloging system. In addition, as aforementioned, the proposed ontology could be useful as subontology to a measurement/evaluation process ontology as well.

Particularly, for the conceptualisation step to the metrics and indicators ontology we have employed, as intermediate knowledge representation schemata for the domain, a mixture of tables as suggested by the *Methontology* strategy and the well-known UML language, which was already used for this end [18].

As results of this step, the metrics and indicators knowledge using a UML conceptual model was yielded, in which the main domain concepts, properties, and relationships are specified as classes, attributes, and relationships respectively. Figure 1 shows the UML class diagram to the ontology for software metrics and indicators.

In addition, an exhaustive glossary of terms, attributes, and relationships are shown in tables 1, 2, and 3 respectively, where the terminology for the metrics and indicators ontology is explicitaly described. We use an adaptation of tables proposed by *Methontology,* (e.g. for table 1, we could have added antonyms terms).

The ontology for software metrics and indicators we are presenting was based as much as possible on the defined terms of the ISO standards. Specifically, eight terms and their exact meaning out of twenty-seven terms of our proposal were fairly used (as we discuss later in section 2.3).

In the next subsection we will describe some aspects of the conceptualisation step for the metrics and indicators ontology in a practical way.

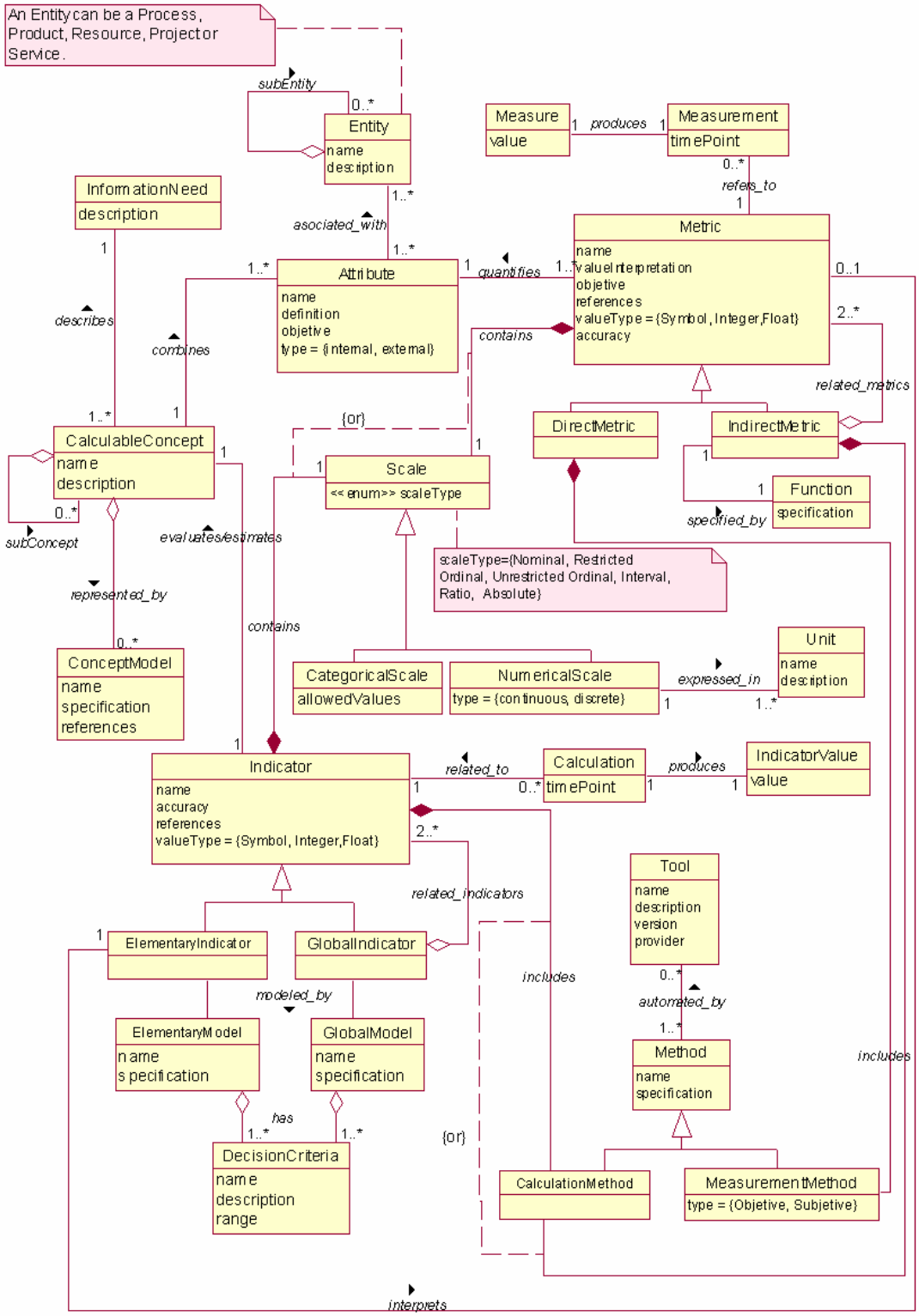

Figure 1. UML diagram to the ontology for software (and web) metrics and indicators.

Table 1. Software metrics and indicators ontology: Glossary of Concepts.

Concept Name	Synonym	Description
Attribute	Property, Feature	A measurable physical or abstract property of an entity [8].
Calculable Concept	Measurable Concept [9]	Abstract relationship between attributes of entities and information needs [10].
Calculation	Computation	Activity that uses an indicator definition in order to produce an indicator's value.
Calculation Method	Computation Method	The particular logical sequence of operations specified for allowing the realisation of a formula or indicator description by a calculation.
Categorical Scale		A scale where the measured or calculated values are categories, and cannot be expressed in units, in a strict sense.
Concept Model		The set of sub-concepts and the relationships between them, which provide the basis for specifying the concept requirement and its further evaluation or estimation.
Decision Criteria		Thresholds, targets, or patterns used to determine the need for action or further investigation, or to describe the level of confidence in a given result [10].
Direct Metric	Base Metric, Single Metric	A metric of an attribute that does not depend upon a metric of any other attribute.
Elementary Indicator		An indicator that does not depend upon other indicators to evaluate or estimate a calculable concept.
Elementary Model		Algorithm or function with associated decision criteria that model an elementary indicator.
Entity	Object	Object that is to be characterised by measuring its attributes [10].
Function	Formula, Equation	Algorithm or formula performed to combine two or more metrics.
Global Indicator		An indicator that is derived from other indicators to evaluate or estimate a calculable concept.
Global Model	Aggregation Model	Algorithm or function with associated decision criteria that model a global indicator.
Indicator		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.
Indicator Value		The number or category assigned to a calculable concept by making a calculation.
Indirect Metric	Derived Metric, Hybrid Metric	A metric of an attribute that is derived from metrics of one or more other attributes.
Information Need		Insight necessary to manage objectives, goals, risks, and problems [10].
Measure		The number or category assigned to an attribute of an entity by making a measurement [8].
Measurement		Activity that uses a metric definition in order to produce a measure's value.
Measurement Method		The particular logical sequence of operations and possible heuristics specified for allowing the realisation of a metric description by a measurement.
Method		Logical sequence of operations and possible heuristics, specified generically, for allowing the realisation of an activity description.
Metric		The defined measurement or calculation method and the measurement scale.
Numerical Scale		A scale where the measured or calculated values are numbers that can be expressed in units, in a strict sense.
Scale		A set of values with defined properties [8].
Software Tool	Software Instrument	It is a tool that automates partially or totally a measurement or calculation method.

Concept Name	Synonym	Description
Unit		Particular quantity defined and adopted by convention, with which other quantities of the same kind are compared in order to express their magnitude relative to that quantity [10].

Table 2. Software metrics and indicators ontology: Attributes Description.

Concept	Attribute	Description
Attribute	Name	Name of an attribute to be identified.
	Definition	An unambiguous description of the attribute meaning
	Objective	Goal or purpose to measuring this attribute
	Type	Attributes can be internal or external [9].
Calculable Concept	Name	Name of a calculable concept to be identified.
	Description	An unambiguous description of the calculable concept meaning.
Calculation	timePoint	Instant when a calculation is performed.
Categorical Scale	allowedValues	List of literals indicating the valid values of a categorical scale.
Concept Model	Name	Name of a concept model to be identified.
	Specification	A formal or semiformal representation of a concept model.
	References	References to bibliographical or URL resources, where additional and authoritative information of a given concept model can be consulted.
Decision Criteria	Name	Name of a decision criterion to be identified.
	Description	An unambiguous description of the decision criterion meaning.
	Range	Numerical values specifying e.g. the lower/upper thresholds for a given criterion.
Elementary Model	Name	Name of an elementary model to be identified.
	Specification	A formal or semiformal representation of an elementary model. It can be e.g. a mathematical or logical representation
Entity	Name	Name of an entity to be identified.
	Description	An unambiguous description of the entity meaning
Function	Specification	A formal or semiformal representation of a function. Synonymous: formula.
Global Model	Name	Name of a global model to be identified.
	Specification	A formal or semiformal representation of a global model. It can be e.g. a mathematical or logical representation
Indicator	Name	Name of an indicator to be identified.
	Accuracy	A quantification of the accuracy level inherent to the way of producing the value of an indicator.
	References	References to bibliographical or URL resources, where additional and authoritative information of the given indicator can be consulted.
	valueType	Type of value that an indicator can assume. It can be a symbol, integer or float.
Indicator Value	Value	Numerical or categorical result assigned to an indicator [10]. Synonymous: data.
Information Need	Description	An unambiguous textual statement describing the information needs
Measure	Value	Numerical or categorical result assigned to an attribute. Synonymous: data.
Measurement	timePoint	Instant when a measurement is performed.
Measurement Method	Type	Indicates the type of measurement method that depends on the nature of the operations used to quantify an attribute. Two types may be distinguished: subjective (quantification involving human judgement), and objective (quantification based on numerical rules) [10].
Method	Name	Name of a method to be identified.
	Specification	A formal or semiformal description of a method.

Concept	Attribute	Description
Metric	Name	Name of a metric to be identified.
	valueInterpretation	An unambiguous textual statement for helping stakeholders to understand the obtained value meaning, e.g. the closer to zero the better.
	Objective	Goal or purpose for applying the specific metric
	References	References to bibliographical or URL resources, where additional and authoritative information of the given metric can be consulted.
	valueType	Type of value that a metric can assume. It can be a symbol, integer or float.
	Accuracy	A quantifier of the accuracy level inherent to the way of producing the value of a metric.
Numerical Scale	Type	A numerical scale can be continuous or discrete.
Scale	scaleType	The type of scales depends on the nature of the relationship between values of the scale [10]. These types of scales are commonly defined: nominal, ordinal (restricted or unrestricted), interval, ratio, and absolute.
Software Tool	Name	Name of a software tool to be identified.
	Description	An unambiguous description of a software tool.
	Version	Number that indicates a software tool version.
	Provider	Indicates the name (or URL) of a software tool supplier.
Unit	Name	Name of a unit to be identified.
	Description	An unambiguous description of the unit meaning

Table 3. Software metrics and indicators ontology: Relationships Description.

Name	Description
associated_with	One or more measurable attributes are associated with one or more entities.
automated_by	One or more methods can be automated by none or several software tools.
combines	A calculable concept combines (associates) one or more measurable attributes.
contains	A metric or an indicator contains a specific scale.
describes	One or more calculable concepts are defined in order to satisfy a concrete information need. So, a calculable concept describes a concrete information need.
evaluates/estimates	An indicator evaluates/estimates a calculable concept.
expressed_in	A numerical scale must be expressed in a specific unit. (In a strict sense, there is no idea of unit for categorical scales)
has	An indicator model has one or more decision criteria.
includes	A metric includes a specific measurement and/or calculation method. An indicator includes a specific calculation method.
interprets	An elementary indicator may interpret none or one specific metric.
modeled_by	An elementary (or global) indicator is modeled by one elementary (or global) model.
produces	A measurement (or calculation) activity produces a specific measure (or indicator) value.
quantifies	One or more metrics can quantify an attribute.
refers-to	A measurement activity is related to a metric (description). None or several measurements can be made on the same metric.
related_indicators	A global indicator can be structured (aggregated) on the basis of two or more related indicators.
related_metrics	An indirect metric can be structured on the basis of two or more related metrics.
related-to	A calculation activity is related to an indicator (description). None or several calculations can be made on the same indicator.
represented_by	A calculable concept can be represented by none or several concept models.
specified_by	An indirect metric is specified by a given function (or formula).
subConcept	A calculable concept may be composed of none or several sub-concepts, which are in turn calculable concepts.
subEntity	An entity may be composed of none or several sub-entities, which are in turn entities.

2.2 An Example

In order to illustrate the main concepts, attributes and relationships, we show a simple example of information needs, calculable concepts, attributes, metrics and indicators, and related terms.

The selection and/or definition of appropiate **attributes** and **indicators** to address an **information need** starts with the specification of a **calculable concept** to be evaluated or estimated. A calculable concept is an abstract relationship between **attributes** of **entities** and information needs. For example, a simple information need may be "*evaluate the link reliability for static pages of a website*".

The calculable concept in this case is *link reliability*. Additional examples of calculable concepts includes reliability, quality in use, productivity, etc.

Considering the level of abstraction a calculable concept can be composed of other subconcepts, which could be represented by a **concept model** (e.g. ISO 9126-1 specifies a quality model based on characteristics and subcharacteristics). A calculable concept is associated to one or more attributes of entities.

An entity is a tangible or intangible object that is characterised by measuring its attributes. Types of entities of interest to software and web engineering are: Project, Product, Service, Process, and Resource. To our example, the *web page* is the (product) entity.

In addition, the attribute is a measurable physical or abstract property of an entity. An entity may have many attributes; only some of them may be of interest for a given calculable concept. For instance, in Fig. 2, attributes that are part of the *link reliability* calculable concept, and a simple concept model are shown (notice this is an exerpt of the quality model specified in [13]).

1. *Link Reliability*
    - 1.1 *Internal Broken Links* (IBL)
    - 1.2 *External Broken Links* (EBL)
    - 1.3 *Invalid Links* (IL)

Figure 2. A concept model for the Link Reliability calculable concept

For a given attribute, there is always at least an empirical relationship of interest that can be captured and represented in the formal domain by means of a **metric**, enabling us to explore the relationship mathematically and/or statistically. The metric contains the information of the defined **measurement** (and/or **calculation**) **method** an **scale**.

An attribute may be measured using different measurement methods and scales, hence one or more metrics can quantify the same attribute. (The reader can see the method definition and derived concepts likewise the scale and **unit** concepts in table 1, and the **scaleType** attribute definition in table 2).

For the above example, we can have the following direct metrics (see [14] for a definition of these metrics):

a) *Internal Broken Links Count* - #IBL for short,

b) *External Broken Links Count* - *#EBL* and,

c) *Invalid Links Count* - *#IL*.

In case we need a ratio –or percentage, with regard to the *Total of Links Count* (#TL), the next indirect metrics can be defined:

d) %IBL = (#IBL / #TL) * 100, and so forth to e) %EBL; and f) %IL.

For the above direct metrics the scale type is *absolute,* represented by a **Numerical Scale** with *Integer* **value type**; for the percentage metrics an absolute scale type can also be considered, as said by Zuse ([20], pp. 237-238). They are also represented by a Numerical Scale but with *Real* value type.

For the above a) and b) direct metrics a specific **objective** measurement method can be applied (e.g. a recursive algorithm that counts each 404 HTTP status code [14]), in addition a **software tool** can be utilized to automate the method. However for the c) direct metric, it is hard to find a tool to automate it. On the other hand, for the d), e), and f) indirect metrics, we can use a calculation method in order to perform the specified formula or **function**.

Finally, the unit of measurement is *links* for the direct metrics, or a normalized unit –as *percentage*, for the others.

One fact worth mentioning is that metrics do not represent the degree to which the specific needs are satisfied. For this reason the **indicator** concept is introduced. In table 1 is defined 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 can have an **elementary indicator** that does not depend upon other indicators to evaluate or estimate a calculable concept, and a **global indicator** that is derived from other indicators. Hence, an elementary indicator can interpret one metric, as shown in Fig 1. We will illustrate these issues for the *Link Reliability* calculable concept.

An elementary indicator for each attribute of the concept model can be defined. For example, given the 1.1 attribute in Fig. 2, *Internal Broken Links Preference or Performance Level* (IBL_P, for short) is the name of the elementary indicator.

The **specification** of the **elementary model** can look like this:

IBL_P = 100% if %IBL = 0;

IBL_P = 0% if %IBL >= X_{max};

otherwise IBL_P =((X_{max} – %IBL) / X_{max}) * 100

if 0 < %IBL < X_{max}

where X_{max} is some agreed upper threshold such as 3

The decision criteria that a model of an indicator may have are the agreed acceptability levels in the given scale; for instance, it is *unsatisfactory* if the **range** is 0 to 40 percent; *marginal*, if it is greater than 40 and less or equal than 60; otherwise, *satisfactory*. For the other attributes of Fig. 2, similar or different elementary indicator models and criteria can be defined.

A global indicator (GI) for evaluating the *Link Reliability* concept can be named as *Link Reliability Preference or Performance Level (LR_P)*. The **specification** of the **global model** can be to our example the *weighted additive scoring model* as follow:

$GI = (W_1\ EI_1 + W_2\ EI_2 + ... + W_m\ EI_m)$;

such that if elementary indicators (EI) are in the percentage scale the following is held:

$0 <= EI_i <= 100$; and the sum of weights must fulfill that,

$(W_1 + W_2 + .. + W_m) = 1$;

if $W_i > 0$; to $i = 1 .. m$ ($m = 3$, in our example).

The purpose of quantitative scoring models for indicators aggregation is to make the evaluation process well structured, and comprehensible by evaluators. In the previous additive model, weights can model the relative importance of the attributes in a given related concept or sub-concept. On the other side, agreed decision criteria for the global model such as the acceptability levels have to be stated as well.

Indicators are ultimately the foundation for interpretation of information needs and decision-making.

2.3 Discussion about Decisions Taken in Choosing some Terms

It is worthy of mention that there are various useful recently issued ISO standards related to software quality models [9], measurement [10], and evaluation processes [8]. The primarily aim of these standards was to reach a consensus about the issued models or processes together with a consensus about the used terminology; however, they do not constitute themselves a formal nor a semiformal ontology.

The ontology for software metrics and indicators we are discussing was based as much as possible on the defined terms of the ISO standards. Specifically, eight terms and their exact meaning out of twenty seven terms of our proposal were fairly used, namely: the attribute [8], decision criteria [10], entity [10], information need [10], measurable concept [10] (we used the "calculable" word instead of "measurable" as we discuss later on), measure [8], scale [8], and unit [10] terms. In addition, we employed almost the same meaning to the metric term as in [8], i.e., "the defined measurement *and calculation* method and the measurement scale". We argue that a direct metric uses just a measurement method meanwhile an indirect metric can use both measurement and calculation methods –an indirect metric is represented by a function or formula that specifies how to combine metrics.

Considering these ISO standards, we have very often observed a lack of sound consensus among the same terms in different documents or sometimes absent terms. For instance, the "metric" term is just used in [8, 9] but not in [10]. Even more, [8, 9] use the terms "direct measure" and "indirect measure" (instead of direct or indirect metric), meanwhile [10] uses "base measure" and "derived measure". In some cases we could state that they are synonymous terms, but in other such as metric, which is defined in [8] as "the defined measurement method and the measurement scale", there is no term with exact matching meaning in [10]. Furthermore, we argue that the measure term is not synonym of the metric term. The measure term defined in [8] (the meaning we adopted) as "the number or category assigned to an attribute of an entity by making a measurement" or in [10] as "variable to which a value is assigned as the result of measurement" reflects the fact of the measure as the resulting value or output for the measurement activity (or process). Thus, we claim the metric concept represents the specific and explicit definition of the measurement activity.

On the other hand, we observe some absent terms in these standards such as "elementary indicator" and "global indicator" (even though in [10] the "indicator" term is defined with similar but not equal meaning as in our proposal), as well as the "concept model", "calculation", and "calculation method" terms that are totally absent. For us, the intended objective for using a measurement method is slightly different for that of using a calculation method. The former is just intended for a measurement activity; the latter, just for a calculation activity.

Focusing us again on the metric term, as indicated elsewhere [20], the metric m represents the mapping m: A -> X, where A is an empirical attribute of one or more entities (the empirical world or domain), 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 method and scale. On the other hand, a semantic distinction between metric and indicator concepts should be raised. The indicator represents a new mapping coming from the interpretation of the metric's value (formal world) into the new variable to which categorical or numerical values can be assigned (the new formal world). In order to do this mapping a model and decision criteria for a specific user information need is considered (as illustrated in section 2.2). It is interesting to observe the definition of the "rating" term in [8] that says "the action of mapping the measured value to the appropriate rating level" in addition to the "indicator" term in the same document that says "a measure that can be used to estimate or predict another measure". However our meaning for the indicator term 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" is broader in the sense of a explicit definition of the calculation activity needed to produce an indicator value.

In order to close this discussion, we would like to remark the introduction of the terms categorical and numerical scale to our ontology (that are not explicitly specified in the ISO standard). This distinction is important to understand the unit concept associated to scales (see these classes in Fig. 1, and the definition of the unit term in table 1) Particularly, a categorical scale is a scale where the measured or calculated values are categories, and cannot be expressed in units, in a strict sense. Instead, in a numerical scale the values are numbers that must be expressed in units.

3 An Application: Metrics and Indicators Cataloging Web System (M&ICWS)

3.1 Architectural Overview of the Cataloging Web System

The metrics and indicators cataloging system will provide a Web-based collaborative mechanism for discussing, agreeing, and adding approved metrics and indicators to the repository on one hand, and a Web-based robust query functionality (based on semantic web principles) for consultation and reuse, on the other hand [15]. These subsystems are outlined in Fig. 3.

From the design of users' point of view, five user's role types with different responsibilities and access privileges were considered, namely: *Administrators, Moderators*, *Reviewers*, *Registered Users* and *Tools/Agents*. The user's role types were discussed in [14].

With regard to the *Final User*, it can be a human being or a software application (e.g., a software tool) using the repository and services. People are able to access the catalog of metrics and indicators by means of searching and browsing functionalities with read-only access permissions. The applications will be also able to access the repository in the same way, i.e., by means of Web services and the SOAP (*Simple Object Access Protocol*) interface.

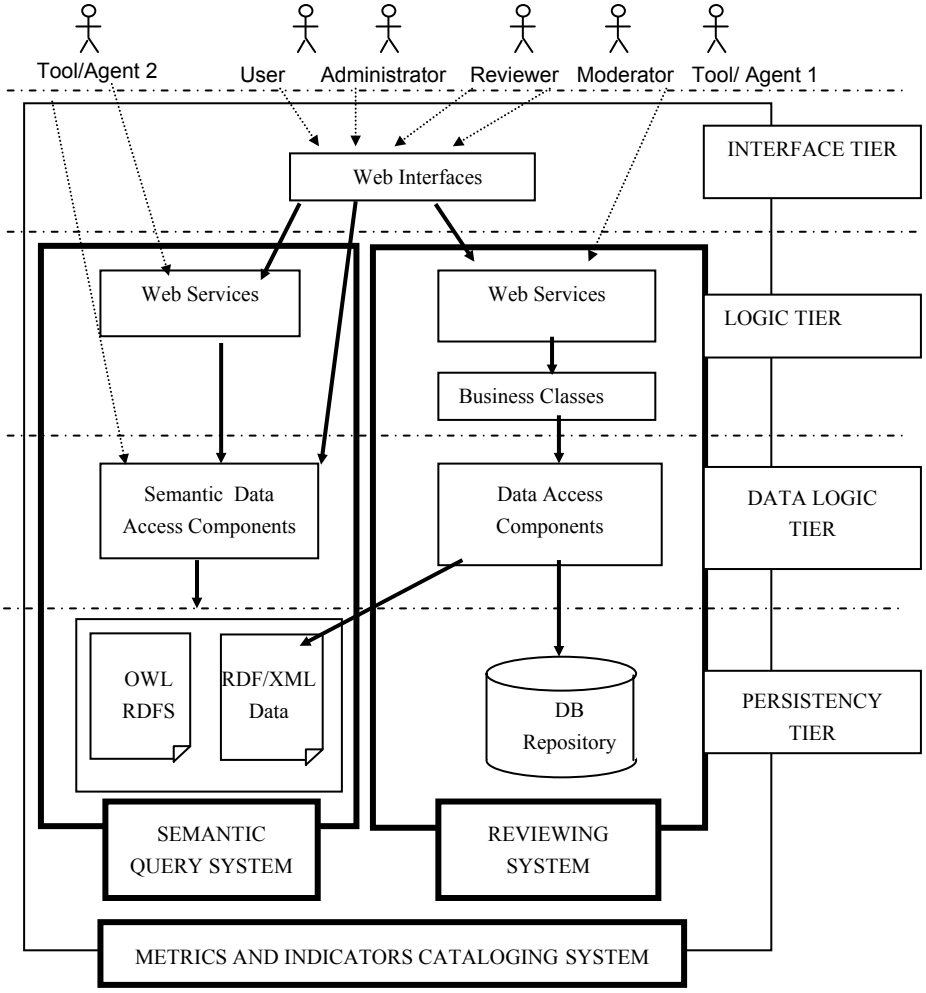

Figure 3. An Architectural view of the Metrics and Indicators Cataloging Web System

On the other side, to design the cataloging architecture, we have chosen a *multi-level* architectural style or, the so-called *n-tier* architecture. Basically, the system is composed of two subsystems, namely: the *Metrics and Indicators Reviewing System*, and the *Metrics and Indicators Semantic Query System.* The former subsystem is the responsible for the management and manipulation of the metrics and indicators for the catalog. It provides the functionality to users in order to perform the metrics reviewing process through the web, and to extract data about metrics and indicators. The latter subsystem is the responsible for the publication of cataloged metrics and indicators making use of the

semantic web principles. It does not implement any management functionality; however, it must be capable of querying on-line semantic documents and repositories. It is made up of three core tiers:

1. *Logic tier*. It implements the querying, searching, and browsing capabilities via web services. It can be used by the interface layer and by other software applications (represented by *Tools/Agents*).
2. *Data Access tie*r. It contains a set of components to accede for example to different on-line repositories and documents based on the Sesame architecture [3].
3. *Persistency* tier. A set of web pages and documents with semantic information about metrics and indicators (specified in OWL, RDFS, and RDF/XML).

To design the *Metrics and Indicators Semantic Query System* we wanted to use *semantic web* principles in order to facilitate reaching the system functionality through the web, with information processing capability. Thus, to fulfil this objective we based our system in a very known architecture for storing and querying RDF data and schema information: *The SESAME architecture*. In the next section we introduce the Sesame architecture, and how this was used for browsing and searching the M&ICWS.

3.2 Overview of the SESAME Architecture

Sesame is a Web-based architecture, which allows persistent storage of RDF data and schema information and subsequent on-line querying of that information [3]. An overview of this architecture is illustrated in Figure 4. In the follow paragraphs we outline the main components.

For persistent storage of RDF data and schema, Sesame needs a scalable repository. Because the Sesame's authors wanted to keep Sesame DBMS-independent, all DBMS-specific code was concentrated in a single architectural layer of Sesame: the *Repository Abstraction Layer* (RAL).

This RAL offers RDF-specific methods to its clients and translates these methods to calls to its specific DBMS. An important advantage of the introduction of such separate layer is that it makes it possible to implement Sesame on top of a wide variety of repositories without changing any of Sesame's other components [3].

Sesame's functional modules are clients of the RAL. Currently, there are three such modules:

- *The RQL query module:* This module evaluates RQL (*RDF Query Language*) queries -see section 3.4 for a pair of examples of RQL queries.
- *The RDF administration module:* This module allows incremental uploading of RDF data and schema information, as well as the deleting of information.
- *The RDF export module:* This module allows the extraction of the complete schema and/or data from a model in RDF format.

As the authors indicate, depending on the environment in which it is deployed, different ways to communicate with the Sesame modules may be desirable. For example, communication over HTTP may be preferable in a Web context, but in other contexts protocols such as RMI (*Remote Method Invocation*) or SOAP (*Simple Object Access Protocol*) may be more suited. In order to allow maximal flexibility, the actual handling of these protocols has been placed outside the scope of the functional modules. Instead, protocol handlers are provided as intermediaries between the modules and their clients, each handling a specific protocol.

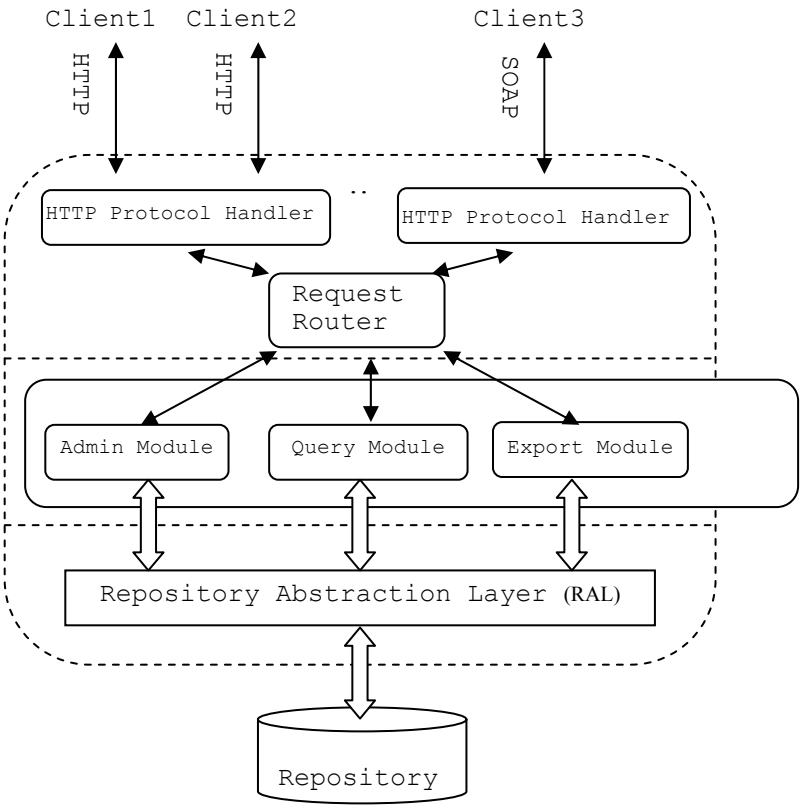

Figure 4. Sesame's architecture (adapted from [3], p. 77).

Finally, the introduction of the repository abstraction layer and the protocol handlers makes Sesame into a generic architecture for RDF/S storage and querying, rather than just a particular implementation of such a system. Adding additional protocol handlers makes it easy to connect Sesame to different operating environments.

3.3. The used Components for our M&ICWS Prototype

The Semantic Query System module of our cataloguing web system (see Fig. 3) must be capable of querying on-line semantic documents containing the metrics and indicators information. As a matter of fact, the Sesame environment is open software that provides storage and querying functions modules for RDF data and schemas. We viewed the Sesame's architecture as an appealing possible solution in terms of reusing existing APIs and tools that also provided flexibility and interoperability.

An architectural view showing the interoperation between the Sesame's architecture and the Semantic Query System is illustrated in fig. 5.

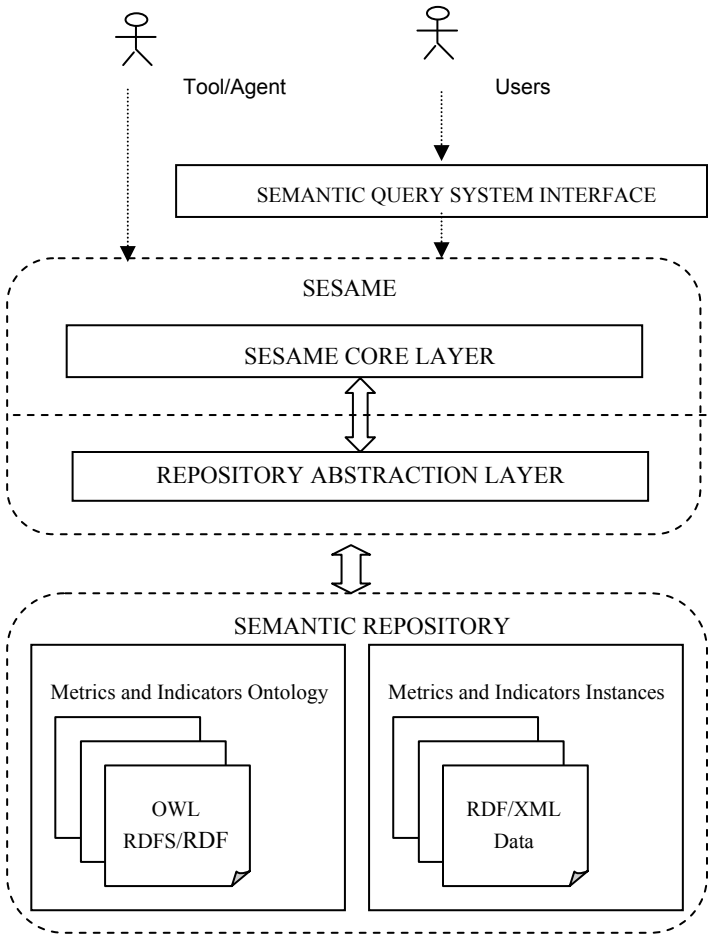

Figure 5. Metrics and Indicators Semantic Query System architecture.

This architecture allows transparent manipulation of the repository for Agents and Tools. Each application can either work with the repository through the existing Sesame modules or by calling web services for more specialized functions of searching and querying. On the other hand, the repository contains a compact body of knowledge related to metrics and indicators that could be used, manipulated, and referred as a whole. Such repository contains both ontological assertions and instance data.

The searching and querying of ontology's schema and instances are handled by the Sesame system by using the RQL language. One of the main RQL features that distinguish it from some other RDF query languages is its capability of querying both RDF schemas and data.

Currently, if more expressive reasoning is necessary (e.g. by means of the OWL language [1]) then the corresponding information should be sent to an external reasoner that processes the query and returns the answer back.

3.4 The Power of Semantic Queries

As mentioned above, taking into account the sate-of-the-art of semantic web technologies and languages, a semantic query language as RQL (*RDF Query Language*) was used in order to retrieve the semantic information from the cataloging system.

RQL is a typed declarative query language, which is based on the evaluation of path expressions on RDF graphs, featuring variables on labels for both nodes (i.e., classes) and edges (i.e., properties).

For the M&I Cataloging Web System this is a powerful feature for exploiting semantic documents and repositories of metrics, by allowing us not only to retrieve metrics and indicators data and its relations but also descriptive information (metadata) of available resources and services in the web, without human-processing intervention. Moreover, RQL has inference capabilities on hierarchies of classes and properties. This feature allows us exploiting additional information, even information that is not explicitly modeled in schemas.

In Table 4, we specify a pair of queries for the metrics and indicators cataloging system in order to illustrate some RQL features and powerness. In the first case, the RQL query acts only on RDF descriptions (instances) without the need of schemas. In the second case, the query retrieves all the properties information related to the Metric class, i.e. the labels of all edges.

Table 4. RQL query examples for the metrics and indicators domain.

Description	**Query Example**
1) Retrieves all instances of attributes and metrics for a subentity (named Webpage)	Select X, Y, Z from http://gidis.ing.unlpam.edu.ar/rdf/RDF-Metricas1#Entity{X}. http://gidis.ing.unlpam.edu.ar/rdf/RDF-Metricas1#Possess{Y}. http://gidis.ing.unlpam.edu.ar/rdf/RDF-Metricas1#IsQuantified{Z} Where X="Webpage"
2) Retrieves all properties names and their ranges to the Metric class	Select @P, range(@P) from {$C}@P Where $C ="http://gidis.ing.unlpam.edu.ar/rdf/RDF-Metricas1#Metric"

3.5 The Usefulness of the M&I Ontology for the Cataloging Web System

As previously commented, a sound metrics and indicators specification, flexible documentation, consultation, and retrieval mechanisms are needed in order to contribute to the comprehension and selection process whether metrics and indicators can be useful, easy to collect, and understand. Regarding the aim of our research, we argued that a well-designed repository of metrics and indicators and a powerful cataloging system can be effectively used to support quality assurance processes such as nonfunctional requirement definition and specification, metrics and indicators understanding and selection, quality testing definition, amongst others. Therefore having an explicit metrics and indicators ontology was a key requirement for our system. Moreover, the proposed metrics and indicators ontology has proven to be the foundation in the designing and prototipical implementation of the cataloging web system with semantic web power.

Figure 6 shows a snapshot of the system, i.e., the semantic browsing and searching capabilities accessed by a register user.

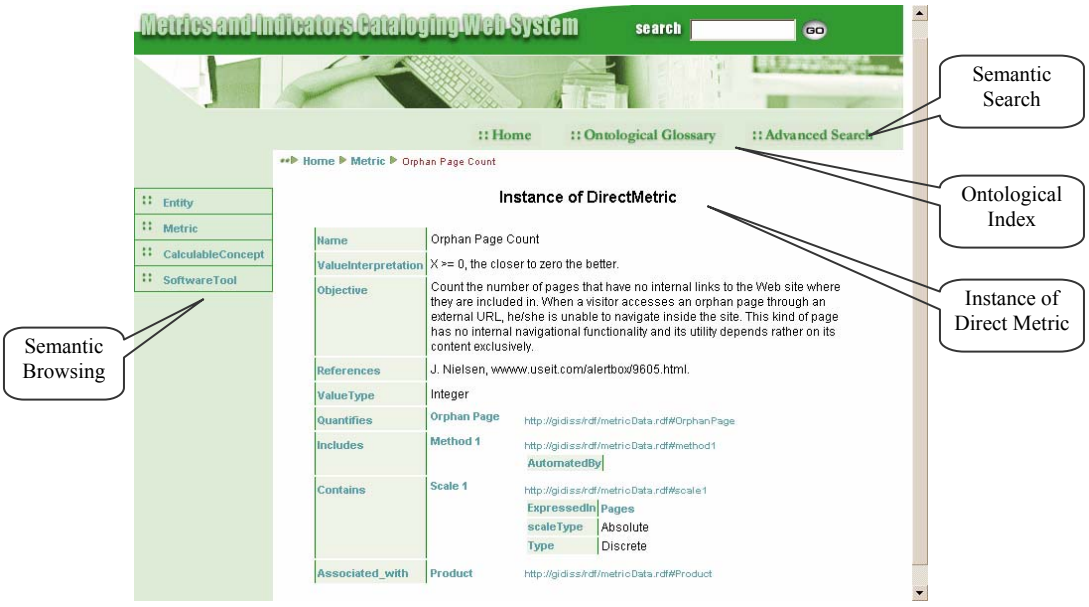

Figure 6. Browsing the M&I catalog system with semantic web power by a registered user.

Particularly, one current line of research is the designing of web services in order to allow the WebQEM_Tool be able to use the repository for retrieving different metrics and indicators metadata in the design phase of an evaluation project (WebQEM_Tool is the supporting tool for the WebQEM [13] methodology).

4 Related Works

Unfortunately, in the last software and web engineering research initiatives concerning measurement and evaluation communities almost no studies have been made towards establishing a sound definition and specification of the metrics and indicators conceptual domain formalised or semiformalised by an ontology. As previously commented, many domain researchers' articles and quoted standards cannot be considered as formal or semiformal ontologies but rather valuable sources of knowledge to building them.

At the moment of publishing our ontology, the closest related work was the recent proposal of the software measurement ontology documented by Genero *et al*. [6], which was inspired mainly in the terms of the ISO 15939 standard. This work had been a consultation source for the our proposal, nevertheless, we have aimed mainly to the metrics and indicators ontology rather than to the measurement process ontology. Our ontology could serve as a subontology for that. In addition, we embraced specific concepts, attributes and relationships that they did not (such as elementary and global indicator, calculation method, among others).

In the same direction, with the aim of reaching a consensus in the measurement ontology, we were maintaining discussions with different Ibero American research groups, where a final technical report containing the glossary of terms will be published in 2004 (we held three physical meetings in 2003). As results of the joint discussions we have not yet reach a total agreement in all the terms so far. The main discrepancies are in the metric and indicator terms. Some participants claim that the definition of the metric term is as follows: "a defined measurement form -i.e., a measurement method, or a function, or an analysis model-, and a scale in order to perform measurements of one or more attributes". Moreover, an indicator is a kind of metric (an inheritance relationship). For instance, under this consideration (likewise in [6]), the double mapping for an indicator is not clearly represented -as analysed in the Section 2.3. Despite these current and enriching discrepancies a final joint report will be issued, which could be referenced as another source of knowledge. Ultimately, it is wise to keep in mind the principles of evolveability and perfectibility of any ontology.

Finally, the REFSENO strategy [16] for specifying formally ontologies (that is rooted in *Methontology* [5]) uses an ontology example for the Goal-Question-Metrics planning artifacts. Even though this example is not directly related to our ontology, the formalism has deserved our attention.

On the other hand, regarding the metrics and indicators cataloguing web system, there is almost no similar initiative in the community, as we know. Maybe the closer work is the ZD-MIS (Zuse/Drabe Measure Information System) CD-ROM delivered with the Zuse's book [20]. But we argue that our system can be more robust in delivering the information, in agreeing metrics and indicators, and in the completeness of metadata used to model metrics and indicators.

Finally, the MiniSQUID prototype [11] has also been reported as a useful tool to support metadata and data set maintenance. As we know this system was intended just for storing metrics data and metadata but not for indicators.

5 Concluding Remarks

In this article we have shown the main concepts, attributes and relationships of the ontology for software metrics and indicators, based as much as possible on the terms of the quoted ISO standards, among other sources (as cited in the introduction section). From a practical point of view having an explicit metrics and indicators ontology was a key requirement for our cataloging web system with semantic web power, as illustrated in Section 3. However, this proposed ontology could also be useful as subontology to a measurement/evaluation process ontology. As Kitchenham *et al*. said [11], without sound and consensuated definitions it is difficult to assure metadata consistency and, ultimately, data values are comparable on the same basis.

We hope this ontology proposal (that was broadened to embrace indicators concepts with regard to our previous proposal [14]), will have a good diffusion within the software and web communities, and also can serve as a trigger for new enriching discussions as well. The stability and maturity of an ontology can also be judged by the level of agreement reached in a domain-specific international community. This involves the evolveability and perfectibility features of any consensuated knowledge building process.

Acknowledgments

This research is supported by the UNLPam-09/F022 research project. We also thank the efforts made by Hernán Molina, and Fernanda Papa in the implementation of the M&ICWS prototype, with semantic web power.

References

1. Bechhofer S., van Harmelen F., Hendler J., Horrocks I., McGuinness D., Patel-Schneider P., and Stein L., OWL Web Ontology Language Reference, W3C Recommendation, 10 February 2004, http://www.w3.org/TR/2004/REC-owl-ref-20040210/
2. Briand, L., Morasca, S. and Basili, V., "An Operational Process for Goal-driven Definition of Measures", *IEEE Transactions on Software Engineering*, 28(12), pp. 1106-1125 (2002).
3. Davies, J., Fensel, D. and Van Harmelen, F., "*Towards the Semantic Web: Ontology-driven Knowledge Management*", John Willey & Sons (2003).
4. Deshpande, Y.; Murugesan, S., Ginige, A., Hansen, S., Schwabe, D., Gaedke, M., White, B., "Web Engineering", *Journal of Web Engineering*, Rinton Press, US, 1(1), pp. 61-73 (2002).
5. Fernández López, M., Gómez-Pérez, A., and Juristo, N., "METHONTOLOGY: From Ontological Art Towards Ontological Engineering", *Proceed. of the AAAI Symposium*. University of Stanford; P.A., California, US, pp. 33-40 (1997).
6. Genero, M.; Ruiz, F.; Piattini, M.; García, F.; and Calero, C.; "An Ontology for Software Measurement", *In proceed. of SEKE'03, 15th Int'l Conference on Software Engineering and Knowledge Engineering*, San Francisco, US, pp 78-84 (2003).
7. Gruber, T. R. A translation approach to portable ontologies. *Knowledge Acquisition*, 5(2): 199-220, (1993).
8. ISO/IEC 14598-1 "International Standard, Information technology - Software product evaluation - Part 1: General Overview" (1999).
9. ISO/IEC 9126-1 "International Standard, Software Engineering - Product Quality - Part 1: Quality Model" (2001).
10. ISO/IEC 15939 "Software Engineering - Software Measurement Process" (2002).
11. Kitchenham B.A., Hughes R.T., Linkman S.G., "Modeling Software Measurement Data", *IEEE Transactions on Software Engineering*, 27(9), pp. 788-804 (2001).
12. Martín, M.; Olsina, L., "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, (2003).
13. Olsina L., Rossi G., "Measuring Web Application Quality with WebQEM", *IEEE Multimedia*, 9(4), pp. 20-29 (2002).
14. Olsina, L.; Lafuente, G. Pastor, O., Towards a Reusable Repository of Web Metrics, *Journal of Web Engineering*, Rinton Press, US, 1(1), pp. 61-7 (2002).
15. Olsina, L.; Martin, M. A.; Fons, J.; Abrahao, S.; Pastor, O., "Towards the Design of a Metrics Cataloging System by Exploiting Conceptual and Semantic Web Approaches"*, In Lecture Notes in Computer Science of Springer, Int'l Conference on Web Engineering (ICWE'03)*, Oviedo, Spain, LNCS 2722, 2003, pp. 324-333 (2003).
16. Tautz, C. and Von Wangenheim, C.; "REFSENO: A Representation Formalism for Software Engineering Ontologies", Fraunhofer IESE-Report No. 015.98/E, version 1.1, (1998)
17. Uschold, M Knowledge level modelling: concepts and terminology. *The Knowledge Engineering Review*, 13(1): 5-29, (1998).
18. Wang, X., and Chan, C.W., "Ontology Modeling Using UML", *7th International Conference on Object Oriented Information Systems Conference (OOIS'2001)*, pp. 59-68 (2001).
19. W3C, WWW Consortium, 2002, "RDF Primer", W3C Recommendation 10 February 2004, http://www.w3.org/TR/2004/REC-rdf-primer-20040210/
20. Zuse, H., *A Framework of Software Measurement*, Walter de Gruyter, Berlín-NY, (1998).