The Visual Semantic Web: Unifying Human and Machine Semantic Web Representations with Object-Process Methodology

Dov Dori
Technion, Israel Institute of Technology, Haifa 32000, Israel
dori@ie.technion.ac.il, and
Massachusetts Institute of Technology, Cambridge, MA 02139, USA
dori@mit.edu

Abstract

The Visual Semantic Web (ViSWeb) paradigm enhances human accessibility to the current Semantic Web technology by enabling the visualization of knowledge. Arguing against the claim that humans and machines need to look at different knowledge representation formats, Object-Process Methodology (OPM) is shown to enable modeling of systems in a single graphic and textual model. ViSWeb provides for representation of knowledge over the Web in a unified way that caters to humans as well as machines. ViSWeb is developed as an OPM-based layer on top of XML/RDF/OWL to express knowledge visually and in natural language. Both the graphic and the textual representations are strictly equivalent. Being intuitive yet formal, they are not only understandable to humans, but are also amenable to computer processing. The advantages of the ViSWeb approach include equivalent graphic-text knowledge representation, visual navigability, semantic sentence interpretation, specification of system dynamics, and complexity management. The ability to use such bimodal knowledge representation that is both human understandable and machine processable is a major step forward in the evolution of the Semantic Web.

1. The Human-Machine Language Orientation Dilemma

The development of the Semantic Web is driven by the assumption that humans and machines must each use a different format of knowledge representation. For example, the RDF [7] introduction reads: "The World Wide Web was originally built for human consumption, and although everything on it is *machine-readable*, this data is not *machine-understandable*" (emphasis in source). Berners-Lee, Hendler and Lassila [5] have noted that "*the Semantic Web is not a separate Web but an extension of the current one, in which information is given well-defined meaning, better enabling computers and people to work in cooperation.*" Constructing a comprehensive Web-based knowledge management system must reconcile this human-machine language orientation dilemma. The bulk of knowledge that continues being gathered on the Web in an ever accelerating rate is expressed in free natural language, which is currently indigestible to machines (e.g. [11]). Still, current technologies for Web-based knowledge management are developed based on the premise that while humans prefer natural language, machines must use XML-like scripts, which humans have to invest great efforts to decipher: "*... instead of asking machines to understand people's language, the new technology, like the old, involves* ***asking people to make some extra effort****, in repayment* ***for*** *which they will get substantial* ***new functionality****.*" [4]. The ViSWeb approach provides this functionality without requiring that effort. This is true not only for RDF [21, 38], but also for OWL, the Web Ontology Language [32], which reads: "*In order to map this (the*

World Wide Web] terrain more precisely, computational agents require machine-readable descriptions of the content and capabilities of web accessible resources. These descriptions ***must be in addition*** *to the human-readable versions of that information.*"

In contrast, the Visual Semantic Web (ViSWeb) approach is founded on the premise that human and machine Web-based knowledge need not necessarily be represented by two distinct formats. ViSWeb is based on Object-Process Methodology (OPM) [14]. Using a bimodal representation of graphics and text, OPM models knowledge about systems of various types and different complexity levels in a single model, which integrates structure and behavior. OPM, described in more detail below, combines a subset of natural language, called Object-Process Language (OPL), with a formal, yet intuitive graphic model, a set of one or more Object-Process Diagrams (OPDs) of exactly the same knowledge expressed in OPL. This dual graphic-textual representation constitutes a solid foundation for generic knowledge representation over the Web.

2. Combining Graphic and Textual Knowledge Representations

A powerful knowledge modeling and communication modality, which is complementary to language, is graphics. Diagrams are often invaluable for describing models of abstract things, especially complex systems. The fact that people from the early caveman days to date have been using some kind of sketching or diagramming technique to express their knowledge or ideas is a testimony to the viability of the graphic representation. However, such representation of our knowledge is valuable only if it is backed by a comprehensive and consistent modeling methodology. Such methodology is essential if we want to represent knowledge, understand complex systems in any domain, and communicate our understanding to others. An accepted diagramming method has the potential of becoming a powerful modeling tool if it constitutes an unambiguous language. In such visual formalism, each symbol must bear defined semantics and the links among the symbols must unambiguously convey some meaningful information that is clearly understood by the diagram readers.

Knowledge Representation Approaches

A number of knowledge representation approaches have been designed with the goal of graphically and/or textually representing knowledge aimed at facilitating human understanding and communication of knowledge. These approaches include concept maps [2, 18, 19, 26, 27], semantic networks [22, 6, 16, 25], XML Topic Map (XTM) [31] conceptual graphs (CGs) [33, 34, 9, 13, 23], Knowledge Interchange Format (KIF) [20], Cyc [11], the Common Logic (CL) standard initiative [35], Unified Modeling Language (UML) [21, 16, 30], and Object-Process Methodology (OPM), which is the basis for the Visual Semantic Web presented in the paper.

Object-Process Methodology

Most interesting and challenging systems are those in which structure and behavior are highly intertwined and hard to separate. Motivated by this observation, Object-Process Methodology (OPM) [14] is a holistic approach to the study and development of systems, which integrates the object-oriented and process-oriented paradigms into a single frame of reference. Structure and behavior, the two major aspects that each system exhibits, co-exist in the same OPM model without highlighting one at the expense of suppressing the other. Due to its structure-behavior integration, OPM provides a solid basis for modeling complex systems in general and those documented through the Semantic Web in particular. The elements of the OPM ontology are entities and links. Entities, the basic building blocks of any system modeled in OPM, are of three types: *objects* with *states*, and *processes*. *Objects* are (physical or informatical) things that exist, while *processes* are things that transform objects. *Links* can be structural or procedural. *Structural links* express static, time-independent relations between pairs of entities. The four fundamental structural relations are Aggregation-participation, generalization-specialization, exhibition-characterization, and classification-instantiation. *Procedural links* connect entities (objects, processes, and states) to describe the behavior of a system.

Behavior is manifested in three major ways: (1) processes can *transform* (generate, consume, or change the state of) objects; (2) objects can *enable* processes without being transformed by them; and (3) objects can *trigger* events that invoke processes if some conditions are met. Accordingly, a procedural link can be a transformation link, an enabling link, or an event link. A *transformation link* expresses object transformation, i.e., object consumption, generation, or state change. An *enabling link* expresses the need for a (possibly state-specified) object to be present, in order for the enabled process to occur. The enabled process does not transform the enabling object. An *event link* connects a triggering entity (object, process, or state) with a process that it invokes. The event types that OPM supports include state entrance, state change, state timeout, process termination, process timeout, reaction timeout, and external events. External events include clock events and triggering by environmental entities such as a user or an external device.

Two semantically equivalent modalities, one graphic and the other textual, jointly express the same OPM model. A set of inter-related Object-Process Diagrams (OPDs), showing portions of the system at various levels of detail, constitute the graphical, visual OPM formalism. Each OPM element is denoted in an OPD by a symbol, and the OPD syntax specifies correct and consistent ways by which entities can be connected via structural and procedural links, each having its specific, unambiguous semantics. OPM assigns special graphical symbols for a selected set of relations (similar to UML class diagrams, only for a larger set of relations). OPCAT (Object-Process CASE Tool) [5] is a Java-based software environment that supports OPM system modeling and evolution. The Object-Process Language (OPL), defined by a context-free grammar, is the textual counterpart modality of the graphical OPD set. OPL is a dual-purpose language, oriented towards humans as well as machines. Catering to human needs, OPL is designed as a constrained subset of English, which serves domain experts and system architects, jointly engaged in analyzing and designing a system, such as an electronic commerce system or a Web-based enterprise resource planning system. Every OPD construct is expressed by a semantically equivalent OPL sentence or phrase. This dual representation of OPM increases the processing capability of humans according to the cognitive theory of multimodal learning proposed by Mayer [8, 1]. The knowledge that OPM can represent is not restricted to just structural, as in CGs and most other knowledge representation formats. It can also be procedural, showing temporal order and enabling cause and effect analysis. Designed also for machine interpretation through a well-defined set of production rules, OPL provides a solid basis for automating the generation of the designed application. Indeed, OPCAT currently generates complete Java code from OPL script and enables the generation of any other formal language.

Unlike the graphic representation in the Semantic Web, which is an auxiliary means to illustrate the machine-oriented, XML-based content, OPDs constitute a complete and consistent visual formalism that goes hand in hand with the OPL. A basic OPM principle is the text-graphic equivalence principle: *Anything that is expressed graphically by an OPD is also expressed textually in the corresponding OPL paragraph, and vice versa.* Following this principle, our goal in developing ViSWeb, the Visual Semantic Web, as in OPM in general, is to specify a system by a set of inter-related Object-Process Diagrams and their completely equivalent corresponding OPL paragraphs. This equivalence implies that both modalities, the graphic and the textual, contain exactly the same information, albeit in two different ways of expression. Due to this complete equivalence, each can be reconstructed from the other. As noted, in spite of the apparent graphics-text redundancy, from a human factors engineering viewpoint, these two modalities activate different cognitive processes and therefore reinforce the understanding of each other and of the system as a whole.

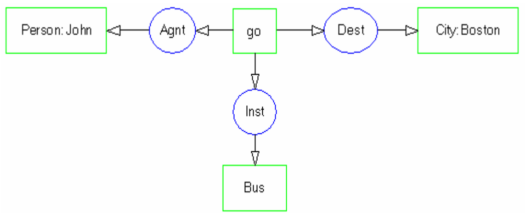
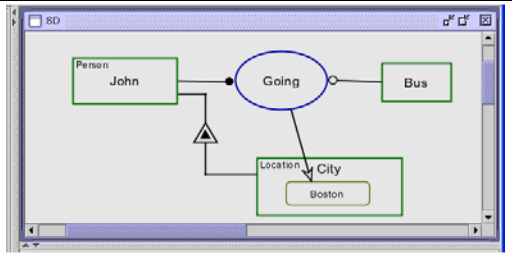
A major problem with most graphic modeling approaches is their scalability: As the system complexity increases, the graphic model becomes loaded with shapes and cluttered with links that cross each other in all directions. The limited channel capacity [24] is addressed by OPM and implemented in OPCAT with three abstraction/refinement mechanisms. These enable complexity management by providing for the creation of interrelated OPDs (along with their corresponding OPL paragraphs) that are limited in size, thereby avoiding information overload and enabling comfortable human processing.

3. Concept Graphs vs. Object-Process Diagrams

The dual graphic and equivalent natural language representation of the single OPM model is both human understandable and machine processable. A modeling system that comes closest to OPM in its dual graphic-textual representation is Conceptual Graphs (CGs), based on [29, 33]. Table 1 compares both the graphic and the textual CG and OPM model representations of the system represented by the natural language sentence "*John is going to Boston by bus.*" Graphically, the CG model has a compact set of symbols: boxes for concepts, ovals for relations, and arrows for the directed links between concepts and relations. OPM has a richer set of symbols, allowing it to be more expressive. While OPDs use boxes and ovals, like CGs, their semantics is different, denoting respectively objects and processes rather than CG concepts and relations. OPM relations are expressed via the various link types. For example, the link from **John** to **Going**, which ends with the black circle, is the agent link, denoting that the **Person** called **John** is the agent of (the human who executes) the process **Going**. Similarly, the link from **Bus** to **Going**, which ends with the white circle, is the instrument link, denoting that the **Bus** is the instrument of the process **Going**. As noted earlier, these special symbols save the need to annotate these links textually. Agent and instrument links are procedural links: they connect an object and a process.

Other OPD procedural links are the result, consumption, and effect links. In addition to procedural links, OPDs feature a family of structural links, each of which connects an object with an object. An example of a structural link in the OPD in Table 1 is the exhibition-characterization relation, denoted by the black-in-white triangle along the line connecting the **Person John** to the **City**. The exhibition-characterization symbol from **Person** to **Location** states that **Location** is an attribute of Person, The CG makes no underlying semantic difference between "`John`", "`Boston`" and "`Bus`" on one hand and "`Go`" on the other hand: all are concepts. In OPM there is a principal difference between these two entity types: The OPM ontology stipulates that objects are things that exist, while processes are things that transform objects by changing their state or by generating/consuming them. The inability of CGs to distinguish between objects and processes is a major hindrance to enhanced expressive power with respect to system dynamics: CGs may be fine for declarative assertions, i.e., statements about what exists in the world and how what exists related to other things that exist. However, when it comes to describing the dynamics of the system, namely its time-dependent behavior, CGs lacks the basic concept of process and makes no distinction between objects and processes, relating to all as concepts. A somewhat similar difference exists between OPM and the OO paradigm, in which Object is the only top-level concept, and processes can only be expressed as operations that objects own.

Table 1. Comparison between the CG (left) and OPM (right) models of "*John is going to Boston by bus.*"

 Person: John Agnt go Dest City: Boston Inst Bus	 SD Person John Going Bus Location City Boston

`[Go]-`
`(Agnt)->[Person: John]`
`(Dest)->[City: Boston]`
`(Inst)->[Bus].`
	The **Person John** exhibits the **Location City.**
	City is **Boston.**
	John handles **Going.**
	Going requires **Bus.**
	Going changes **City** to **Boston.**

Comparing the two textual representations in Table 1 reveals that while CGs may bear direct mapping to language, they do not translate to any subset of natural language, but rather to a symbolic representation. The OPL sentences, on the other hand, are understandable and the OPL paragraph above can indeed be summarized by the original sentence. Note that the OPL script unfolds a small five-sentence "story." In order to set up the framework for this story, **John** was assigned the attribute **City**, which is an instance of the class **Location** and is assigned the value **Boston**. OPL sentences are written in plain English that is easily readable and understandable to humans with no prior training whatsoever. The same cannot be said about the LF script of CGs, as demonstrated in the bottom left of Table 1. Another quantum leap is still required to convert this script to a natural language sentence like "*John is going to Boston by bus.*"

Summarizing the main differences between CGs and OPM we note that:

(1) The symbol set of CGs is more compact than that of OPDs but expressing the same complex semantics in CGs requires to use at least twice the number symbols required in OPD, yet the semantics is more explicit in OPDs.

(2) The CG formalism is probably better than OPM with respect to support for logic. While OPM does allow AND, OR, and XOR relations, as well as

Boolean objects, it currently does not have the notion of quantifiers and cannot deduce new knowledge from existing knowledge. This is an issue that is being considered for inclusion in the next OPM versions.

(3) The text generated by OPM, the OPL paragraph, is a subset of English, enabling any English speaker to readily understand it, while the LF, the textual form of CGs is still in symbolic form that is not legible to untrained humans.

(4) CGs are purely declarative and have no notion of system dynamics, which is a major feature of OPM.

(5) CGs are not scalable, while OPM has this capability via its scaling mechanisms.

In view of the fundamental differences, the choice of OPM as a basis for the Visual Semantic Web is quite obvious. We continue with a brief survey of RDF and the use of graphics in the Semantic Web.

4. The Semantic Web and the RDF Syntax

RDF, the Resource Description Framework [21, 8] aims at making the knowledge resources that are available on the Web amenable to machine interpretation, compilation, or other types of processing, by imposing some structure on the pieces of knowledge. RDF provides a basis for a number of emerging initiatives, such as the Dublin Core Metadata Initiative [17], an open forum engaged in the development of interoperable online metadata standards. The RDF Syntax document [21] introduces a model for representing RDF metadata as well as a syntax for encoding and transporting this metadata for interoperability of independently developed Web servers and clients. The syntax it presents uses the eXtensible Markup Language (XML), because one of the goals of RDF is to enable specifying semantics for data based on XML in a standardized manner. RDF and XML are complementary in that RDF is a model of metadata and only addresses encoding issues by reference. Such issues include internationalization and character sets, required by transportation and file storage. More importantly, the XML syntax of RDF is only one of the possibilities for encoding RDF and, as noted in [21], alternate ways to represent the same RDF data model may emerge. Indeed, this paper proposes an OPM-based alternative on top of the XML syntax that is human- and machine-oriented at the same time. This syntax enables bimodal, dual graphic-textual representation of the system model for human consumption, while possessing a level of formality that makes it amenable to machine processing.

The Semantic Web makes only limited use of graphical models. In RDF these are directed graphs, where subjects and objects are nodes, and predicates are labels along the edges, directed from a subject to an object. However, since the Semantic Web in its current form and philosophy is geared primarily to cater to needs of machines, and since machines do not need to read diagrams, the visual aspect of the information and knowledge modeling is not well developed. Semantic Web documents show few graphs early on but then abandon them and focus on the XML-based syntactical aspects of the machine-oriented language. RDF is a model for representing named properties and property values [21]. RDF properties may be thought of as attributes of resources and, in this sense, correspond to traditional attribute-value pairs. RDF properties represent relationships between resources, and RDF Schemas, which are instances of RDF data models, are Entity-Relationship (ER) diagrams. The basic RDF data model consists of the following three object types: **Resource –** Anything described by an RDF expression; **Property –** A specific aspect, characteristic, attribute, or relation used to describe a resource; and **Statement –** A specific resource – the *subject*, together with a named property – the *predicate*, plus the value of that property for that resource – the *object*. Following [8], consider first the sentence "*Ora Lassila is the creator of the resource http://www.w3.org/Home/Lassila.*" Translated to RDF format, this sentence can be interpreted as having the subject (resource) http://www.w3.org/Home/Lassila, the predicate (property) creator, and the object (literal) "Ora Lassila". RDF uses directed graphs to specify these notions graphically, where subjects and objects are nodes, and predicates are labels along the edges, which are always directed from a subject to an object, as in Figure 1. A resource node in the graph is drawn as an oval (ellipse), while a literal node is drawn as a rectangle.

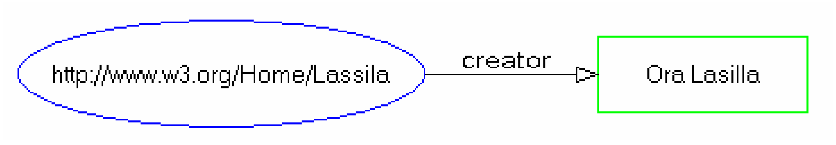


Figure 1. A simple RDF graph example from [21]

the graph in Figure 1 is to be interpreted as "*http://www.w3.org/Home/Lassila has creator Ora Lassila*", and in general "*<subject> HAS <predicate> <object>*". Applying the RDF/XML Validation Service [38] using the RDF/XML script in Table 2 yields the graph in Figure 2.

Table 2. The RDF/XML script that generates the graph in Figure 2.

```
<?xml version="1.0"?>
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:s="http://description.org/schema/">
  <rdf:Description about="http://www.w3.org/Home/Lassila">
    <s:Creator>Ora Lassila</s:Creator>
  </rdf:Description>
</rdf:RDF>
```

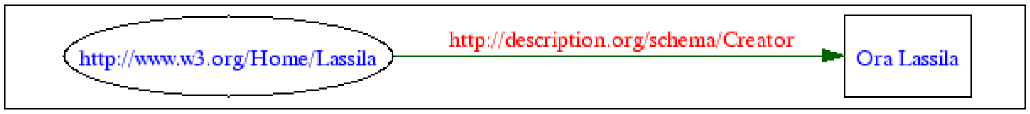

http://www.w3.org/Home/Lassila
http://description.org/schema/Creator
Ora Lassila

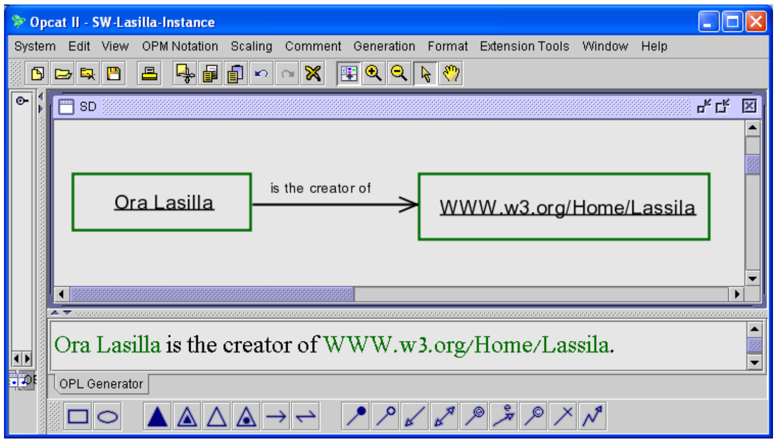
Figure 2. The RDF graph of the data model listed in Table 2 generated automatically by the RDF/XML Validation Service [37]

The Semantic Web is based on a principle similar to that of OPM, where relations (called properties in the SW nomenclature) are edges of a graph rather than nodes, as in CGs. The Visual Semantic Web [16] (ViSWeb) alternative to the RDF/XML knowledge representation [13] takes advantage of the integrated graphic-text formal yet intuitive infrastructure that OPM provides. Figure 3 is a ViSWeb spec (Visual Semantic Web specification), which expresses the example in Figure 1 in a bimodal fashion, both as an Object-Process Diagram (OPD) and an Object-Process Language (OPL) text. The OPD contains two object instances: **Ora Lasilla** and WWW.w3.org/Home/Lassila. To conform to

OMG UML 1.4 [10, 15], object names (i.e., instances of object classes) in OPDs are underlined, as in UML object diagrams.

A tagged structural link, depicted as an open arrow, such as the one pointing from **Person** to **URI** in Figure 4, expresses the nature of the relation between these two objects. The tag is the text recorded along the structural link. The value of this tag is '**is the creator of**'. The value is a phrase, such that when the name of the source object, **Ora Lasilla** (an instance of the class **Person**) is concatenated with the tag value (i.e., the phrase) '**is the creator of**' followed by the name (value) of the **URL**, one automatically gets the following OPL sentence, which is also generated automatically by OPCAT and recorded at the bottom of the OPD in Figure 3.

Ora Lasilla is the creator of WWW.w3.org/Home/Lassila**.**



Opcat II - SW-Lasilla-Instance
System Edit View OPM Notation Scaling Comment Generation Format Extension Tools Window Help
SD
Ora Lasilla
is the creator of
WWW.w3.org/Home/Lassila
OPL Generator

Ora Lasilla is the creator of WWW.w3.org/Home/Lassila.

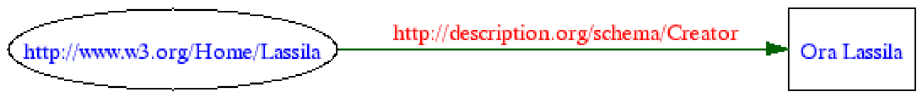
Figure 3. The example in Figure 1 expressed as a ViSWeb spec (Visual Semantic Web specification), consisting of an Object-Process Diagram (OPD) at the top window and its corresponding, automatically-generated Object-Process Language (OPL) one-sentence paragraph at the bottom window.

The automatic generation of the OPL sentence in this simple case was done by concatenating the name of the object at the source of the tagged structural link, **Ora Lasilla**, with the text string of the structural link's tag, **is the creator of**, with the name of the destination object, WWW.w3.org/Home/Lassila.

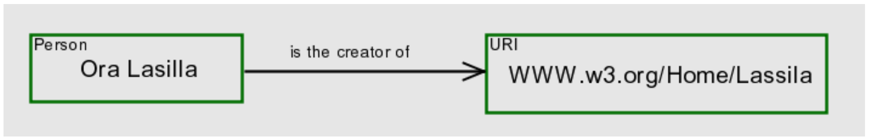
In RDF terminology, this OPL sentence is a *statement*, in which a specific resource – the *subject*, **Ora Lasilla** in our case, together with a named property – the *predicate*, '**is the creator of**' in our case, plus the value of that property for that resource – the *object*, WWW.w3.org/Home/Lassila in our case. Each word in an object (and process) name is capitalized, while in link names (tags) they are not. As Figure 5 shows, names of objects and link names (tags) appear in different colors (which can be set by the user). Even though there are spaces between the words, using the capitalization rule above it is possible to mechanically parse the sentence even without the human-oriented color cues. Table 3 compares RDF and OPM with respect to this example. For each method, the three elements and the respective parts of the example are written first, and below them are the graphical and textual representations of the RDF graph in Figure 1 and the OPD in Figure 5.

Table 3. Comparison between RDF and OPM applied to the example in Figure 1 and Figure 5

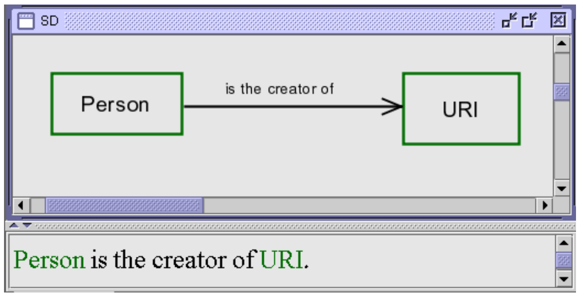
RDF/XML	*Object (domain):*
	`Http://www.w3.org/Home/Lassila`
		Predicate (property):
		`Creator`
			Subject (range):
			`Ora Lassila`

	Graphics:  http://www.w3.org/Home/Lassila http://description.org/schema/Creator Ora Lassila		

	Text:
	`<?xml version="1.0"?>`
	`<rdf:RDF`
	`  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"`
	`  xmlns:s="http://description.org/schema/">`
	`  <rdf:Description about="http://www.w3.org/Home/Lassila">`
	`    <s:Creator>Ora Lassila</s:Creator>`
	`  </rdf:Description>`
	`</rdf:RDF>`		

OPM/ViSWeb	*Source object:* **Ora Lasilla**	*Structural link tag:* **is the creator of**	*Destination object:* WWW.w3.org/Home/Lassila
	Graphics:  Person Ora Lasilla is the creator of URI WWW.w3.org/Home/Lassila		
	Text: **Ora Lasilla is the creator of** WWW.w3.org/Home/Lassila**.**		

While the OPM model still does not account for namespaces, which are treated in the sequel, comparing this OPL/ViSWeb sentence to the RDF/XML script in Table 3, it is not difficult to see the benefit of using a more human-readable version, which, while still machine-readable, does not require the human reader to act like a mechanical XML parser.



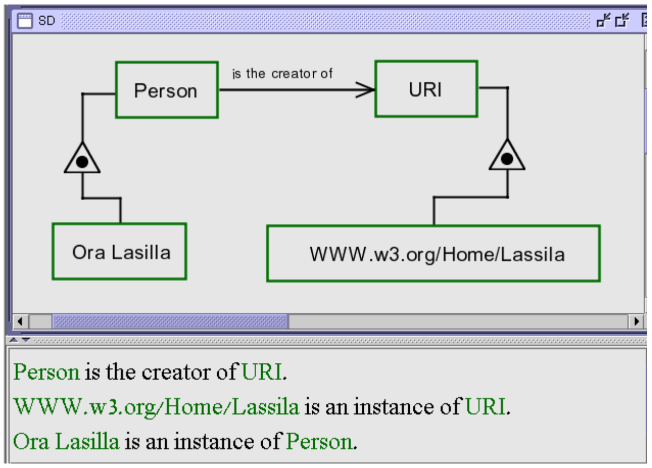
Person is the creator of URI.

Figure 4. A ViSWeb schema showing the class OPD along with its corresponding OPL sentence, to which the ViSWeb spec in Figure 3 conforms

5. The ViSWeb Schema: A Template for a ViSWeb Spec

The lines under the two objects in Figure 3 denote the fact that these are object instances, not object classes. The class information is still missing in this OPD. Figure 4 shows a *ViSWeb schema*, an OPD-OPL template that contains class information. Note that the ViSWeb schema follows the OPM text-graphic equivalence principle: the OPD and the OPL paragraph are completely reconstructible from each other. Each ViSWeb spec conforms to a ViSWeb schema. Thus, the ViSWeb schema in Figure 3 conforms to the ViSWeb spec in Figure 4. This ViSWeb schema can be thought of as a template that expresses a rule. In our example, the rule stipulates that the source (which in RDF schema terminology is termed the *domain*) of the relation (the RDF *predicate*) '**is the creator of**' is an object that belongs to the class **Person**, and that the destination (*range*) of that relation is an object that belongs to the class **URI**.

Having established the **Person-URI** ViSWeb schema, we can now use it to add the object instance for each of the two classes. This is done in the instantiated schema shown in Figure 5, where the ViSWeb schema of Figure 4 and the ViSWeb spec of Figure 3 are combined. The combination uses the OPM classification-instantiation relation, which is denoted as a bulleted triangle whose tip is linked to the class and whose base is linked to the instance. Note that the instances **Ora Lasilla** and WWW.w3.org/Home/Lasilla need not be underlined here to denote that they are instances. The underlining of the instance names is only mandatory if the class information is not present in the OPD, but here this is indicated by the classification-instantiation links from the classes to the respective instances.



Person is the creator of URI.
WWW.w3.org/Home/Lassila is an instance of URI.
Ora Lasilla is an instance of Person.

Figure 5. The instantiated ViSWeb schema generated by adding the instance specification of Figure 3 to the class information in the ViSWeb schema in Figure 4.

The OPL paragraph of the OPD in Figure 5 is shown in Figure 5 as well:

Person is the creator of URI.
WWW.w3.org/Home/Lassila is an instance of **URI.**
Ora Lasilla is an instance of **Person.**

Note that predicates (such as **is the creator of**) do not have explicit instance names that are distinct from their class names. Thus, for example, we use the same predicate in The tagged structural relation '**is the creator of**' from the class **Person** to the class **URI** is inherited to their respective instances, so there is an implicit tagged structural relation with the same tag, '**is the creator of**', from **Ora Lasilla**, an instance of the class **Person**, to WWW.w3.org/Home/Lassila, an instance of the class **URI**. Applying template information and using chaining rules one can establish that **Ora Lasilla is the creator of** WWW.w3.org/Home/Lassila although this is not explicit in the OPM model in Figure 5. However, the instantiated ViSWeb schema in Figure 5 is space-consuming and it requires the reader to realize the existence of the implicit tagged structural relation. These two problems are solved in the compact version of the instantiated ViSWeb schema of Figure 5, shown in Figure 6.

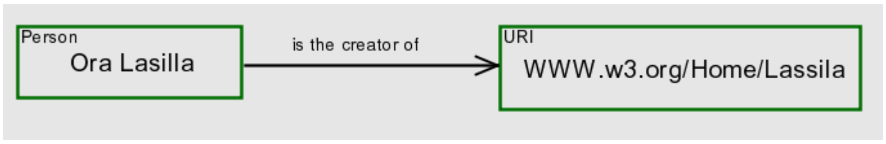


Figure 6. A compact version of the instantiated ViSWeb schema in Figure 5

The OPL paragraph that corresponds to the OPD in Figure 6 is also more compact than the three-sentence OPL paragraph of Figure 5, as it consists of just one sentence:

The **Person Ora Lasilla is the creator of** the **URI** WWW.w3.org/Home/Lassila**.**

This sentence combines the OPL schema sentence from Figure 3, which is "**Person is the creator of URI.**" with the OPL instance sentence Figure 4, which is "**Ora Lasilla is the creator of** WWW.w3.org/Home/Lassila**.**" In the new OPL sentence, which reflects both the classes and the instances, we added the class information of both **Ora Lasilla**, which is **Person**, and of WWW.w3.org/Home/Lassila, which is **URI**. **Ora Lasilla** is classified in the OPL sentence as belonging to the class **Person** by preceding the name of the instance by the reserved word "The" followed by the class name **Person**. Likewise, the string WWW.w3.org/Home/Lassila was classified as belonging to the class **URI** by preceding the value of the string by the reserved word the followed by the class name **URI**. The corresponding quoted sentence is `The 'Person' 'Ora Lasilla' 'is the creator of' the 'URI' 'WWW.w3.org/Home/Lassila'.`

6. OPM Namespace Specification

Namespaces [7] are definitions of terms and relations of some domain ontology. The OPL sentence "The **Person Ora Lasilla is the creator of** the **URI** WWW.w3.org/Home/Lassila**.**" does not specify the namespaces which contain the definitions of **Person**, **URI**, and the structural link tag (predicate) '**is the creator of**'. In contrast, the XML script in Table 3 does mention two namespaces, `rdf` and `docs`. The namespaces, which are part of the XML tags, enable us to know that we are looking at a `Description` in the sense defined in the `rdf` namespace definition, that the value of its `about` attribute is `"http://www.w3.org/Home/Lassila"`, and that the value of the `Creator` entity, as defined in the `docs` namespace, is `Ora Lassila`. This information is clearly richer than what can be extracted from the RDF graph in Figure 1, since that graph does not specify any namespace information.

The RDF graph is only an auxiliary means to make it easier for humans to "get the picture." It is not required to contain all the information expressed by the corresponding XML script and therefore cannot replace it (although the RDF Validator [37] does so, albeit in a manner that is not very user friendly). The OPM text-graphics equivalence principle mandates that any piece of information contained in the OPL paragraph that corresponds to an OPD be represented in the OPD, and vice versa, making the OPD and its OPL paragraph fully equivalent in terms of information content. To keep up with the text-graphics equivalence principle, we introduce the concept of namespace to both the OPD and the OPL.

Let us assume that the subject **Person** and the object **URI** are both defined in the namespace whose name is **Semantic Web** and whose URI is WWW.SemanticWeb.org/definitions. We further assume that the predicate '**is the creator of**' is defined in the namespace whose name is **Documents** and whose URI is WWW.Documents.org/definitions. The OPD in Figure 7 elaborates on that of Figure 6, as it provides the complete namespace information. The ViSWeb convention is to stack all the namespaces used in the OPD at its top left corner. Each namespace is recorded in an object box, with the string "**Namespace:** `<blank>` `<namespace_name>`" appearing at the top left corner of the box and the corresponding URI recorded at the bottom of the box. Here, `<namespace_name>` is the name of the namespace. Two namespaces names appear in Figure 7: **Semantic Web** and **Documents**. Using these two namespace specifications, instances in an OPD can be annotated not just with the class specification, as in Figure 6, but also with the namespace within which the class is specified, preceding the class name, as in Figure 7. Thus, **Semantic Web: Person** is the complete namespace and class specification of the **Person** instance **Ora Lasilla**, and **Semantic Web: URI** is the complete namespace and class specification of the **URI** instance WWW.w3.org/Home/Lassila. Finally, **Documents:** is the complete namespace specification of the predicate (tagged structural link in OPM terminology) '**is the creator of**'.

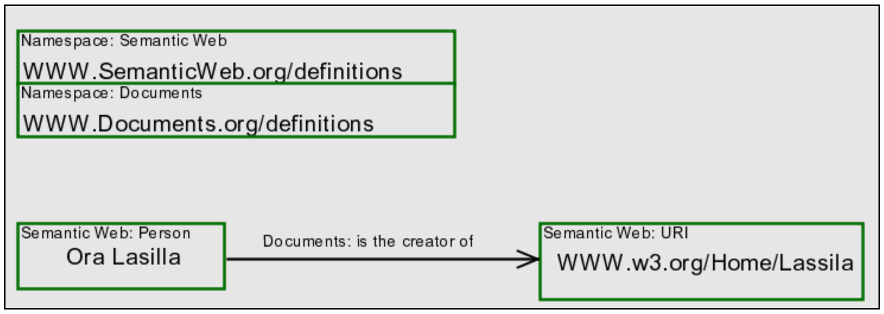


Figure 7. The OPD of Figure 5 with the namespace object boxes at the top left corner of the OPD.

The following two OPL namespace declaration sentences are the textual equivalents of the two namespace boxes stacked at the top left of Figure 7.

The namespace **Semantic Web** is at URL WWW.SemanticWeb.org/definitions**.**

The namespace **Documents** is at URL WWW.Documents.org/definitions**.**

Just as namespace graphical specifications in an OPD are part of the graphical syntax of ViSWeb, OPL namespace declaration sentences are part of the textual syntax of ViSWeb. Based on the above two namespace declarations, the following class and relation definition sentences are:

The namespace **Semantic Web** defines the class **Person.**

The namespace **Semantic Web** defines the class **URL.**

The namespace **Documents** defines the relation **'is the creator of'.**

The Default Namespace Convention

Usually, most if not all the names in a single OPD are defined in the same namespace. Thus, it is redundant and cumbersome to specify separately for each name that it is defined within that namespace. A simplifying OPM *default namespace convention* is that the namespace at the top of the namespace stack in the OPD is the default namespace, so any class in the OPD which is defined within this default namespace does not require that the namespace name precedes it. In our example, the default namespace declaration sentence is:

The default namespace **Semantic Web** is at WWW.SemanticWeb.org/definitions**.**

Applying this default namespace convention, the OPL paragraph (collection of OPL sentences) that corresponds to the OPD in Figure 7 is:

The default namespace **Semantic Web** is at WWW.SemanticWeb.org/definitions**.**

The namespace **Documents** is at WWW.Documents.org/definitions**.**

The namespace **Documents** defines the relation **'is the creator of'.**

The **Person Ora Lasilla is the creator of** the **URI** WWW.w3.org/Home/Lassila**.**

The XML script that specifies the analogous semantics, where the **Semantic Web** namespace is replaced by `rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"` and the **Documents** namespace is replaced by `s="http://description.org/schema/"` is the one listed in Table 2.

Wrapping the ViSWeb specification with XML and the DRF Equivalent

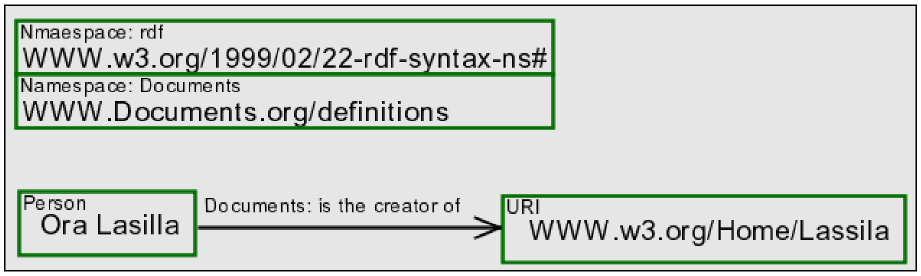
Having finalized the ViSWeb specification, we now describe how it is all wrapped with XML for being amenable to Web transfers and manipulations. We call this form XML/ViSWeb. We also describe how this XML/ViSWeb is translated into the XML/RDF standard. The following namespace declaration sentence, which is a constant part of any XML/ViSWeb text, specifies the OPM namespace:

The namespace **OPM** is at WWW.ObjectProcess.org/definitions**.**

The entire ViSWeb OPL specification is incorporated into the XML syntax by simply enclosing it within the `<OPM:OPL>` and `</OPM:OPL>` tags, as shown in Table 4. The OPD is likewise enclosed within the `<OPM:OPD>` and `</OPM:OPD>` tags. For human consumption, the actual graphic display of the OPD is presented in the XML/ViSWeb specification, as shown in Table 4. For machines, the actual OPD is replaced in the corresponding XML/RDF script by its XMI [11] representation. Comparing the human-oriented XML/ViSWeb specification to its corresponding machine-oriented XML/RDF translation in Table 4, the advantages for humans of the former over the latter are evident:

- Graphically, the machine-oriented XMI [28] specification of the ViSWeb OPD is rendered and displayed for humans as an intelligible diagram that contains the same information as the corresponding ViSWeb OPL script below it.
- Textually, the ViSWeb OPL script contains only sentences in a subset of natural English, which humans can read and understand with significantly less effort than required for performing "mental compilation." Such mental compilation is what humans are effectively required to execute when they encounter any XML/RDF script that they wish to interpret.

Table 4. Comparison between the complete human-oriented XML/ViSWeb specification of our example and its corresponding machine-oriented XML/RDF translation

Human-oriented XML/ViSWeb	`<?xml version="1.0"?>`
	`<OPM:OPD>`
	
	 Nmaespace: rdf
	WWW.w3.org/1999/02/22-rdf-syntax-ns#
	Namespace: Documents
	WWW.Documents.org/definitions
	Person
	Ora Lasilla
	Documents: is the creator of
	URI
	WWW.w3.org/Home/Lassila
	`</OPM:OPD>`
	`<OPM:OPL>`
	The namespace **OPM** is at WWW.ObjectProcess.org/definitions**.**
	The default namespace **rdf** is at WWW.w3.org/1999/02/22-rdf-syntax-ns#**.**
	The namespace **Documents** is at WWW.Documents.org/definitions**.**
	The namespace **Documents** defines the relation **'is the creator of'.**
	The **Person Ora Lasilla is the creator of** the **URI** WWW.w3.org/Home/Lassila**.**
	`</OPM:OPL>`

Machine-Oriented XML/RDF	`<?xml version="1.0"?>`
	`<OPM:OPD>`
	`-- Here comes the XMI [28] specification of the OPD, which enables its rendering shown above for human consumption. --`
	`</OPM:OPD>`
	`<OPM:OPL>`
	` xmlns:OPM="http://www.ObjectProcess.org/Definitions"`
	` <rdf:RDF>`
	`  xmlns:rdf=http://www.w3.org/1999/02/22-rdf-syntax-ns#`
	`  xmlns:Documents="http://documents.org/defintions">`
	`   <rdf:Description about="http://www.w3.org/Home/Lassila">`
	`   <Documents:'is the creator of'>`
	`     Ora Lassila`
	`   </Documents:'is the creator of'>`
	`   </rdf:Description>`
	`  </rdf:RDF>`
	`</OPM:OPL>`

The idea, then, is to show humans human-oriented XML/ViSWeb specification, exemplified by the top part of **Table 4**, while the machine will still be able to process its "pure" XML/RDF translation, shown at the bottom of **Table 4**. In order to do this, we must have a utility for bi-directional translation between ViSWeb and RDF.

7. Adding Attributes

Continuing with the example from [21], for specifications that are more complex, a compound resource can be created, as the following sentence and the corresponding graph in Figure 8 demonstrate:

"The individual referred to by employee id 85740 is named Ora Lassila and has the email address lassila@w3.org. The resource http://www.w3.org/Home/Lassila was created by this individual."

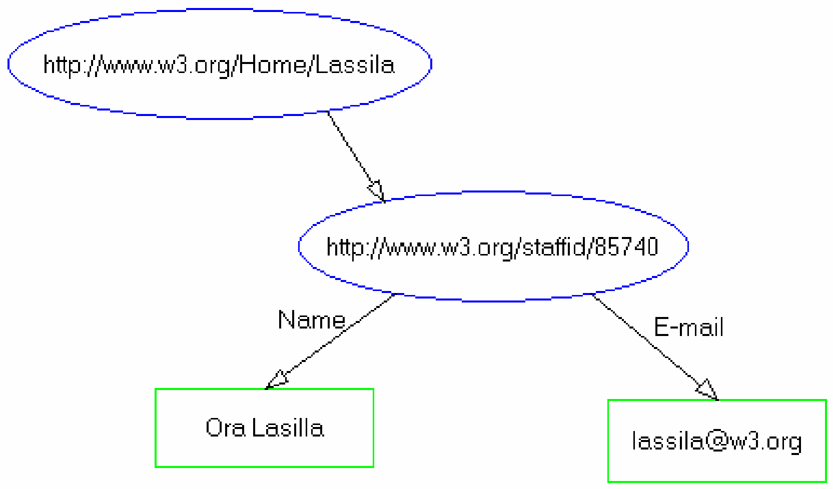

http://www.w3.org/Home/Lassila
http://www.w3.org/staffid/85740
Name
E-mail
Ora Lasilla
lassila@w3.org

Figure 8. An identified property with structured value [21]

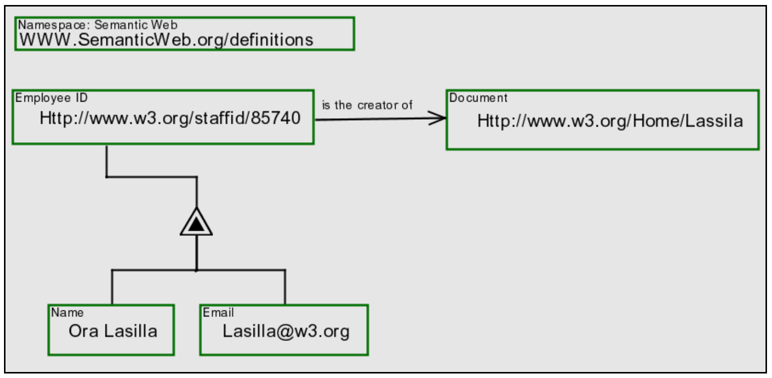


Figure 9. The OPD that corresponds to the graph in Figure 8

The OPL paragraph that corresponds to the OPD in Figure 9 is:

The default namespace **Semantic Web** is at WWW.SemanticWeb.org/definitions.
The **Employee ID** WWW.w3.org/staffid/85740 **is the creator of** the **Document** WWW.w3.org/Home/Lassila.
The **Employee ID** WWW.w3.org/staffid/85740 exhibits the **Name Ora Lasilla** and the **Email** Lasilla@w3.org.

The OPL reserved word exhibits expresses the exhibition-characterization relation (the relation between a class and its attributes, symbolized by a black-in-white triangle) from

The **Employee ID** Http://www.w3.org/staffid/85740 to the **Name Ora Lasilla** and to the **Email** Lasilla@w3.org.

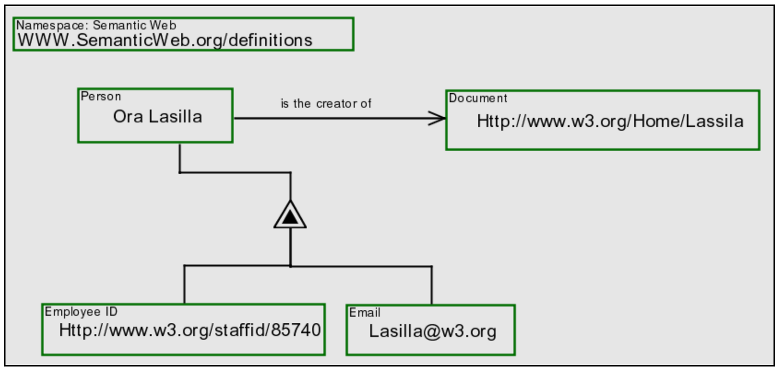


Figure 10. A better representation of the information presented in the OPD in Figure 9

A better representation of the information presented in the OPD in Figure 9 is shown in the OPD of Figure 10. The **Employee ID** is now an attribute of the **Person** rather than the other way around. That this is a better way of modeling is clearly seen when we compare the OPL paragraph below, which corresponds to the OPD in Figure 10, to the previous OPL paragraph, which corresponds to the OPD in Figure 9.

The default namespace **Semantic Web** is at WWW.SemanticWeb.org/definitions**.**

The **Person Ora Lasilla is the creator of** the **Document** WWW.w3.org/Home/Lassila**.**

The **Person Ora Lasilla** exhibits the **Employee ID** WWW.w3.org/staffid/85740 and the **Email** Lasilla@w3.org**.**

8. Advantages of the Visual Semantic Web Paradigm

The ViSWeb paradigm has a number of important advantages over present OWL/RDF/XML approaches, which are summarized in this section.

1. **Graphic-text knowledge representation:** The powerful graphic-text bimodal representation of OPM is extended to the Visual Semantic Web paradigm. Rather than having to mentally parse cryptic XML scripts, knowledge is presented to the user in a subset of natural language as well as diagrammatically. The two modalities complement each other, so if something is unclear in one representation, the other can be consulted for clarification. Using ViSWeb, one can ask for a translation from XML/RDF to XML/ViSWeb in order to get both visualization and a human-readable version of the XML syntax. The graphic representation can then be manipulated, changed, or augmented. Any such change would be reflected in the ViSWeb OPL script, and through it transparently back to the XML/RDF machine-oriented syntax. This way, working in a round-trip engineering mode, the human gets to think and develop ideas in a user-friendly environment without compromising the technical soundness of her/his work.

2. **Visual navigability:** In addition to the advantages of putting to work the "two sides of the human brain," the visual and the lingual, there are benefits that are unique to the Semantic web. The formal robust, yet intuitive, diagrammatic display enables users to surf and navigate the Web in a visual way in search for knowledge.

3. **Semantic sentence interpretation:** In spite of the aspiration of the Semantic Web, the basis of the RDF framework is syntactic rather than semantic: it draws on the concepts of *subject*, *predicate* and *object*, which are parts of speech used to analyze natural language sentences from a syntactic viewpoint. The same semantics can be expressed by inverse syntactic expressions. For example, without changing the semantics, we could easily switch the roles of subject and object in the example of Figure 1 by writing the sentence as "*The resource http://www.w3.org/Home/Lassila was created by Ora Lassila.*" Now the (syntactic) subject is http://www.w3.org/Home/Lassila, the object is *Ora Lassila*, and the predicate is "*was created by*". While the parts of speech are turned upside down and the predicate was changed from active to passive, the meaning of the sentence is still the same. The proposed OPM-based ViSWeb paradigm is based on a sound ontology of *objects* with *states* and *processes*: Objects are things that (at least potentially, and possibly at some state) exist, while processes are things that happen to objects and transform them (i.e., create or destroy them, or change their state). Based on this ontology, sentences can be interpreted semantically rather than syntactically. In OPM, each structural relation pair has a forward direction and a backward direction [14], so for example the forward relation "*is the creator of*" is paired with "*was created by*." This helps overcome syntactic differences and establish semantic equivalence.

4. **Specification of system dynamics:** Current work on the Semantic Web places emphasis on declaratively specifying structural knowledge, which relates to the static aspect of systems. Structural knowledge pertains to relations among objects that are not related to the objective of the system or the way it operates. According to Berners-Lee [3], "*the RDF model is basically an opening of the ER model to work on the Web.*" A typical ER model involves entity types, each with its set of relationships. "*The RDF model is the same, except that relationships are first class objects: they are identified by a URI, and so anyone can make one.*" This is a purely static world view, where everything can be expressed in terms of structural, time-independent relations. However, a major part of the knowledge about a system is functional (what is its purpose) and dynamic (how it operates). The current SW offers very little in this regard. Since OPM combines function, structure, and behavior in the same bimodal model, it provides a sound infrastructure for representing system dynamics and function in the ViSWeb model. While the details are beyond the scope of this work, suffice it to mention that knowledge about reactive and real-time systems requires treatment of events, conditions, actions, state transitions, and time exceptions, to name but a few major issues. All those and mode can be modeled in OPM.

5. **Complexity management:** A major problem in real-life systems is their complexity due to the sheer amount of knowledge details. In addition to the OWL [32] set operators that can be translated into OPM as demonstrated above, OPM has built in abstraction-refinement mechanisms, including in-zooming and out-zooming, unfolding and folding, and state expression and suppression. These provide for building hierarchies of knowledge representation in general and over the Web in particular, enabling navigation up and down abstraction-refinement hierarchies.

9. Summary and Future Work

The Visual Semantic Web (ViSWeb) paradigm proposes to unify human and machine representations of knowledge. The foundation for this unification is Object-Process

Methodology (OPM), which advocates the integration of a system's structure and behavior is a single, graphic and textual model. The paper has presented the principles and outlined an implementation for the ViSWeb. Like OPM, the ViSWeb model enables the representation of static and dynamic knowledge using a combination of Object-Process Language (OPL), a subset of English, and Object-Process Diagrams (OPDs), an equivalent visual formalism. The advantages of this approach include graphic-text knowledge representation, visual navigability, semantic sentence interpretation, specification of system dynamics, and complexity management. As noted in [21], "*It is also important to understand that this XML syntax is only one possible syntax for RDF and that alternate ways to represent the same RDF data model may emerge.*" Indeed, this work presents an OPM-based approach to representing the Semantic Web on top of the RDF data model, which is expressed graphically, using OPDs, and textually in OPL, a subset of natural English which is also "machine understandable," i.e., amenable to parsing and converting back to the XML-based RDF syntax.

Future work will proceed in both the theoretical and practical paths. The theory will focus on extending the idea behind the ViSWeb paradigm and its initial specification, presented in this work, to cover other important knowledge and system representation aspects. Based on OPM, ViSWeb will be able to handle not only the declarative static structural aspects of knowledge, which is the focus of the current Semantic Web initiative, but also procedural, dynamic behavioral aspects, as well as functional ones. The practical work will augment the current capabilities of OPCAT so it will be suitable for modeling the various ViSWeb requirements presented here, and provide the services of bi-directional RDF-ViSWeb compilation. An even more ambitious goal is to design and build a Web crawler which will automatically generate ViSWeb representations of knowledge stored in Web pages. Accomplishing even some of these goals will greatly benefit the huge World Wide Web user community by providing them with a friendly semantic surfing tool and relieving them from the need to mentally compile XML scripts.

References

[1] Anderson, J. R., Bower, G. H. Human associative memory. Winston and Sons, Washington, D.C., 1973.

[2] Arnheim, R. Visual Thinking. University of California Press, Berkeley, California, 1969.

[3] Berners-Lee, T. What the Semantic Web can represent, 1998. http://www.w3.org/DesignIssues/RDFnot.html

[4] Berners-Lee, T. and Hendler, J. Scientific publishing on the semantic web. Nature. 2001. http://www.nature.com/nature/debates/e-access/Articles/bernerslee.htm

[5] Berners-Lee, T. Hendler J. and Lassila, O. The Semantic Web. Scientific American, May 2001.

[6] Brachman. R. On the epistemological status of semantic networks. In Associative Networks: Representation and Use of Knowledge by Computer. N.V. Findlee, Ed. Academic Press. New York, NY, pp. 3-50, 1979.

[7] Bray, T., Hollander, D. and Layman, A. Namespaces in XML, World Wide Web Consortium Recommendation, 14 January, 1999. http://www.w3.org/TR/REC-xml-names

[8] Brickley, D. and Guha R.V. RDF Vocabulary Description Language 1.0: RDF Schema, W3C work in progress draft, 30 April, 2002. http://www.w3.org/TR/rdf-schema

[9] Chein, M., M.L.Mugnier.Conceptual Graphs:Fundamental Notions.Revue d'Intelligence Artificielle,vol.6,n.4,p.365-406,1992.

[10] Corby, O., Dieng, R., and Hebert C. A Conceptual Graph Model for W3C RDF Proceedings of the Int. Conf. on Conceptual Structures (ICCS), 2000. http://www.int.gu.edu.au/kvo/reading/oliviericcs2000.pdf

[11] Cyc. OpenCyc, org, 2002. http://www.opencyc.org/

[12] Delteil, A. Faron, C. A Graph-Based Knowledge Representation Language. Proc. 15th

European Conference on Artificial Intelligence (ECAI) 2002. http://www-sop.inria.fr/acacia/personnel/Alexandre.Delteil/ecai.pdf

[13] Delugach, H. Conceptual Graphs Homepage, 2003. http://www.cs.uah.edu/~delugach/CG/

[14] Dori, D. Object-Process Methodology - A Holistic Systems Paradigm, Springer Verlag, Berlin, Heidelberg, New York, 2002. www.ObjectProcess.org

[15] Dori, D. Why Significant Change in UML is Unlikely. Communications of the ACM, pp. 82-85, Nov. 2002.

[16] Dori, D. Reinhartz-Berger, I. and Sturm, A. OPCAT – A Bimodal CASE Tool for Object-Process Based System Development. Proc. IEEE/ACM 5th International Conference on Enterprise Information Systems (ICEIS 2003), Angers, France, pp. 286-291, 2003. www.ObjectProcess.org

[17] Dublin Core Metadata Initiative. 2002. http://www.dublincore.org/

[18] Gaines, B.R. and Shaw, M.L.G. Concept maps as hypermedia components. International Journal of Human Computer Studies, 43(3), pp. 323-361, 1995.

[19] Genesereth, M.R. Knowledge Interchange Format. Draft proposal American National Standard (dpANS), NCITS.T2/98-004. 1998. http://logic.stanford.edu/kif/dpans.html

[20] Lassila, O. and Swick, R. Resource Description Framework (RDF) Model and Syntax Specification. W3C Recommendation, 22 February, 1999. http://www.w3.org/TR/REC-rdf-syntax

[21] Lehman, F. (Ed.) Semantic Networks in Artificial Intelligence. Pergamon Press, Oxford, UK, 1992.

[22] Martin, P. and Eklund, P. Embedding knowledge in web documents: CGs versus XML metadata languages. Proc. 7th Int. Conf. on Conceptual Graphs (ICCS'99), Springer-Verlag, 1999.

[23] Mayer, R.E. Multimedia Learning. Cambridge University Press, NewYork, NY, 2001.

[24] McTear, M.F. (Ed.) Understanding Cognitive Science. Ellis Horwood, Chichester, UK, 1988.

[25] Novak, J.D. A Theory of Education. Cornell University Press, Ithaca, Illinois, 1977.

[26] Novak, J.D. and Gowin, D.B. Learning How to Learn. Cambridge University Press, New York, NY, 1984.

[27] OMG UML1.4. Object Management Group, Unified Modeling Language Version 1.4, September 2001. http://www.omg.org/technology/documents/formal/uml.htm

[28] OMG XMI. Object Management Group, XML Metadata Interchange (XMI) Specification Version 1.2, January 2002. http://cgi.omg.org/docs/formal/02-01-01.pdf

[29] Peirce, C.S. Collected Papers of Charles Sanders Peirce. In Hartshorne, C. and Weiss, P. (Eds.), Harvard University Press, Cambridge, MA, 1932.

[30] Peleg M. and Dori, D. The Model Multiplicity Problem: Experimenting with Real-Time Specification Methods. IEEE Transaction on Software Engineering, 26, 8, pp. 742-759, 2000.

[31] Pepper, S. and Moore G. XML Topic Maps (XTM) 1.0. TopicMaps.Org Specification, 2001. http://www.topicmaps.org/xtm/1.0/

[32] Smith, M. K., McGuinness, D. Volz, R., and Welty, C. Web Ontology Language (OWL) Guide Version 1.0. W3C Working Draft, 4 November, 2002. http://www.w3.org/TR/2002/WD-owl-guide-20021104/

[33] Sowa, J.F. Conceptual Structures: Information Processing in Mind and Machine. Addison-Wesley, Reading, MA, 1984.

[34] Sowa, J.F. Conceptual Graph Standard, 2000 http://users.bestweb.net/~sowa/cg/cgstandw.htm#Header_44

[35] Sowa, J.F. The Common Logic Standard initiative. 2002 http://suo.ieee.org/email/msg08241.html

[36] Sowa, J.F. (1999) Knowledge Representation: Logical, Philosophical, and Computational Foundations. Brooks Cole Publishing Co., Pacific Grove, CA.

[37] W3C RDF Validation Service, 2003. http://www.w3.org/RDF/Validator/