A Well-Founded Software Measurement Ontology

Monalessa Perini BARCELLOS[a,b], Ricardo de Almeida FALBO[a] and Rodrigo DAL MORO[a]

[a] *Department of Computer Science, Federal University of Espírito Santo – Brazil*
[b] *COPPE, Federal University of Rio de Janeiro – Brazil*

Abstract. Software measurement is a relatively young discipline. As a consequence, it is not well defined yet, making the terminology used diverse. In order to establish a basic conceptualization regarding this domain, in this paper we present a Software Measur ement Ontology, developed grounded in the Unified Foundational Ontology.

Keywords. Domain Ontology, Foundational Ontology, Software Measurement Ontology

Introduction

Nowadays, software measurement is a key process for software project management and software process improvement. Measurement provides organizations with the objective information they need to make informed decisions that impact their business performance. Successful organizations use measurement as part of their day-to-day activities [1]. Depending on its maturity level, a software organization can perform measurement in different ways. In the initial maturity levels, such as the levels 2 and 3 of the Capability Maturity Model Integration (CMMI), the focus is on developing and sustaining a measurement capability that is used to support management information needs [2]. In high maturity levels, such as CMMI levels 4 and 5, measurement should be performed for the purpose of statistical process control, in order to understand the process behavior [3].

Considering this context, we started to work on a strategy to support organizations to obtain and maintain measurement repositories suitable for statistical process control, as well as to perform measurements appropriately in this context. This strategy is composed of two main components [4]: (i) *Instrument for Evaluating the Suitability of a Measurement Repository to Statistical Process Control*, which goal is to evaluate existing measurement repositories and to determine their suitability to statistical process control, identifying corrective actions, when necessary; and (ii) *Body of Recommendations for Software Measurement*, which aims to supply guidelines on how to perform measurement suitable for statistical process control. However, for both the instrument and the body of recommendations to be useful, people should agree on the same conceptualization regarding the software measurement domain. Thus, we added a third component to the strategy: a Software Measurement Ontology (SMO), which

aims to capture the conceptualization involved in this domain, including traditional and high maturity aspects of software measurement. It should provide the basic vocabulary to be used in the other two components, also guiding their definition. Thus, we are looking for what Guizzardi [5] calls a domain reference ontology, i.e., an ontology that is constructed with the sole objective of making the best possible description of the domain in reality, with regard to a certain level of granularity and viewpoint.

A reference ontology is to be a special kind of conceptual model, an engineering artifact with the additional requirement of representing a model of consensus within a community. It is a solution-independent specification with the aim of making a clear and precise description of domain entities for the purposes of communication, learning and problem-solving. As a consequence, a reference ontology should be represented by an ontologically well-founded language [6]. Such a language must explicitly commit to fundamental ontological distinctions in their metamodels, given by a foundational ontology [6, 7]. Although recent research initiatives, such as [8, 9], have elaborated on the importance of foundational ontologies in the development of domain ontologies, such an approach has still not been broadly adopted.

Concerning the domain of software measurement, there are quite a few initiatives committed with ontology-based modeling and formalization of this domain, among them [10, 11]. Nonetheless, as a rule, such initiatives are not committed to the use of a foundational ontology as their basis, and, consequently, are obliged to rely on models of low expressivity.

For the reasons just mentioned, we decided to develop a Software Measurement Ontology (SMO), which is partially presented in this paper. In order to represent our ontology, we have used an ontologically well-founded UML modeling profile proposed in [7, 9]. This profile comprises a number of stereotyped classes and relations implanting a metamodel that reflects the structure and axiomatization of the Unified Foundational Ontology (UFO) [7, 9].

This paper is organized as follows: Section 1 provides a brief description of the software measurement domain; Section 2 introduces the ontological engineering approach employed in this work; Section 3 presents an overview of the SMO and presents in more details two of its sub-ontologies; Section 4 discusses related work; and finally, Section 5 concludes the paper.

1. A Brief Description of the Software Measurement Domain

Software Measurement is a primary tool for managing software life cycle activities, assessing the feasibility of project plans, and monitoring the adherence of project activities to those plans. It is also a key discipline in evaluating the quality of software products and the capability of organizational software processes [12].

The software measurement process includes, among others, the following activities: plan the measurement process, perform the measurement process, and evaluate measurement [12].

Software measurement should be driven by the information needs of the organization. Based on these information needs, first the organization has to define which entities (processes, products, resources and so on) will be the target of software measurement and which of their characteristics should be measured. Second, the organization has to define which measures are to be used to quantify those characteristics. Measures can be base measures, i.e., functionally independent of other

measures, or derived measures, which are defined as a function of other measures. Once defined the measures, procedures for data collection, analysis and reporting should be defined. Finally, criteria for evaluating the collected data and the measurement process should be defined. Once the measurement is planned, it can be effectively performed. Measurement execution involves the application of the measurement procedures to get a value for each measure of a characteristic of an entity. Once data are collected, they should be analyzed using the analysis procedure, and the results should be communicated. Finally, the measurement process and its products should be evaluated in order to identify potential improvements.

Measurement provides the information required for decision making in several levels. Project managers, for instance, can use the information provided to perform a more efficient communication, to delineate the specific goals of the projects, to identify and solve problems in advance, and to make and justify key decisions. The organizations, in turn, can use the information provided to elaborate realistic plans for the projects, to compare the performance of the current projects with their plans, to guide the investments and process improvement decisions, and to aid to foresee if the ongoing projects will achieve the goals initially established [1].

As any relatively young discipline, software measurement is not well defined yet. When analyzing different proposal and standards, it becomes apparent that the terminology used is diverse. Many times, the same concept is designated by different terms in different proposals. Others, the same term refers to different concepts. In fact, the problem of terminology harmonization still needs to be solved [11]. In this context, a software measurement ontology, providing a set of coherent concepts, with the relations between them, and axioms constraining their interpretation, is of great value.

2. The Ontological Engineering Approach Employed

In order to develop the Software Measurement Ontology, partially presented in the next section, we have employed the SABiO (Systematic Approach for Building Ontologies) method [13, 14]. This method has been tested for the last ten years in the development of a number of domain ontologies in areas ranging from Harbor Management to Software Process to Electrocardiogram domain. SABiO prescribes an iterative process comprising the following activities: (i) *purpose identification and requirement specification* that concerns to clearly identify the ontology purpose and its intended uses, i.e., the competence of the ontology by means of competence questions; (ii) *ontology capture*, when relevant concepts, relations, properties and constraints should be identified and organized; (iii) *ontology formalization*, which comprises the definition of formal axioms in First-Order Logic; (iv) *integration of existing ontologies*, which involves the search for existing ontologies with reuse and integration in mind; (v) *ontology evaluation*, for identifying inconsistency as well as verifying the truthfulness with the ontology purpose; (vi) *ontology documentation.*

It is important to highlight that competence questions play a prominent role in this method by defining the scope and purpose of the domain conceptualization being developed, and serving as a testbed for ontology evaluation, since the competence questions are the questions the ontology is supposed to answer [13].

Another important feature of this method is that it suggests the use of a conceptual model written in a UML profile and a dictionary of terms to aid communication with domain experts. Concerning the UML profile, in this paper, we do not use the UML

profile originally proposed by SABiO. Since our focus is on developing a domain reference ontology grounded in a foundational ontology, we have used an extension of the ontologically well-founded UML modeling profile proposed in [7, 9]. This profile comprises a number of stereotyped classes and relations implanting a metamodel that reflects the structure and axiomatization of the Unified Foundational Ontology (UFO) [7, 9]. UFO is a foundational ontology that has been developed based on a number of theories from Formal Ontology, Philosophical Logics, Philosophy of Language, Linguistics and Cognitive Psychology. It is composed by three main parts. UFO-A is an ontology of *endurants*. A fundamental distinction in UFO-A is between *Particulars (Individuals)* and *Universals (Types)*. Particulars are entities that exist in reality possessing a unique identity, while Universals are patterns of features, which can be realized in a number of different particulars [7]. UFO-B is an ontology of *perdurants* (events). The main distinction between perdurants and endurants is that endurants are wholly present or not, while perdurants happen in time [9]. UFO-C is an ontology of social entities (both endurants and perdurants) built on the top of UFO-A and UFO-B. One of its main distinctions is between *agents* and *objects*. Agents are capable of performing actions with some intention, while objects only participate in events [9].

A complete description of UFO falls outside the scope of this paper. However, in the sequel we give a brief explanation of its elements that are important for this paper. These elements are summarized in Figure 1.

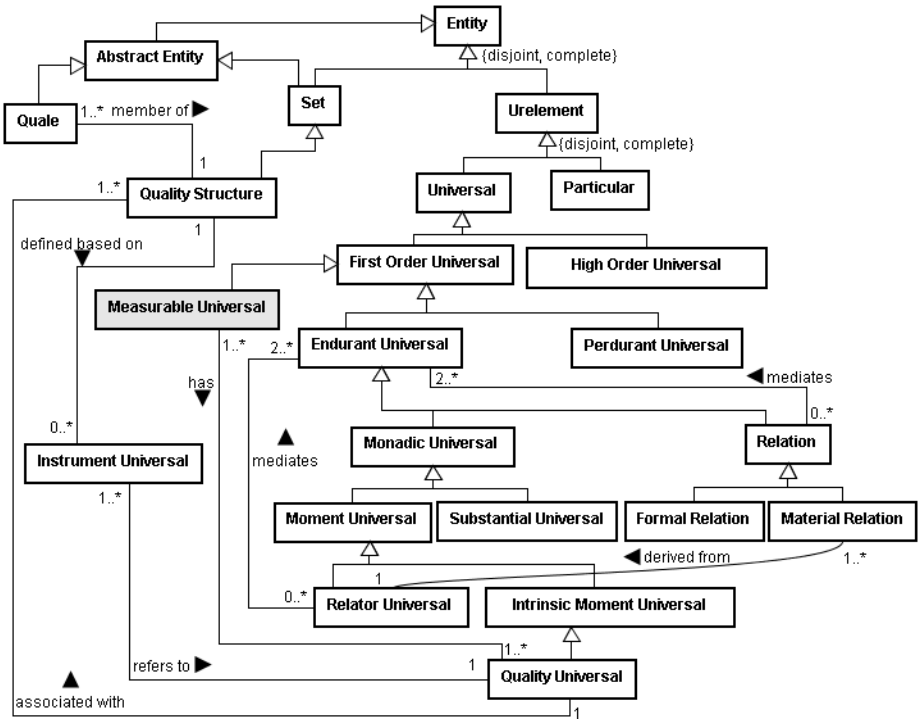

Figure 1. An UFO-A Fragment.

An entity is something perceivable or conceivable. It is the most general concept in UFO. The model depicted in Figure 1 focus on universals, i.e., from a philosophical standpoint, we are more interested in an ontology of universals, not in one of particulars. Universals are patterns of features, which can be realized in a number of

different entities. They can be first order universals, i.e., universals whose instances are particulars, or high order universals, which are universals whose instances are also universals. Concerning first order universals, a fundamental distinction in UFO is the one between endurants and perdurants (events). Endurants persist in time maintaining their identity. Perdurants, in contrast, unfold in time with their multiple temporal parts.

Endurant universals can be monadic universals or relations. Monadic universals, in turn, can be further categorized into substantial universals and moment universals (properties). A moment is an endurant that is existentially dependent of another endurant, in the way, for example, that the color of an apple depends on the apple in order to exist. Existential dependence can also be used to differentiate intrinsic and relational moments. Intrinsic moments are dependent of one single endurant (e.g., color). Relators depend on a plurality of endurants (e.g., an employment) and, for this reason, provide the material connection between these endurants. In other words, we can say that they are the foundation for material relations such as "working at". Thus, material relations require relators in order to be established. Formal relations, in contrast, hold directly between individuals.

A quality universal is an intrinsic moment universal that is associated with a quality structure. Every quality universal is associated to a quality structure, which can be understood as a measurement structure (or a space of values) in which individual qualities can take their values. A quale is a point (a value) in a quality structure. For instance, the quality universal Weight is associated to a space of values that is a liner structure isomorphic to the positive half-line of the real numbers. For the same quality universal, there can be potentially many quality structures associated with it, but a quality structure is always associated with a unique quality universal. An instrument is used to associate a quality universal with values (qualia) in a quality structure. For a given quality universal, there can be different quality structures associated with different instruments.

In Figure 1, the concept Measurable Universal is shown detached in grey. It was introduced in UFO during this work to deal with the following situation: since both endurants and perdurants can be measurable entities, we introduce this concept as the class of entities that have some quality universal (i.e., an intrinsic moment that is associated with a quality structure).

Regarding substantial universals, as shown in Figure 2, while persisting in time, substantials can instantiate several substantial universals. Some of these types a substantial instantiates necessarily (i.e., in every possible situation) and define what the substantial is. These are the types named kind (for general substantials). There are, however, types that a substantial instantiates in some circumstances but not in other circumstances. These are named phases and roles. A phase is a type instantiated in a given time period but not necessarily in all periods. A role is a type instantiated in a given context, such as the context of a given event participation or a given relation. For instance, Person can be considered a kind, while Child and Adult would be phases of person, and Student would be a role that a person plays when enrolled in some educational institution. Finally, a category is a type that classifies substantials that belong to different kinds but that share a common essential property (i.e., a property that they must not lack). For instance, Rational Entity can be judge to represent an essential property that is common to all its instances and then it is a category.

Taking into account kinds, an important distinction in UFO is between agents and objects. According to UFO-C, an object kind is a non-agentive substantial universal. Its instances (objects) do not act. They can only participate in actions. Object kinds can

also be further categorized into physical (e.g., Book) and social (e.g., Language). A normative description kind is a social object kind whose instances define one or more rules/norms recognized by at least one social agent (e.g., a method describing a set of directives on how to perform some activity within an organization).

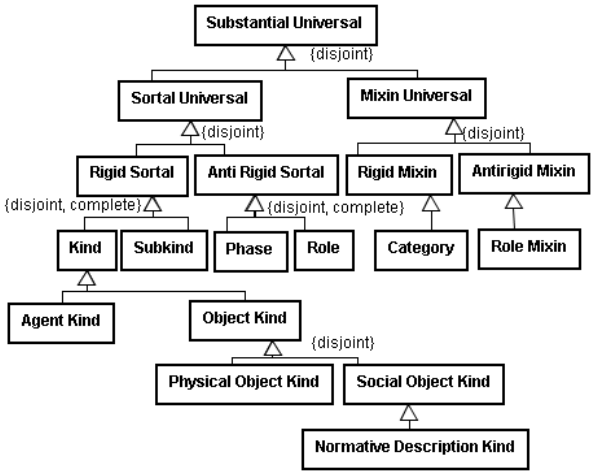

Figure 2. UFO´s Fragment related to Substantial Universals.

3. The Software Measurement Ontology (SMO)

According to SABiO, first we identified the SMO purpose. We want to capture the conceptualization involved in measurement in general, but also more specific aspects concerning measurement in high maturity levels. The main intended use of the SMO is to establish a common vocabulary to guide the definition of the strategy cited in the beginning of this paper. In this sense, the SMO should bring in a software measurement theory that is independent of specific applications. Although this is the main intended use of the SMO, we foresee its use also as a conceptual specification for developing and integrating supporting tools to the Measurement Process.

Since the software measurement domain is strongly related to the domains of software processes and organizations, we looked up to ontologies in these domains. We decided to use the software process ontology, which is grounded in UFO, described in [9]. In fact, the reuse of this ontology leads us to choose UFO as the foundational ontology to be use in our work. Concerning the domain of software organizations, we decided to reuse the software organization ontology proposed by Villela et al. [15]. This ontology, however, was not developed grounded in a foudational ontology, and thus, first, we reengineered it [16].

Since the scope of the SMO is very complex, we applied a decomposition mechanism allowing building the ontology in parts. The adopted strategy was to define sub-ontologies, starting from a core ontology, considering measurable entities and measures. From this core ontology, other five sub-ontologies were defined, each one treating some aspect as the way an organization performs measurement evolves. Thus the core ontology considers measurement in both initial and high maturity levels. The others sub-ontologies are more focused on aspects that are more important for measurement in high maturity levels. Figure 3 shows the sub-ontologies developed as

UML packages, and their relationship as dependency relationships. In this figure the dependency relationships indicate that concepts and relations of a sub-ontology / ontology are used by another.

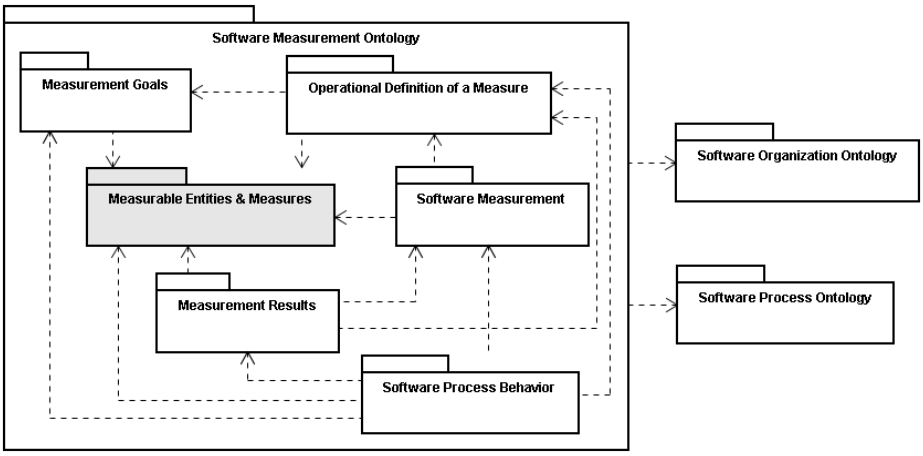

Figure 3. Sub-ontologies of the Software Measurement Ontology.

The Measurable Entities & Measures Sub-ontology treats the entities that can be submitted to measurement, their properties that can be measured and the measures used to measure them. As said before, it is the core of the SMO, and its concepts and relations are used by all the other sub-ontologies. The Measurement Goals Sub-ontology deals with the alignment of measurement to organizational goals. The Operational Definition of Measures Sub-ontology addresses the detailed definition of aspects related to measure collection and analysis established by an organization according with its measurement goals. The Software Measurement Sub-ontology refers to the measurement per se, i.e., collecting and storing data for measures. The Measurement Results Sub-ontology handles the analysis of the collected data for getting information supporting decision making. Finally, the Software Process Behavior Sub-ontology refers to applying the measurement results in the analysis of the behavior of the organizational software processes.

For each one of the sub-ontologies we accomplished the other steps of SABiO. As a result, the SMO ontology is composed of: (i) sets of competency questions, describing the sub-ontologies' requirements specification; (ii) structural conceptual models written in the ontologically well-founded UML profile defined based on UFO, capturing the concepts and relations of the sub-ontologies; (iii) a dictionary of terms, providing descriptions for each term in the ontology in natural language; and (iv) first order logic axioms, capturing constraints that are not captured by the conceptual models developed. With the sub-ontologies in hand, we needed to evaluate them. Since the SMO was not implemented in a programming language, such as OWL, we had to make the evaluation manually. For each competency question, we created an entry in a table showing the concepts, relations and axioms used to answer it. As a complementary way to evaluate the ontology, we also instantiate it with elements extracted from measure repositories of organizations.

Due to space limitation, in this paper it is not possible to present all sub-ontologies. Thus, in the rest of the paper we discuss part of the Measurable Entities & Measures Sub-ontology, presenting some of its competency questions, conceptual models and

axioms. Also, part of its evaluation is shown. To this sub-ontology, several standards were considered, such as ISO/IEC 15939 [12], ISO/IEC 9126 [17], ISO/IEC 15504 [18], CMMI-Dev [2], Practical Software Measurement (PSM) [1], as well as other ontologies regarding the same domain, in special the one presented in [11].

As said before, the Measurable Entities & Measures Sub-ontology regards elements of entities (such as processes, artifacts and so on) that can be measured and the measures used to measure them. Basically, this sub-ontology should be able to answer, among others, the following competency questions:

CQ1. Which are the types of entities that can be measured?
CQ2. What is the type of a measurable entity?
CQ3. Which are the measurable elements of a measurable entity?
CQ4. Which are the measurable elements that characterize all measurable entities of the same type?
CQ5. Which measurable elements can be directly measured?
CQ6. From which other measurable elements can an indirectly measurable element be measured?
CQ7. Which measures can be used to quantify a measurable element?
CQ8. Which is the nature of a measure considering its dependency with others?
CQ9. Which measures should be measured to compute a derived measure?
CQ10. What is the scale of a measure?
CQ11. What is the type of a scale?
CQ12. Which are the values of a scale?
CQ13. What is the measure unit of a measure?
CQ14. Which are the measurement procedures that are applied to a measure?
CQ15. Which are the measurement analysis procedures that are applied to a measure?
CQ16. What are the calculation formulas used in a measurement procedure?
CQ17. Which are the measures involved in a measure calculation formula?
CQ18. Which measures are correlated to a given measure?

Figure 4 shows the conceptual model that addresses the competency questions CQ1 to CQ 12. In this figure, a **Measurable Entity** is anything that can be measure, such as processes, artifacts, projects and resources, and thus correspond to a *Measurable Universal* in UFO. Measurable entities can be classified according to types (**Measurable Entity Type**). For instance, process is a type of measurable entity. Although not shown in Figure 4, we use the Measure Entity Type as a specialization criterion, giving rise to a hierarchy of measurable entities. Thus, Measurable Entity Type is a *High Order Universal* in UFO, since its instances are, in fact, universals, more specifically subclasses of Measurable Entity.

As defined in UFO, *Measurable Universals* must have *Quality Universals*. In the case of SMO, these quality universals are **Measurable Elements**. All measurable entities of the same type should be characterized by the same measurable elements. For instance, projects can have their size measured. Thus, we say that a Measurable Element characterizes a Measurable Entity Type, and the following axiom holds: If a measurable entity *men* is an instance of the measurable entity type *t* and *t* is characterized by the measurable element *mel*, then *mel* also characterizes *men* (**A1**).

$$(\forall men \in MeasurableEntity, t \in MeasurableEntityType, mel \in MeasurableElement)$$
$$(instanceOf(men, t) \wedge characterizes(mel, t) \rightarrow characterizes(mel, men))$$

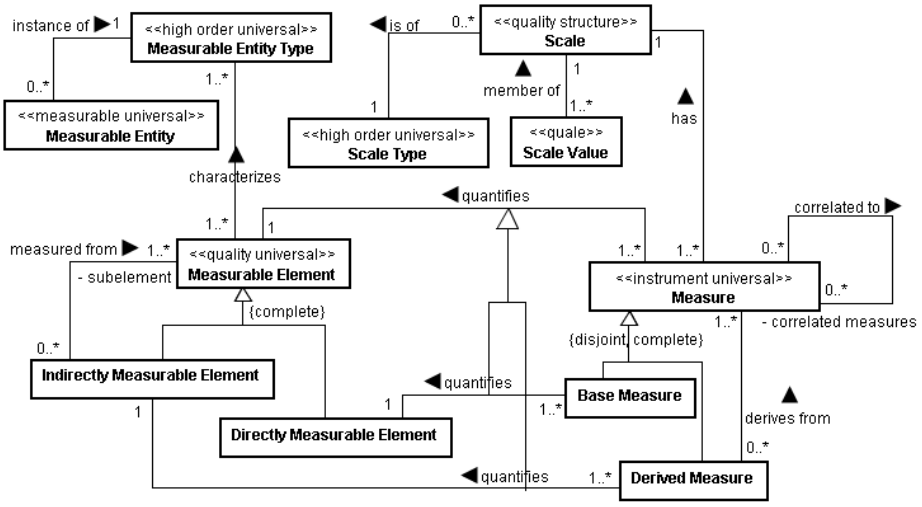

Figure 4. Fragment I of the Measurable Entities & Measures Sub-ontology.

Measurable Elements can be directly or indirectly measured. **Indirectly Measurable Elements** are measured by means of other measurable elements, said its sub-elements. The following axiom constrains the measurable elements that can be sub-elements of an indirectly measurable element: If a measurable element *mel* is sub-element of a indirectly measurable element *ime*, then *mel* and *ime* should characterize the same measurable entity type *t* (**A2**).

$$(\forall mel \in MeasurableElement, ime \in IndirectlyMeasurableElement)\ (subelement(mel, ime) \rightarrow (\exists t \in MeasurableEntityType)\ (characterizes(mel, t) \wedge characterizes(ime, t))$$

Moreover, the relation "*measured from*" is transitive, that is, if a measurable element *mel* is sub-element of an indirectly measurable element *ime1*, and *ime1* is sub-element of another indirectly measurable element *ime2*, then *mel* is also sub-element of *ime2* (**A3**).

$$(\forall ime1, ime2 \in IndirectlyMeasurableElement, mel \in MeasurableElement)\ ((subElement(mel, ime1) \wedge subElement(ime1, ime2) \rightarrow subElement(mel, ime2))$$

As discussed in Section 2, according to UFO, quality universals are associated with quality structures and instruments are used to associate a quality universal with values (qualia) in a quality structure. Based on this fragment of UFO, in the SMO, we consider a **Measure** as been an *Instrument Universal* that allows associating **Measurable Elements** (*Quality Universal*) with **Values** (*Quale*) of a **Scale** (*Quality Structure*). Thus a Measure quantifies a Measurable Element and has a Scale composed by Scale Values. Moreover, a Scale is of a **Scale Type**, which is a *High Order Universal*, since its instances are, in fact, universals, more specifically subclasses of Scale (not shown in the model). Measures can be correlated to other measures, indicating, for instance, that they are related to the same goal, or that they have a cause-effect relationship and so on. Finally, Measures can be classified into **Base Measures**, which are functionally independent of other measures and used to quantify Directly Measurable Elements, and **Derived Measures**, which are defined as a function of other measures and used to quantify Indirectly Measurable Elements. Concerning the relationships between the subtypes of measures and measurable elements, the following

axiom holds: If a derived measure *dme* quantifies an indirectly measurable element *ime* and *dme* derives from another measure *m*, then exists a measurable element *me* that is quantified by *m* and that is sub-element of *ime* (**A4**).

$$(\forall dme \in DerivedMeasure, m \in Measure, ime \in IndirectlyMeasurableElement)$$
$$(quantifies(dme, ime) \wedge derivesFrom(dme, m)) \rightarrow (\exists me \in MeasurableElement$$
$$(quantifies(m, me) \wedge subElement(me,ime)))$$

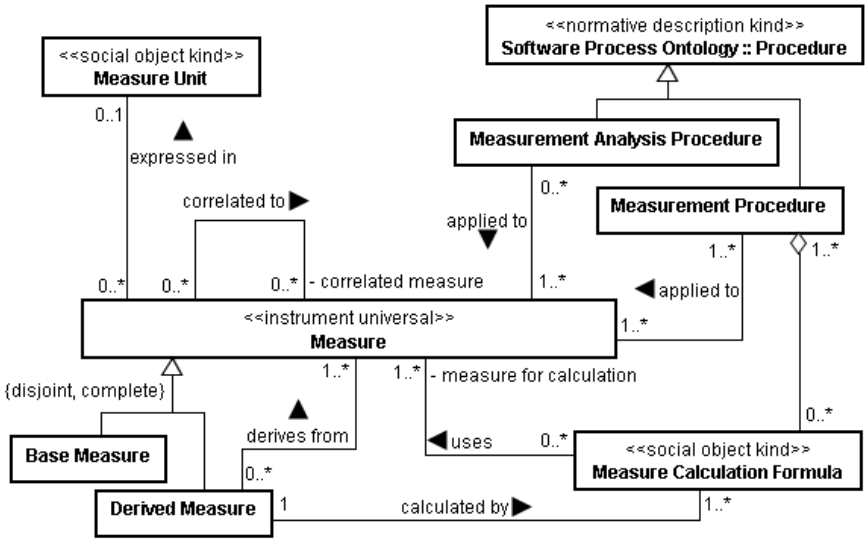

Figure 5. Fragment II of the Measurable Entities & Measures Sub-ontology.

Figure 5 shows the conceptual model that addresses the competency questions CQ13 to CQ 18. This figure shows that a **Measure** is expressed in a **Measure Unit**, which is a *Social Object Kind* in UFO. In order to guide measure collection and analysis, **Measurement Procedures** and **Measurement Analysis Procedures** are defined. Both are **Procedures** (a concept reused from the Software Process Ontology described in [9]), which is a *Normative Description Kind* in UFO, since a procedure defines one or more rules/norms recognized by at least the organization that adopts it. **Derived Measures** are calculated by **Measure Calculation Formulas**, which, in turn, uses other measures as measures for calculation. Those relations are constrained by the following axiom: if a derived measure *dme* is calculated by a measure calculation formula *mcf* that has a measure *m* as a measure for calculation, then *dme* is derived from *m* (**A5**).

$$(\forall dme \in DerivedMeasure, m \in Measure, mcf \in MeasureCalculationFormula)$$
$$(calculatedBy(dme, mcf) \wedge measureForCalc(m, mcf) \rightarrow derivesFrom(dme, m))$$

Concerning correlation between measures, when a derived measure *dme* is calculated by a measure calculation formula *mcf* that has another measure *m* as a measure for calculation, then *m* is a correlated measure of *dme* (**A6**).

$$(\forall dme \in DerivedMeasure, m \in Measure, mcf \in MeasureCalculationFormula)$$
$$(calculatedBy(dme, mcf) \wedge measureForCalc(m, mcf) \rightarrow correlatedMeasure(m, dme))$$

Finally, Measurement Procedures can aggregate Measure Calculation Formulas. Furthermore, if a derived measure *dme* is calculated by a measure calculation formula

mcf, then there should be a measurement procedure *mp* applied to *dme* such as *mcf* is part of *mp* (**A7**).

$(\forall dme \in DerivedMeasure, mcf \in MeasureCalculationFormula)\ (calculatedBy(dme, mcf) \rightarrow (\exists mp \in MeasurementProcedure\ (appliedTo\ (mp,dme) \wedge partOf(mcf,mp))))$

As said before, to evaluate the ontology, we related the concepts, relations and axioms of the SMO to the competency questions answered by them, as well as we used individuals (extracted from measure repositories of organizations) to evaluate if the ontology was actually able to represent concrete situations of the real world. Table 1 shows part of the table developed to evaluate the ontology, considering competency questions CQ 6 and CQ 9.

Table 1. Evaluating the SMO

CQ	Concept 1	Relation	Concept 2	Axioms
6	Indirectly Measurable Element	is subtype of	Measurable Element	A2, A3
	Indirectly Measurable Element	measured from	Measurable Element (sub-element)	
	Measurable Element	characterizes	Measurable Element Type	
9	Derived Measure	derives from	Measure	A4
	Indirectly Measurable Element	is subtype of	Measurable Element	
	Directly Measurable Element	is subtype of	Measurable Element	
	Measure	quantifies	Measurable Element	
	Indirectly Measurable Element	measured from	Measurable Element	

Concerning CQ 6 – From which other measurable elements can be an indirectly measurable element measured? – the relation "*measured from*" is the main responsible for answering this question. However, this relation is constrained by the types of the indirectly measurable element and its sub-elements (A2). Also, axiom A3, which says that this relation is transitive, acts on the answer of this competency question.

Regarding CQ 9 – Which measures should be measure to compute a derived measure? – the relations "*derives from*" and "*quantifies*" (including its subtypes not shown in Table 1) are the main responsible for answering this question. These relations should respect axiom A4 that establishes a relationship between the measures used to compute a derived measure and the sub-elements from which the corresponding indirectly measurable element is measured from.

Finally, some instantiations were also used to evaluate the SMO. In this paper we discuss measurement aspects related to measuring an organizational Requirements Management Process. The Requirements Management Process of the Organization X (a Measurable Entity) is a Requirements Management Process (Measurable Entity Type), which is characterized, among others, by Requirements Stability (Indirectly Measurable Element). Requirements Stability is measured from Agreed Requirements and Changed Requirements (both Directly Measurable Elements). Requirements Change Rate is a Derived Measure used to quantify Requirements Stability. Its scale is the real interval ranging from 0 to 1 and, thus, it is a Ratio Scale (Scale Type). The Requirements Change Rate is derived from two base measures: Number of Agreed

Requirements and Number of Changed Requirements, which in turn quantifies the elements Agreed Requirements and Changed Requirements, respectively. The Measure Calculation Formula applied to compute the Requirements Change Rate is: Requirements Change Rate = Number of Changed Requirements / Number of Agreed Requirements. Number of Agreed Requirements and Number of Changed Requirements are correlated measures of Requirements Change Rate. The measurement procedure used to measure Requirements Change Rate is to compute the measure calculation formula described above in the period comprised between two Requirements Specification Agreements. The measurement analysis procedure establishes that the value obtained should be analyzed in order to see if it diverges significantly from values obtained in similar projects of the organization. If this is true, the causes should be investigated and corrective actions should be accomplished.

4. Related Work

As said in the introduction of this paper, there are quite a few initiatives related to developing ontologies for the Software Measurement domain. The two most known are the one proposed by Martín and Olsina [10], called here MO-ontology, and the one proposed by Bertoa, Vallecillo and García [11], called here BVG-ontology. Our Measurable Entities & Measures Sub-ontology has many commonalities with these two ontologies, since they are inspired in some common sources, such as ISO/IEC 15939 [12]. Nevertheless, both initiatives are not committed to a foundational ontology.

Concerning the similarities, there are many common concepts, such as concepts related to measurable entities (entities in the MO and BVG-ontology), measures (metrics in the MO-ontology, and measures in BVG-ontology), measurable elements (attributes in the MO and BVG-ontology), scale and procedures (methods in the MO-ontology and approaches in the BVG-ontology). However, there are also many differences. Entities, both in MO and BVG-ontology, are considered to be composed of sub-entities. The whole-part relation used, however, is not in conformance with the whole-part theory discussed in UFO. Moreover, we do not agree with this relation. For example, people acting as a human resource role (role in the sense of UFO) can be considered a measurable entity. But they do not have sub-entities. Thus, in our SMO ontology we decided not to include this relation.

In MO-ontology, Numerical Scale (a subtype of Scale) is expressed in one or more Units. For us (and also in the BVG-ontology), measures are expressed in measure units, not scales. Further considering scales, UFO says that a quality structure has one or more qualia as members. Thus, since scales are quality structures, it is essential to model scale values as a concept in the SMO, what is not done in both MO and BVG-ontology.

Another problem is the concept of Indicator. In our view, although not discussed in this paper, an Indicator is a role (in the sense of UFO) that a measure plays when used to indicate the achievement of a goal. In MO-ontology, there is no relationship between measure and indicator. In BVG-ontology Indicator is considered a sub-type of Measure, but there isn't anything saying that it is a role. The consequence is that this conceptualization does not capture that all indicators must be associated to a goal, since it is this relationship that characterizes the role itself. Although not presented in this paper, in our SMO ontology, an Indicator is always associated with an Objective,

which in turn is related to an Information Need. In the BVG-ontology, an indicator can be (or not) associated with an Information Need.

Both MO and BVG-ontologies use the concept of "Measurable Concept" with the following definition borrowed from ISO/IEC 15939: "Abstract relationship between attributes of entities and information needs". In our view, this definition does not make sense. In the best case, according to UFO, it would be a relator, since relators mediate the relata of a material relation. However, this concept is not clear, and thus we prefer not to include it in our SMO.

In our SMO we distinguish between directly and indirectly measurable elements. Both MO and BVG-ontologies do not do that. Moreover, considering the whole SMO (and not only its core sub-ontology), our SMO treats many other aspects that are not the target of those ontologies. Also, both MO and BVG-ontologies do not discuss constraints, and consequently axioms formalizing them.

In fact, there are several differences between our SMO and the MO and BVG-ontologies. Due to the limitations of space, in this paper it is not possible to discuss all of them in details. However, as a main conclusion, we claim that the use of UFO as a foundational ontology for supporting the development of a domain reference ontology was of great value and a distinction point of our work when compared with these two others. It allowed identifying several problems and drove the ontology engineering process, making explicit ontological commitments that were implicit and elucidating conceptual mistakes.

5. Conclusions

Although several researchers argue in favor of using a foundational ontology as a basis for developing domain ontologies, among them Guarino [19], Guizzardi and colleagues [9] and Fielding and colleagues [8], few works have explored this use. This is the case of the Software Measurement domain, which the proposed ontologies are in general lightweight ontologies, that are not grounded in foundational ontologies. In this paper we presented a Software Measurement Ontology (SMO) developed based on the Unified Foundational Ontology [7, 9]. UFO was chosen because it has been used to evaluate, re-design and integrate (meta) models of conceptual modeling languages, as well as to evaluate, re-design and give real-world semantics to domain ontologies [9].

The proposed ontology is used to establish a common vocabulary used in a strategy to support organizations to obtain and maintain measurement repositories suitable for statistical process control, as well as to perform measurements appropriately in this context. This strategy is composed of three components [4]: the SMO itself, an Instrument for Evaluating the Suitability of a Measurement Repository to Statistical Process Control, and a Body of Recommendations for Software Measurement. The instrument has already been evaluated by experts and used in real cases. The preliminary results point out to its usefulness and also for an agreement of the vocabulary used. The body of recommendations is ready and now it is being evaluated by experts. The feedback given by them will conclude our preliminary evaluation of the vocabulary used and, as a consequence, of the ontology.

As future work, we intend to use the proposed SMO as a conceptual specification for developing and integrating tools supporting the Measurement Process, both in initial and high maturity levels.

References

[1] J. McGarry, D. Card, C. Jones, B. Layman, E. Clark, J. Dean, F. Hall, *Practical Software Measurement: Objective Information for Decision Makers*, Addison-Wesley, Boston USA, 2002.

[2] CMMI Product Team, *CMMI for Development Version 1.2*, Software Engineering Institute, Pittsburgh USA, 2006.

[3] W.A. Florac, A.D. Carleton, *Measuring the Software Process: Statistical Process Control for Software Process Improvement*, Addison-Wesley, Boston USA, 1999.

[4] M.P. Barcellos, A.R.C. Rocha, R.A. Falbo, An Ontology-based Approach for Software Measurement and Suitability Measures Basis Evaluation to Apply Statistical Software Process Control in High Maturity Organizations, Proceedings of the ER2009 PhD Colloquium, Gramado, Brazil, 2009.

[5] G. Guizzardi, On Ontology, ontologies, Conceptualizations, Modeling Languages and (Meta)Models, In O. Vasilecas, J. Edler, A. Caplinskas (Org.). *Frontiers in Artificial Intelligence and Applications, Databases and Information Systems IV*. IOS Press, Amsterdam, 2007.

[6] V. Zamborlini, B. N. Gonçalvez, G. Guizzardi, Codification and Application of a Well-Founded Heart-ECG Ontology, Proceedings of the 3rd Workshop on Ontologies and Metamodels in Software and Data Engineering, Campinas, Brazil, 2008.

[7] G. Guizzardi, *Ontological Foundations for Structural Conceptual Models*, Universal Press, The Netherlands, 2005.

[8] J.M. Fielding, J. Simon, W. Ceusters, B. Smith, Ontological Theory for Ontology Engineering, Proceedings of the Ninth International Conference on the Principles of Knowledge Representation and Reasoning, Whistler, Canada, 2004.

[9] G. Guizzardi, R.A Falbo, R.S.S. Guizzardi, Grounding Software Domain Ontologies in the Unified Foundational Ontology (UFO): The case of the ODE Software Process Ontology. In Proceedings of the XI Iberoamerican Workshop on Requirements Engineering and Software Environments, 2008, 244-251.

[10] M.A.Martin, L. Olsina, Towards an Ontology for Software Metrics and Indicators as the Foundation for a Cataloging Web System, First Latin American Web Congress, 2003.

[11] M.F. Bertoa, A. Vallecillo, F. García, An Ontology for Software Measurement, In: C. Calero, F. Ruiz, M. Piatini (Eds.), *Ontologies for Software Engineering and Software Technology*, Springer-Verlag Berlin Heidelberg, 2006.

[12] ISO/IEC 15939, Systems and Software Engineering – Measurement Process, 2007.

[13] R.A. Falbo, C.S. Menezes, A.R.C. Rocha, A Systematic Approach for Building Ontologies. Proceedings of the 6th Ibero-American Conference on Artificial Intelligence, Lisbon, Portugal, Lecture Notes in Computer Science, vol. 1484, 1998.

[14] R.A. Falbo, Experiences in Using a Method for Building Domain Ontologies. In: Proc. of the 16th International Conference on Software Engineering and Knowledge Engineering, International Workshop on Ontology In Action. Banff, Canada, 2004.

[15] K. Villela, A.R. Rocha, G.H. Travassos et al., The Use of an Enterprise Ontology to Support knowledge Management in Software Development Environments. *Journal of the Brazilian Computer Society*, Brazil, 11(2): 45-59, 2005.

[16] M.P. Barcellos, R.A. Falbo, Using a Foundational Ontology for Reengineering a Software Enterprise Ontology. Proceedings of the Joint International workshop on Metamodels, Ontologies, Semantic Technologies, and Information Systems for the Semantic Web, 2009.

[17] ISO/IEC 9126, Software Engineering – Product Quality, 2001.

[18] ISO/IEC 15504, Information Technology – Process Assessment, 2003.

[19] N. Guarino, Formal Ontology and Information Systems. In Proceedings of International Conference in Formal Ontology and Information Systems, pp 3-15, 1998.