An Ontological Analysis of the ISO/IEC 24744 Metamodel

Fabiano B. RUY[1,2], Ricardo A. FALBO[1], Monalessa P. BARCELLOS[1] and Giancarlo GUIZZARDI[1]

[1] *Ontology and Conceptual Modeling Research Group (NEMO), Computer Science Department, Federal University of Espírito Santo, Vitória, Brazil*
+55-27-4009-2167

[2] *Informatics Department, Federal Institute of Espírito Santo, Serra, Brazil*

Abstract. This paper presents an ontological analysis of the Software Engineering Metamodel for Development Methodologies (SEMDM), provided by the ISO/IEC 24744 Standard. An ISO initiative intends to use SEMDM as source of an ontology for standards harmonization purposes, and the ontological analysis can help to ensure the required semantics. This analysis is done in the light of the Unified Foundational Ontology (UFO). As result, we present some of the problems identified in SEMDM, as well as alternative model fragments solving them.

Keywords. Ontological analysis, Foundational ontology, Standards harmonization, ISO/IEC 24744, Unified Foundational Ontology

Introduction

Software Engineering (SE) is a broad and complex domain. To deal with quality issues, a variety of models and industry-specific standards can be used as references, such as models to improve quality management (e.g. ISO 9001), models for software quality management (e.g. CMMI, and ISO/IEC 12207), models for IT governance (e.g. ITIL, and COBIT), bodies of knowledge (e.g. SWEBOK); among others [1]. Some of these reference models are widely used in industry, often simultaneously. However, the combination of multiple models, developed with different aims and/or by different groups, leads to interoperability problems.

Each reference model defines its own scope, structure of process entities, definitions, terminology, quality systems and approach, amongst other things [2]. These divergences affect not only models of different sources, but also the ones developed by the same group. The International Organization for Standardization (ISO) recognizes this problem, and is now attempting an initiative aiming at harmonizing its own standards [3]. SE standards developed under ISO/IEC JTC1's SC7 have been identified as employing terms whose definitions vary significantly between standards. This led to an ISO request for a study group to investigate the creation of an ontological infrastructure aiming to be a single coherent underpinning for all SC7 standards, present and future [3].

This study group is working since 2012, and proposes a layered framework comprising an ontology network [3]. In the basis of the proposed framework, there is the Definitional Elements Ontology (DEO), providing definitions for particular concepts, and constraints that dictate how they must be related and hence configured in some future definitional standard. DEO is intended to be created from two major sources [3]: ISO/IEC 24744 (SEMDM) [4], a metamodel for SE with a set of definitions of process-focused terms and their interrelationships; and ISO/IEC 24765 (SEVOCAB) [5], a collection of terms from SC7 standards. From DEO, Configured Definitional Ontologies (CDOs) can be defined for specific domains. From a CDO, ontologies specific to particular standards, called Standard Domain Ontologies (SDOs), can be derived. The framework also considers, in a future, to extend DEO by considering ontological distinctions put forward by foundational ontologies (such as sortals and moments [6]). This extension is called AFOS (Advanced Foundational Ontology for Standards) [3].

The SEMDM metamodel, as an essential source for DEO, is one of the basis of the entire framework, providing semantics for all ISO/SC7 standards. Thus, the consistency of this ontological basis is crucial for the success of such initiative. We claim that SEMDM, as well as any other model used as basis for developing the DEO ontology, must be previously analyzed in the light of a foundational ontology. The idea behind ontological analysis is to provide a sound foundation for modeling concepts, if assumed that such concepts are aimed at representing reality [7]. Several efforts have shown the benefits of ontological analysis, such as [8, 9, 10], which includes: (i) the rigorous definition of models, in terms of real-world semantics; (ii) the identification of problems in the definition, interpretation or usage of concepts; and (iii) recommendations for model formality improvements.

In this way, we argue that a mechanism to provide truly ontological foundations to the ISO framework should be used now for defining DEO, and not in the future, as is the approach being currently considered by the ISO study group (considering that AFOS is a future work). In our view, using a foundational ontology for grounding DEO is essential for producing robust formal models with real-world semantics and reduced problems. Moreover, we claim that we do not need a new foundational ontology (such as is the case of AFOS) for doing this work. In contrast, we can use an existing foundational ontology, such as DOLCE [11] or UFO [6], for this purpose.

This paper presents the ontological analysis of the SEMDM metamodel, using the Unified Foundational Ontology (UFO) [6] as our semantic foundation. We identify consistency problems in SEMDM fragments, and point out some suggestions in order to improve these model fragments. We choose UFO because it has been constructed with the primary goal of developing foundations for conceptual modeling. Consequently, UFO addresses many essential aspects for conceptual modeling, which have not received a sufficiently detailed attention in other foundational ontologies. Examples are the notions of material relations and relational properties. For instance, this issue did not receive up to now a treatment in DOLCE, which focuses solely on intrinsic properties (qualities). Moreover, UFO has been employed in many semantic analyses, such as [10, 12, 13].

This paper is organized as follows. The section 1 presents model fragments of both UFO and SEMDM that are relevant to this paper. Section 2 presents the ontological analysis we have performed. Related works are discussed in Section 3. Finally, Section 4 presents our final considerations.

1. UFO Foundations and the SEMDM Metamodel

This section presents parts of the Unified Foundational Ontology (UFO) and ISO/IEC 24744 Metamodel (SEMDM). Only the main model fragments that are used in the ontological analysis are described.

1.1. The Unified Foundation Ontology - UFO

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 of three main parts: UFO-A, an ontology of endurants; UFO-B, an ontology of perdurants (events); and UFO-C, an ontology of social entities (both endurants and perdurants) built on the top of UFO-A and UFO-B. In the sequel, we describe some UFO concepts, only the ones that are important for this paper. This description is based mainly on [12, 14].

Figure 1 shows a fragment of UFO-A. A fundamental distinction in UFO-A is between particulars and universals. ***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. A special type of universal is ***High Order Universal***, whose instances are universals. ***Substantials***[1] are existentially independent particulars. ***Moments,*** in contrast, are particulars that can only exist in other particulars, and thus they are *existentially dependent* on them. Existential dependence can also be used to differentiate intrinsic and relational moments: ***Intrinsic moments*** are dependent on only one single individual (e.g., a color), while ***Relators*** depend on a plurality of individuals (e.g., a marriage). ***Relations*** are entities that link together other entities. ***Formal relations*** hold between two or more entities directly, without any further intervening individual. ***Material relations***, conversely, have material structure of their own, deriving from a ***Relator***, which mediates the related entities. The relations between a relator and the connected entities are said ***mediation relations***.

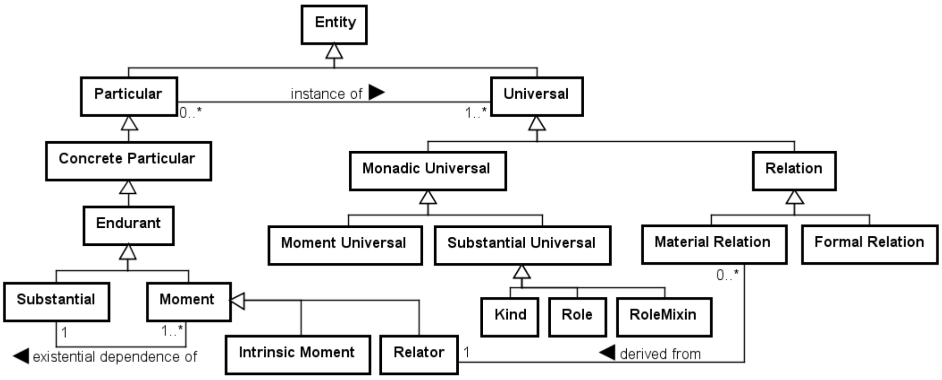

Figure 1. A Fragment of UFO-A – An Ontology of Endurants.

While persisting in time, substantial particulars can instantiate several ***Substantial Universals***. Some of these types, a substantial particular instantiates necessarily (i.e., in every possible situation) and they define what this entity is. These are the types named ***Kind***. There are, however, types that a substantial also instantiates in some

[1] Technically, a substantial does not existentially depend on other substantials which are disjoint from it [6].

circumstances, but not in others, such as is the case of ***Roles***. A ***Role*** is a type instantiated in the context of a given event participation or of a given relation (e.g., student). The abstractions of common properties of roles are represented by ***Role Mixins***. Both ***Kind*** and ***Role*** are ***sortal substantial universals***, but ***Kind*** is a ***rigid sortal***, while ***Role*** is an ***anti-rigid sortal***. ***Role Mixin*** is an anti-rigid ***mixin substantial universal***. Although not represented in Figure 1, ***Sortal Universal***, ***Rigid Sortal***, ***Anti-rigid Sortal*** and ***Mixin Universal*** are concepts of UFO-A. For details see [6].

Figure 2 depicts a fragment of UFO-B. UFO-B makes a distinction between enduring and perduring particulars (*endurants* and *events*). ***Endurants*** are said to be wholly present whenever they are present, i.e., they *are in time*, (e.g., a person). ***Events***, in contrast, are particulars composed of temporal parts, i.e., they *happen in time* in the sense that they extend in time accumulating temporal parts (e.g., a business process). ***Events*** can be atomic or complex. ***Atomic Events*** have no proper parts, while ***Complex Events*** are aggregations of at least two events (that can themselves be atomic or complex). ***Events*** are ontologically dependent entities in the sense that they existentially depend on their participants in order to exist. Moreover, since events happen in time, they are framed by a ***Time Interval***.

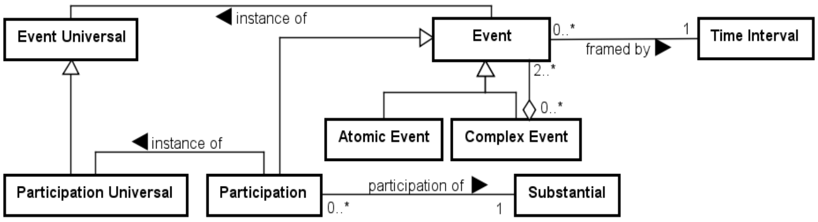

Figure 2. A Fragment of UFO-B – An Ontology of Events.

Figures 3 to 5 show fragments of UFO-C. As shown in Figure 3, one of the main distinctions made in UFO-C is between agents and non-agentive objects. An ***Agent*** is a substantial that creates *actions*, perceives *events* and to which we can ascribe mental states (***Intentional Moments***). ***Agents*** can be physical (e.g., a person) or social (e.g., an organization). A ***Human Agent*** is a type of ***Physical Agent***. An ***Object***, on the other hand, is a substantial unable to perceive events or to have intentional moments. ***Objects*** can also be further categorized into physical (e.g., a book) and social objects (e.g., money). A ***Normative Description*** is a type of ***Social Object*** that defines one or more rules/norms recognized by at least one social agent and that can define nominal universals such as social objects and social roles. A ***Plan Description*** is a special type of normative description that describes ***Complex Action Universals*** (plans).

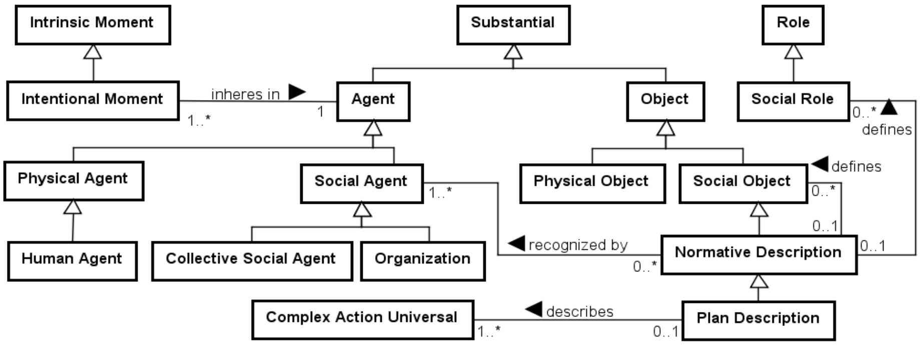

Figure 3. A Fragment of UFO-C: Distinction between Agents and Objects.

Intentional Moments are intrinsic moments. They can be ***Social*** or ***Mental Moments***. A ***Social Commitment*** is type of ***Social Moment*** establishing a commitment of an agent towards another. A special type of ***Commitment*** is an ***Appointment***, which is a commitment whose goal explicitly refers to a time interval (e.g., a scheduled task). Like commitments, appointments can be either ***Internal*** or ***Social Appointments***.

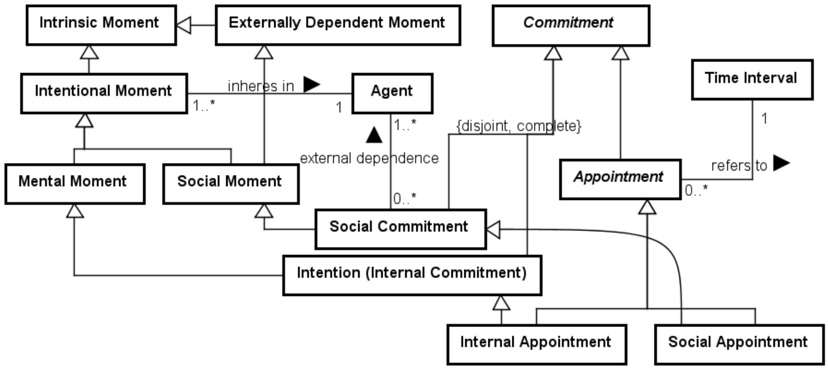

Figure 4. A Fragment of UFO-C: Commitments and Appointments.

Finally, ***Actions*** are intentional events, i.e., they have the specific purpose of satisfying some intention. As ***Events***, actions can be atomic or complex. A ***Complex Action*** is composed of two or more participations. These participations can themselves be intentional (i.e., be themselves actions) or unintentional events. Only agents can perform actions. An object participating in an action does not have intention. ***Object Participations*** can be of the following types: ***Creation***, ***Change***, ***Usage*** or ***Termination***.

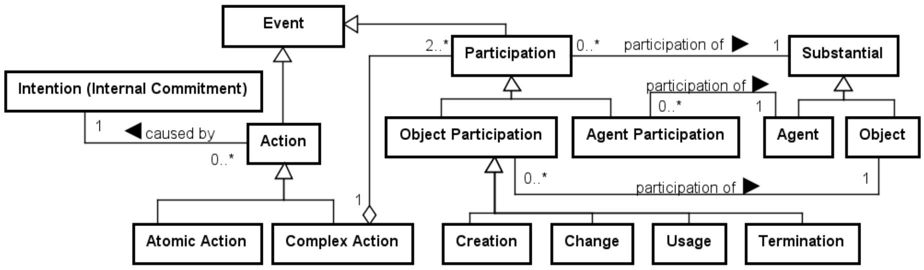

Figure 5. A Fragment of UFO-C: Actions and Participations.

1.2. The Software Engineering Metamodel for Development Methodologies - SEMDM

SEMDM is a metamodel establishing a framework for the definition and extension of development methodologies in information-based domains (IBD) [4]. It is conceived as a model of both the *methodology* and the *endeavour* domains. A methodology specifies the process to be executed, usually as a set of related activities, tasks and/or techniques, together with the work products that must be manipulated at each moment and by whom, possibly including models, documents and other inputs and outputs. An endeavour is an IBD development effort aimed at the delivery of some product or service through the application of a methodology. Modeling the methodology and endeavour domains at the same time gives rise to pairs of classes in the metamodel that represent the same concept at different levels of classification. This pattern of two classes in which one of them represents "kinds of" the other is called a *powertype pattern* in [4], and it is clearly related to the notion of *high order universal* in UFO.

Figure 6 shows the *endeavour level* of SEMDM. Since most of the SEMDM classes arose from the *powertype pattern*, the *methodology level* is structurally very similar, differing in the Person class (exists only in the *endeavour*) and the attributes and some cardinalities. The complete models of SEMDM are presented in [4]. The *methodology* and *endeavour levels* are divided into five mains class groups, namely: *Work Units*, which deals with jobs performed, or intended to be performed; *Stages*, regarding managed time frames; *Producers*, concerning agents with responsibility to execute work units; *Work Products*, which refers to artifacts of interest; and *Model Units*, dealing with components of models (not addressed in this paper).

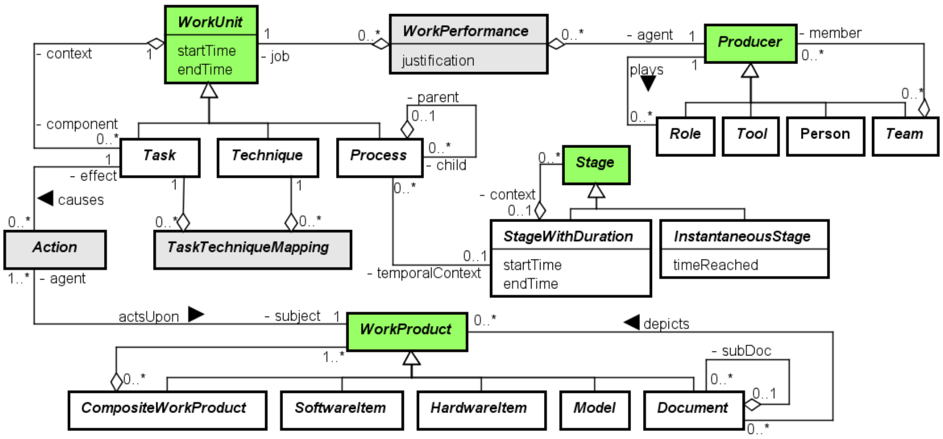

Figure 6. SEMDM Endeavour level.

2. SEMDM Ontological Analysis

This section presents the ontological analysis of SEMDM, using UFO as basis. It is worth pointing out that the analysis focus is on foundational aspects, searching for model inconsistencies that could be solved with a foundational ground. The SEMDM model, as part of an International Standard, has a large acceptance and solid knowledge background, which are not in question here. For performing this analysis, we have selected some SEMDM fragments. As the concepts of the fragments are analyzed, the foundations are discussed for the identified problems, and some suggestions are made by text or by new model fragments. Concepts of ISO are written in **bold**, concepts of UFO in ***bold italics*** (and shown detached in the models), and concepts introduced by the ontological analysis are written underlined. The following three subsections present the ontological analysis of the SEMDM group classes of Process, Product and Producer.

2.1. Process Classes

The main process classes in SEMDM are **WorkUnit** and **WorkUnitKind**. According to SEMDM, "a work unit is a job performed, or intended to be performed, within an endeavour". A work unit kind, in turn, is a specific kind of work unit that is to be instantiated in several endeavours.

In terms of UFO, a **WorkUnitKind** can be viewed as an ***Action Universal***. **WorkUnit**, in the other hand, is an overloaded concept. Since a work unit is defined as "a job performed, or intended to be performed, within an endeavour", the class **WorkUnit** collapses two concepts in UFO: ***Action***, which can be used to represent a job performed, and ***Internal Appointment***, which can be used to represent a job intended to be performed within an endeavour in a certain period of time. **WorkUnit** has temporal properties: **startTime** is the point in time at which the work unit is started; **endTime** is the point in time at which the work unit is finished. Since it is not clear if a work unit is an action or an appointment, it is not possible to say if **startTime** refers to the expected start time of the appointment or to the actual start time of the event. The same applies to **endTime**. Thus, for disambiguating the notion of **WorkUnit**, we suggest introducing the concepts of Scheduled Work Unit and Performed Work Unit, as Figure 7 shows.

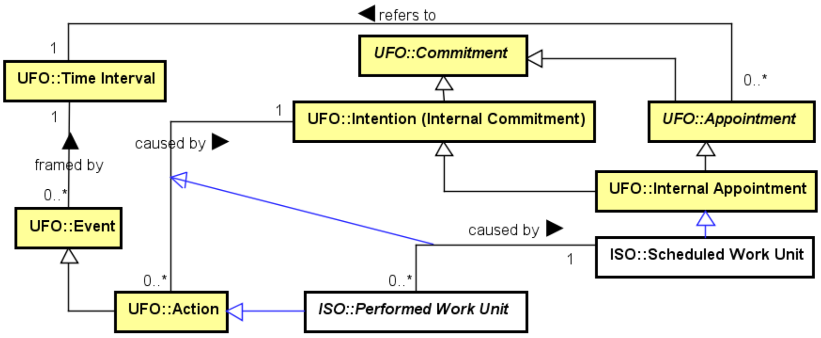

Figure 7. Scheduled Work Unit x Performed Work Unit.

This problem also manifests in the case of **Stage**, which is defined as "a managed time frame within an endeavour". In this case, it is not clear if the time frame of a stage refer to an expected or the actual time frame. Thus, analogously to **WorkUnit**, we suggest to introduce the concepts of Scheduled Stage and Performed Stage. In this paper, due to space limitation, we advance our analysis considering only the notion of Performed Work Unit, i.e., **WorkUnit** as an intentional event (***Action***).

In SEMDM, **WorkUnit** is specialized into **Process**, **Task** and **Technique**. A process is a large-grained work unit that operates within a given area of expertise; a task is a small-grained work unit that focuses on what must be done in order to achieve a given purpose; and a technique is a small-grained work unit that focuses on how the given purpose may be achieved. However, ontologically analyzing, the distinction between these subtypes of **WorkUnit** is not clear. What does it mean to be large or small-grained? In SEMDM, work units, in general, can be decomposed in tasks. Processes, in turn, can further be decomposed in sub-processes. We know that in Software Engineering this fuzzy distinction is frequently applied. However, we advocate against this indeterminacy, which can be resolved by considering a mereological distinction between simple and composite work units. Thus, we suggest introducing the notions of Composite Performed Work Unit, Simple Performed Work Unit and Composite Task, as Figure 8 shows. A Composite Performed Work Unit is composed of at least two disjoint Performed Work Units, since it does not make sense to say that a composite work unit is composed of zero or even only one work unit (a direct consequence of considering here the weak supplementation axiom in mereology). A Simple Performed Work Unit, or just Simple Task, in turn, is a work unit that is not composed of other work units. Regarding composite work units, we distinguish between **Process** and Composite Task. Process is a Composite Performed

Work Unit that is not part of any other composite work unit. Composite Task, in turn, is a Composite Performed Work Unit that is part of another composite work unit.

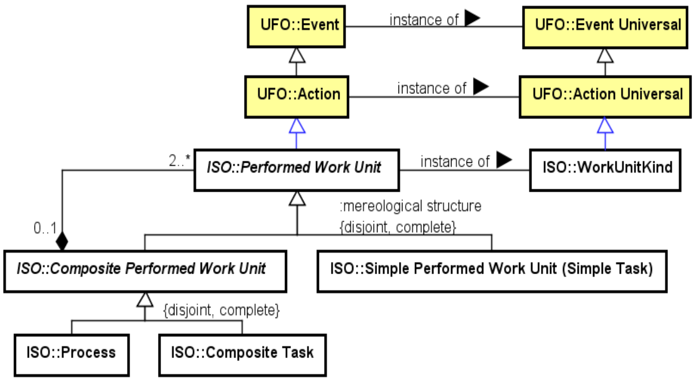

Figure 8. Simple and Composite Work Units.

It is worth to point out that **Technique** is not included in Figure 8 because our analysis shows that it is not a subtype of **WorkUnit**. In SEMDM, tasks and techniques are considered as small-grained work units. The distinction between task and technique lies in the fact that the former focuses on *what* must be done in order to achieve a given purpose, while the latter focuses on *how* the given purpose may be achieved. Following this definition, it becomes clear that a technique is not a work unit. As described in ISO/IEC 24744, a technique can be used to accomplish a given task. According to UFO, **Technique** is a ***Normative Description***, more specifically a ***Plan Description***, i.e. a normative description that defines a plan (***Complex Action Universal***).

Take the example of a technique given in the standard: CRC Cards. Suppose that an organization decides to follow the following workflow for this technique: (T1) Choose a coherent set of use cases; (T2) Walk through the scenario, naming cards and responsibilities; (T3) Vary the situations to test the cards; (T4) Add cards and push cards to the side to let the design evolve; (T5) Write down the key responsibility decisions and interactions. This plan describes a Composite Task Kind, which is composed of five **WorkUnitKinds** (in this case, Simple Task Kinds). When applying this technique in a particular endeavour, work units of these types are instantiated. Thus, it does not make sense to talk about **TechniqueKind**, but only about **Technique**, and thus we suggest eliminating the **TechniqueKind** class.

To capture the fact that techniques applies to **TaskKinds**, SEMDM defines the **TaskTechniqueMappingKind** class. However, once a technique describes a plan for a given composite task kind, this technique applies (is recommended) to it. Thus, we suggest eliminating also the **TaskTechniqueMappingKind** class.

To capture the fact that a technique is being actually used to accomplish a given task, SEMDM defines the **TaskTechniqueMapping** class. An instance of this class represents the fact that, in an endeavour, a given technique is being used to accomplish a given task. In terms of UFO, **TaskTechniqueMapping** is an ***Object Participation***, more specifically, a ***Usage*** participation, since the technique itself is not changed during this use. Figure 9 shows the model fragment modeling the notion of **Technique** as a ***Plan Description***. In this figure, we rename the **TaskTechniqueMapping** class to Work Unit Technique Mapping to align it to the proposed model.

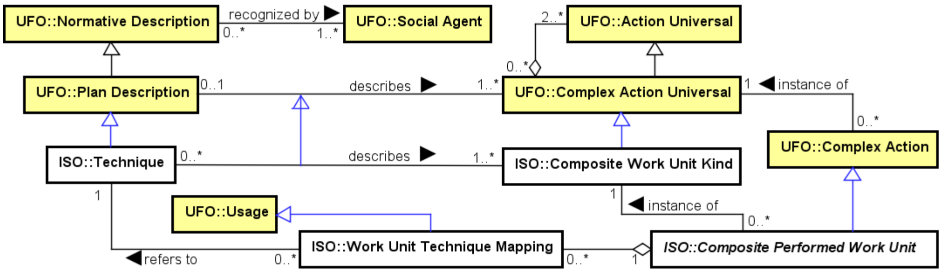

Figure 9. Technique as a Plan Description.

It is important to notice that there is another concept in SEMDM that is aligned with the notion of ***Normative Description*** in UFO, namely, **Guideline**. Guideline is an indication of *how* some methodology elements can be used. However, in contrast with **Technique**, **Guideline** does not describe a plan. Thus, to prevent the confusion of a technique with a guideline, we require that a technique must always describe a plan (Composite Work Unit Kind).

2.2. Product Classes

The main product classes in SEMDM are **WorkProduct** and **WorkProductKind**. According to SEMDM, "a work product is an artefact of interest for the endeavour". A work product kind, in turn, is "a specific kind of work product, characterized by the nature of its contents and the intention behind its usage". **WorkProduct** has five subtypes: **SoftwareItem**, **HardwareItem** (respectively a piece of software or hardware that is of interest to the endeavour), **Model** (an abstract representation of some subject that acts as the subject's surrogate for some well defined purpose), **Document** (a durable depiction of a fragment of reality) and **CompositeWorkProduct** (an aggregate of other elements). Counterpart subtypes exist for **WorkProductKind**.

In terms of UFO, a **WorkProductKind** can be viewed as an ***Object Universal***, while a **WorkProduct** is an ***Object***. Concerning the subtypes of **WorkProduct**, this class hierarchy mixes up two different specialization criteria: one effectively dealing with the nature of different types of work products (encompassing the first four subtypes), and another regarding the mereological structure of work products, addressing the fact that work products can be composed of other work products. Thus, we argue that two generalization sets have to be considered, as Figure 10 shows.

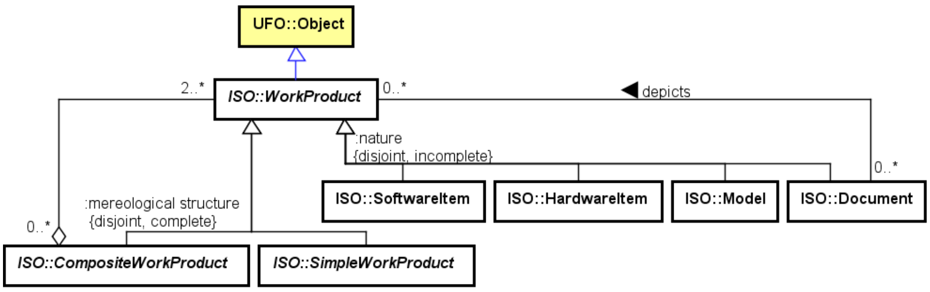

Figure 10. Work Product Types.

The first one deals with the mereological structure of work products, and introduces the concept of Simple Work Product, as a work product that is not composed of other work products. A **CompositeWorkProduct**, in turn, shall be composed of at least two other work products (weak supplementation). This class hierarchy is complete and disjoint. The second one deals with the nature of work products, and is disjoint and incomplete. We consider that this class hierarchy should be considered incomplete, because it is possible to envision other types of work products not yet covered by the current subtypes of **WorkProduct**, such as software systems delivered to clients, or software services provided to customers. Note that **WorkProduct**, **CompositeWorkProduct** and Simple Work Product are shown as abstract classes. This is because these concepts are non-sortals that classifies entities carrying different principles of identity, and thus that cannot be directly instantiated [6]. Therefore, instances of **WorkProduct** (and consequently of these two abstract specializations) should necessarily be instances of (exactly one) of the subtypes of **WorkProduct** in the nature generalization set.

According to SEMDM, a document may depict a number of work products, as well as a document may be composed of other documents. In our view, the different types of work products can be mereological complex, i.e., the distinction between simple and composite is orthogonal to the different subtypes of **WorkProduct**. For instance, documents can be composed of other documents; software items can be composed of other software items (such as a program composed of several functions); hardware items can also be composed of other hardware items. In this fragment of the ontology, it is important to axiomatize the parthood relationship between work products constraining which types of work products can be part of other types of work products. For the sake of space limitations, these axioms are not developed in this paper.

Finally, the link between the process and the product fragments of SEMDM is achieved by means of **Action** and **ActionKind** in the following manner: "An action is a usage event performed by a task upon a work product". Actions represent the fact that specific tasks use specific work products. An action kind is defined as “a specific kind of action, characterized by a given cause (a task kind), a given subject (a work product kind) and a particular type of usage". Action kinds describe how tasks of specific kinds use work products of specific kinds, including the nature of such usage (**ActionKind.type** = {create | modify | readOnly | delete})

In terms of UFO, an **ActionKind** is an ***ObjectParticipationUniversal*** linking a **WorkProductKind** to the **WorkUnitKinds** in which they can be handled. **ActionKind.type** captures the same object participation types admitted in UFO. **Action**, in turn, is an ***Object Participation***, which can be a ***Creation***, ***Change***, ***Usage*** or ***Termination***, depending on how the **WorkProduct** participates in the Performed Work Unit. We recommend changing the name of the relationship between Performed Work Unit and **Action**. In SEMDM, tasks are said to "cause" actions. In UFO, this is not a causality relationship, but a whole-part relation between ***Complex Action*** and ***Participation***, as Figure 11 shows.

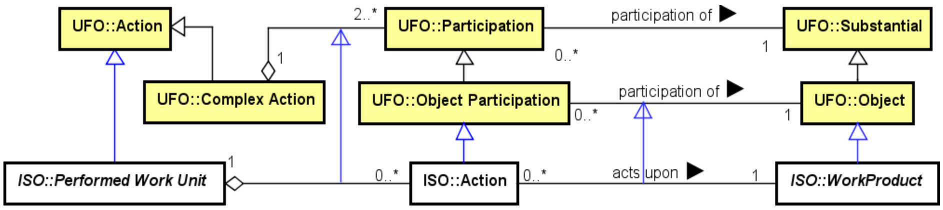

Figure 11. SEMDM Actions as Object Participations.

2.3. Producer Classes

The main producer classes in SEMDM are **Producer** and **ProducerKind**. According to SEMDM, "a producer is an agent that has the responsibility to execute work units". A producer kind, in turn, "is a specific kind of producer, characterized by its area of expertise". **Producer** is specialized into **Role** (a collection of responsibilities that a producer can take), **Tool** (an instrument that helps another producer to execute its responsibilities in an automated way), and **Team** (an organized set of producers that collectively focus on common work units). Counterpart subtypes exist for **ProducerKind**. **Producer** has an additional subtype, **Person**, which allows taking into account individual persons at the endeavour level. **Producer** is also related to **WorkUnit** through **WorkPerformance**, so links between units of work and the assigned and/or responsible producers are possible.

In the light of UFO, **Team** and **Person** are ***Agents***. **Team** is a ***Collective Social Agent***, while **Person** is a ***Human Agent***. In the other hand, **Tool** is an ***Object***. These subtypes of **Producers** have a very different nature. This makes us question: are all of them actually producers? As discussed in Section 1, one of the primary distinctions in UFO-C is the one between ***Agents*** and ***Objects***. ***Agents*** act motivated by intentions and can assume responsibilities. ***Objects***, in turn, do not act. Taking this distinction into account, we claim that only **Team** and **Person** can be considered **Producers**. Moreover, **Producer** should be considered a ***RoleMixin*** (an anti-rigid and externally dependent non-sortal [6]) that is dependent on at least one **WorkPerformance**, indicating that the **Producer** participates in the performance of a <u>Performed Work Unit</u>. Note also that **Producer**, as a ***RoleMixin***, is an anti-rigid class, while **Person** and **Team** are rigid classes. Someone becomes a producer when participates in a **WorkPerformance**. In the other hand, a person is always instance of **Person**. Since a rigid class cannot be subtype of an anti-rigid class [6], **Person** and **Team** cannot be subtypes of **Producer**. Thus, for correctly modeling the *rolemixin pattern* [6], we need to introduce the roles <u>Person Producer</u> and <u>Team Producer</u> as subtypes of **Producer**. In summary, for a **Person/Team** to play the role of <u>Person Producer</u> / <u>Team Producer</u> (subtypes of **Producer**), she/it must be participating in at least one **WorkPerformance**. Figure 12 shows the resulting model fragment concerning **Producer**.

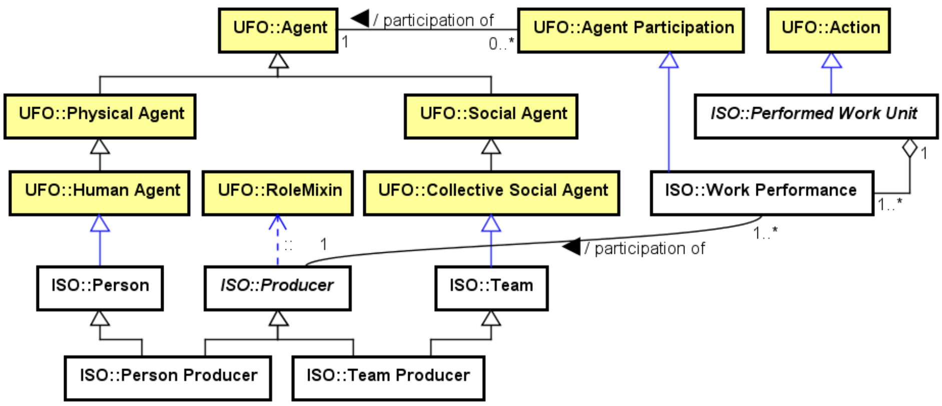

Figure 12. Producers.

Still regarding **Team**, in SEMDM, a team is composed of zero or more **Producers**. Since Team is a ***Collective Social Agent***, it must be composed of at least two members.

And these members should be ***Agents***. Thus, it sounds strange to say that a **Role** or a **Tool** is part of a **Team**, since they are not agents. Only **Persons** and other **Teams** can compose a **Team**. Moreover, teams are defined independently of their participation in **WorkPerformances**, and thus it does not make sense to say that a **Team** is composed of **Producers**. In fact, **Persons** are allocated to **Teams**, giving rise to the concept of Person-Team Allocation, and that **Teams** are allocated to other **Teams**, giving rise to the concept of Team-Team Allocation, as Figure 13 shows. These two relators are connecting entities, linking a **Team** to its members (**Persons** and other **Teams**). For ensuring that a team is composed of at least two members, an axiom is made necessary. Finally, this model fragment can be enriched by pointing which **RoleKind** a **Person/Team** plays in a Person-Team Allocation/Team-Team Allocation.

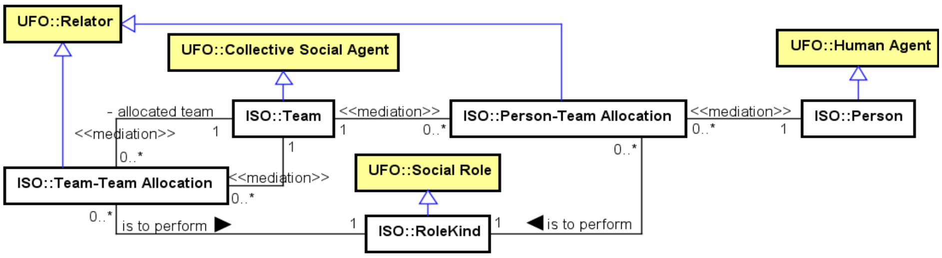

Figure 13. Team Composition.

Concerning **Role**/**RoleKind**, for ontologically analyzing these notions, we should answer first a question: what is the difference between **Role** and **RoleKind** in SEMDM? Let us examine the definitions and examples given in [4] for these classes.

RoleKind: "A role kind is *a specific kind of role, characterized by the involved responsibilities*". Ex.: "In a given methodology, it is necessary that close contact is maintained with the customers. To capture this independently of any person or group in particular, the method engineer introduces a role kind “Customer Liaison”". Note that responsibility is a key element in **RoleKind** definition. This is so important that **RoleKind** has an attribute (**responsibilities**) for describing it. In terms of UFO, **RoleKind** is a ***Social Role*** that is defined by a ***Normative Description*** recognized by the organization, and that establishes the responsibilities associated to a number of **RoleKinds**.

Role: "A role is a collection of responsibilities that a producer can take". Ex.: "During a certain project, Mary is in charge of writing the user documentation. Mary leaves the project midway and John takes over with the same responsibilities. This collection of responsibilities, which could be called “technical writer”, is a role". How can a role be a collection of responsibilities, since this is the definition of **RoleKind**? “Technical writer”, as well as “Customer Liaison”, is a social role. Mary and John in the example are persons that instantiate the social role “Technical writer”.

Finally, regarding **Tool**, according to SEMDM, "a tool is an instrument that helps another producer to execute its responsibilities in an automated way". A tool may assist a set of producers. As previously discussed, in our view, **Tool** is not a subtype of **Producer**. In UFO, **Tool** is an ***Object***, and as such can be used as a resource in the performance of a work unit. Figure 14 shows the model fragment addressing this view.

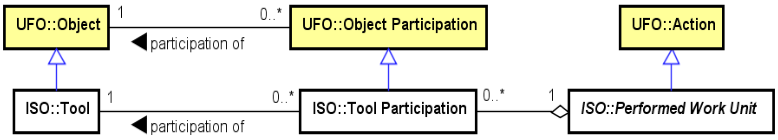

Figure 14. Tools.

3. Related Work

Several studies have attempted to give ontological foundations to conceptual models, mainly the ones used as basis for integration or for construction of new models. A significant work in this context is [9], where Smith ontologically analyses the ISO 15926 (Lifecycle Integration of Process Plant Data Including Oil and Gas Production Facilities) and points out that it is marked by a series of defects and, unlike is proposed, it is not an ontology. Smith presents his ontological principles and problems of considering a data model as an ontology. Analogous to our work, the analysis is done in an International Standard to be used for interoperability purposes. Thus, our work (including the general conclusions achieved) is quite similar to [9]. In contrast, we took UFO as basis for our analysis.

Some studies apply foundational ontologies to conduct ontological analysis, such as [15], which presents an ontological analysis of four interoperability standards, and [7], which evaluates reference models. Both works use as basis for the ontological analysis the Bunge-Wand-Weber (BWW) foundational ontology. In [16], Hejá et al. ontologically analyzed SNOMED CT, a comprehensive medical terminology, based on DOLCE. As discussed in the introduction of this paper, our choice for UFO to conduct this ontological analysis is primary because it was constructed with the main goal of developing foundations for conceptual modeling. Several works use UFO as ground for ontological analysis. UFO has been used to evaluate, re-design and give real-world semantics to languages, models and domain ontologies. A much related work is [12], which ontologically analyses the Software Process Ontology (SPO) in the light of UFO, reengineering it. Due to the domain proximity, several analogous problems were identified and corrected.

4. Conclusions

There are many initiatives to harmonize standards [1, 2, 3, 17], such as the one planned by the ISO/SC7 Study Group. These initiatives often use ontologies to treat the problem of semantic interoperability. Thus, we need high quality ontologies, serving as basis for integration. International Standards are consolidated sources of knowledge that reflect a shared conceptualization. However, most of them were not designed to be an ontology, presenting problems from a foundational point of view. Therefore, we advocate that ontological analysis, grounded by foundational ontologies, is the key to conceive quality ontologies from the models/metamodels underlying standards. Ontological analysis is an important tool to promote improvements in models and ontologies. By ontologically analyzing them, we include a foundational layer beneath the concepts and relations, providing the needed ground and consistency to turn them into high quality ontologies.

In this paper, we present an ontological analysis of the ISO/IEC 24744 Metamodel. We have identified problems and pointed out some solutions. Although in some cases we conduct a deep analysis, we do not analyze the entire SEMDM. Thus, besides the presented suggestions, we could point out two general recommendations in order to turn SEMDM into a quality ontology as needed by the ISO harmonization framework: (i) the nature of all concepts must be analyzed from a foundational perspective before being introduced in the resulting ontology; and (ii) constraints must be specified (e.g., defining properly axioms in OCL or FOL), guaranteeing consistency.

Acknowledge

This research is funded by the Brazilian Research Funding Agency CNPq (Process Number 485368/2013-7).

References

[1] C. Pardo, F.J. Pino, F. Garcia, M.T. Baldassarre, M. Piattini, From chaos to the systematic harmonization of multiple reference models: A harmonization framework applied in two case studies. *The Journal of Systems and Software* **86** (2013), 125-143.

[2] C. Pardo, J.P. Pino, F. Garcia, M. Piattini, M.T. Baldassarre, An ontology for the harmonization of multiple standards and models. *Computer Standards & Interfaces* **34** (2012), 48–59.

[3] B. Henderson-Sellers, C. Gonzalez-Perez, T. McBride, G. Low, An ontology for ISO software engineering standards: 1) Creating the infrastructure. *Comp. Standards & Interf.* **36**, 3 (2014), 563-576.

[4] ISO/IEC, ISO/IEC 24744. Software Engineering – Metamodel for Development Methodologies. ISO, Geneva (2007).

[5] ISO/IEC/IEEE, ISO/IEC/IEEE 24765. Systems and Software Engineering - Vocabulary. ISO, Geneva (2010).

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

[7] P. Fettke and P. Loos, Ontological Evaluation of Reference Models Using the Bunge-Wand-Weber Model. *Proceedings of the 2003 Americas Conference on Information Systems*, Florida, USA (2003).

[8] G. Shanks, E. Tansley and R. Weber, Using Ontology to Validate Conceptual Models. *Communications of the ACM* **46** (2003), 85-89.

[9] B. Smith, Against Idiosyncrasy in Ontology Development. *Formal Ontology and Information Systems* – FOIS´2006, Baltimore (2006).

[10] J.P.A. Almeida, E.C.S. Cardoso, G. Guizzardi, On the Goal Domain in the RM-ODP Enterprise Language: An Initial Appraisal based on a Foundational Ontology. 6th Workshop on ODP for Enterprise Computing – WODPEC'2010, Proc. of the 14th IEEE Int. Enterprise Distributed Object Computing Conference Workshops – EDOCW'2010, *IEEE Computer Society Press* (2010), 382-390.

[11] A. Gangemi, et al., Sweetening ontologies with DOLCE. *Knowledge engineering and knowledge management: Ontologies and the semantic Web* (2002) 166-181.

[12] A.C. Bringuente, R.A. Falbo, G. Guizzardi, Using a Foundational Ontology for Reengineering a Software Process Ontology. *Journal of Information and Data Management* **2** (2011), 511-526.

[13] E. Eessaar, R. Sgirka, An Ontological Analysis of Metamodeling Languages. *Information Systems Development*, Springer New York (2013), 381-392.

[14] G. Guizzardi, R.A. Falbo and 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*, Recife (2008).

[15] P. Green, M. Rosemann, M. Indulska, C. Manning, Candidate interoperability standards: An ontological overlap analysis. *Data & Knowledge Engineering* **62**, Issue 2 (2007), 274-291.

[16] G. Héja, G. Surján, P. Varga, Ontological analysis of SNOMED CT. *BMC Medical Informatics and Decision Making* **8** (2008).

[17] B. Henderson-Sellers, T. McBride, G. Low, C. Gonzalez-Perez, Ontologies for International Standards for Software Engineering. ER, *Lecture Notes in Computer Science* **8217**. Springer (2013), 479-486.