Object-Process Methodology and Its Application to the Visual Semantic Web

Pre-Conference Tutorial PT1
ER-2003, Chicago
October 12, 2003

Dov Dori
Technion, Israel; MIT, USA
www.ObjectProcess.org
What is Object-Process Methodology (OPM)?

- A comprehensive patented systems modeling, engineering, and lifecycle support paradigm

- Two major features:
  - Unification of function, structure and behavior in a single model
  - Bi-modal expression of the model via intuitive yet formal graphics and equivalent natural language
OPM’s Building Blocks are Things: Objects and Processes
States are situations of objects
A process changes an object state
A process generates a new object
Combining Behavior with Structure

Robot can be disengaged or engaged.
Couple consists of Robotic Arm and Workpiece.
Engaging changes Robot from disengaged to engaged.
Engaging yields Couple.
Why Object-Process Methodology?

- As the inherent complexity and interdisciplinary nature of systems increases, the need for a universal modeling, engineering, and lifecycle support approach becomes ever more essential.

- The unnecessary complexity and software orientation of UML – the current standard language – calls for a simpler, formal, generic paradigm for systems development.
Complexity Management
Animated Simulation: Starting
Animated Simulation:
Engaging activated
Animated Simulation:
Locating occurs, Workpiece affected
Animated Simulation:
Gripping occurs, Couple generated
OPM Elements: Entities and Links

- Entities:
  - Object
  - State
  - Process

- Links:
  - Structural link
  - Procedural link
OPM Entities

- **Object**: A thing that exists for some time
- **State**: A situation at which an object can be
- **Process**: A thing that transforms an object
OPM Structural Links

Links denoting persistent relations between objects

Fundamental:
- Aggregation-participation
- Exhibition-characterization
- Generalization-specialization
- Classification-instantiation

General:
- Unidirectional tagged structural relation
- Bidirectional tagged structural relation
Aggregation-participation

A structural relation between the whole and its parts

*Car*, which is physical, can be *broken*, ultimately *junk* or initially *drivable.*

*Car* consists of *Steering Wheel, Engine, Body* and *Wheel.*
Exhibition-characterization

A structural relation between a thing and its features

Car, which is physical, exhibits Model, Year, Make and Seats.

Car consists of Steering Wheel, Engine, Body and Wheel.
A structural relation between a thing and its specializations (known as the “is-a” relation)

- SUV is a Car.
- Wagon is a Car.
- Sedan is a Car.
- Car, which is physical, exhibits Model, Year, Make and Seats.
A structural relation between a thing and its instances

My SUV is an instance of SUV.
Car, which is physical, exhibits Model, Year, Make and Seats.
SUV is a Car.
General tagged structural link

A structural relation between two things whose semantics is expressed through its tag.
General tagged structural link

A structural relation between two things whose semantics is expressed through its tag.
General tagged structural link

A structural relation between two things whose semantics is expressed through its tag.

Bidirectional tagged structural link

Unidirectional tagged structural link
Bi-directional tagged structural link with two tags

Each tag results in a separate sentence.

Germany is east of France.
France is west of Germany.
A bi-directional tagged structural link with one tag

A structural relation between two things whose semantics is expressed through its tag.
Null tag structural link

A structural relation between two things with an empty tag

Unidirectional null tag OPL

Bidirectional null tag OPL

Couple is related to Account.
Land and Country are equivalent.
Participation constraints

Structural relation can be associated with various quantities

Many Floors can be climbed in a Skyscraper.

Skyscraper is constructed of many Floors.
Participation constraints

The aggregation-participation link can have participation constraints on the parts.
Setting participation constraints

Each part in the aggregation-participation link can be set separately
Syntax and Semantics Consistency Checking

Trying to make a process part of an object results:

PC consists of Box, Keyboard, Mouse and Monitor.

An object and a process cannot be connected with an aggregation-participation relation.
OPM Procedural Links

Links between a process and the object it transforms or a state of that object

- Agent link
- Instrument link
- Consumption/result link
- Effect link
- Event links
- Condition link
- Exception link
- Invocation link
Two Procedural Link Types

- Enabling Links
  - Agent link
  - Instrument link
- Transforming Links
  - result/consumption/input/output link
  - Effect link… more
Enabling Links

- Link objects that enable the process but are not transformed by it
  - Agent link
    Links a human or a group of humans that trigger the process or participate in it
  - Instrument link
    Links a non-human object that the process requires in order for it to occur or execute
Agent Link

**Diagram:**
- **Builder** connects to **Construction**.
- **Construction** connects to **Skyscraper**.
- **Constructor** connects to **Building**.
- **Building** connects to another **Building**.

**Textual Descriptions:**
- **Constructor** handles **Building**.
- **Building** yields **Building Object**.
- **Builder** handles **Construction Process**.
- **Construction Process** yields **Skyscraper**.
House is physical.
Constructor handles Building.
Building requires Plan.
Building yields House.
Procedural Links

- Enabling Links
  - Agent link
  - Instrument link

- Transforming Links
  - result/consumption/input/output link
  - Effect link
Creating a new object: Result link

Account Opening yields Account.
Consuming an existing object: Consumption link
Changing object state

Validating affects Credit Card from expired to valid.
Activating affects Account from frozen to active.
Expiring affects Credit Card from valid to expired.
State-specified object generation

File can be read-write or read only.
File Creating yields read-write File.
States can be initial or final

Account can be suspended, ultimately closed or initially open.
Suspending affects Account from open to suspended.
Opening yields open Account.
Closing affects Account from suspended to closed.
Object Essence

Can be informatical (default) or physical

Object-Process Diagram (OPD)

Object-Process Language (OPL) sentence(s)
Processes transform objects

- Three transformation options:
  - Change an object state
  - Create (generate) a new object
  - Consume (destroy) an existing object
Enabling Links: Agent and Instrument

House is physical.
Constructor handles Building.
Building requires Plan.
Building yields House.
Adding physical Instruments

- House is physical.
- Crane is physical.
- Concrete Mixer is physical.
- Constructor handles Building.
- Building requires Crane, Concrete Mixer and Plan.
- Building yields House.
Systemic vs. environmental objects

- Architect is environmental and physical.
- Constructor is physical.
- House is physical.
- Crane is physical.
- Concrete Mixer is physical.
- Plan is environmental.
- Architect and Constructor handles Building.
- Building requires Crane, Concrete Mixer and Plan.
Raw Material is consumed, House created

Concrete Mixer is physical.
Plan is environmental.
Architect and Constructor handles Building.
Building requires Crane, Concrete Mixer and Plan.
Building yields House.
Building consumes Raw Material.
Complexity Management in OPM

Three refinement/abstraction mechanisms:

- In-zooming/out-zooming
- Unfolding/folding
- State expression/state suppression
In-zooming

Exposing the details inside a complex process
In-zooming

A new OPD is created with Building enlarged.
In-zooming

The sub-processes of the Building process are depicted inside the in-zoomed process ellipse.
In-zooming is recursive
House, which is physical, consists of Foundations. Foundations is physical.
Steel is a Raw Material.
Concrete, which is physical, is a Raw Material.
Architect is environmental and physical.
Raw Material is physical.
Concrete Mixer is physical.
Plan is environmental.
Air Compressor is physical.
Constructor is physical.
Steel Frame Forming, which is physical, requires Plan.
Steel Frame Forming consumes Steel.
Digging, which is physical, requires Air Compressor.
Digging yields Holes.
Concrete Mixing & Pouring, which is physical, requires Certificate and Concrete Mixer.
Concrete Mixing & Pouring consumes Holes and Concrete.
Constructor handles Foundation Constructing.
Foundation Constructing yields Foundations of House.
Architect handles Frame Inspecting.
Frame Inspecting yields Certificate.
Simulation by animation

**Raw Material** is physical.

**Constructor** is physical.

**Architect**, which is environmental and physical, is creator of **Plan**.
Simulation setup
Animation parameter setup

![Animation Settings](attachment:image)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step Duration</td>
<td>1000 msec</td>
</tr>
<tr>
<td>Process Duration</td>
<td>Fixed 3 steps</td>
</tr>
<tr>
<td></td>
<td>Random 5 - 15 steps</td>
</tr>
<tr>
<td>Reaction Time</td>
<td>Fixed 1 steps</td>
</tr>
<tr>
<td></td>
<td>Random 5 - 15 steps</td>
</tr>
<tr>
<td>Default Object Instances</td>
<td>One</td>
</tr>
<tr>
<td></td>
<td>Many</td>
</tr>
<tr>
<td>Use Automatic Initiation</td>
<td>Enabled</td>
</tr>
<tr>
<td>Automatic Move Between OPD</td>
<td>Disabled</td>
</tr>
<tr>
<td>Animation Mode</td>
<td>Continuous</td>
</tr>
<tr>
<td></td>
<td>Step by Step</td>
</tr>
<tr>
<td>Random State Selection</td>
<td>Enabled</td>
</tr>
</tbody>
</table>

[![Save Default Close](attachment:image)]
Green objects already exist
Building started
Building in-zoomed
Foundation Constructing in-zoomed
Foundation Constructing in-zoomed
Certificate is being created
Foundations are being created, Holes and Concrete consumed
Foundations are ready, Steel and Concrete consumed
Walls are being constructed, Nails and Lumber consumed
Roof is being constructed, Tiles consumed
House ready, Raw Material gone
Downloading OPCAT 2: www.ObjectProcess.org

OPM - Official Web Site... - Microsoft Internet Explorer

OPM - Downloads

If you wish to join our member list to get news and updates please press here.

OPCAT 2.09 - Object-Process CASE Tool Version 2.09

Version: OPCAT 2.09
Released: 16/02/2003
File Size: 4.98 MB (zipped)

Description:

This version requires Java SDK 1.4.

This version contains the animation module, as well as a UML generator, a document generator, and an OPL to ODF converter.

OPCAT 2.07 - Object-Process CASE Tool Version 2.07

Version: OPCAT 2.07
Released: 2/2007
File Size:

Description: This version contains the animation module and runs in Java SDK 1.4 and 1.5.

The developers are not responsible for any direct or indirect damage as a result of using the software.
Translating to UML Diagrams

- Use Case Diagram
- Sequence Diagram
- Statechart
- Class Diagram
- Deployment Diagram
- Activity Diagram
Example: Defining a JAVA Condition

Translation to: java
Operation Number: 0
Condition:

**Complex Condition**

Action:
insertAtLocation
path
file
location

Translation:
```xml
<Method name="getthe$$subjectThingName$$">
  public $$subjectThingName$$ getthe$$subjectThingName$$() {
    return the$$subjectThingName$$;
  }
</Method>
<Method name="setthe$$subjectThingName$$">
  public void setthe$$subjectThingName$$ ($$subjectThingName$$ $new$$subjectThingName$$) {
    the$$subjectThingName$$ = new$$subjectThingName$$;
  }
</Method>
```
// File OrderPayingAndSupplying.java representing the complex process OrderPayingAndSupplying

package OrderSystem;

import opmTypes.*;

public class OrderPayingAndSupplying extends opmProcess {
    public OrderPayingAndSupplying () {
    }
    public boolean preConditionHolds (Boolean theInventoryEmpty) {
        boolean check = true;
        if (! ( (theInventoryEmpty.booleanValue()==false) || (theInventoryEmpty.booleanValue()==true)))
            check=false;
        return check;
    }
    public void run (Boolean theInventoryEmpty, Order theOrder, Product theProduct,
                    Receipt theReceipt) {
        if (preConditionHolds (theInventoryEmpty)) {
            // Effect theOrder
            // Effect theProduct
            // Yield theReceipt
            theReceipt = new Receipt();
            OrderPaying theOrderPaying = new OrderPaying();
            theOrderPaying.run(theOrder, theReceipt);
            OrderSupplying theOrderSupplying = new OrderSupplying();
            theOrderSupplying.run(theProduct, theOrder);
        }
    }
}
ViSWeb – The Visual Semantic Web

Objectives:

- Unifying human and machine knowledge representations with Object-Process Methodology (OPM)
- Enhancing the current Semantic Web technology
- Representing knowledge over the Web in a unified way that caters to human perceptions while also being machine-processable.
The Human-Machine Language Orientation Dilemma

Semantic Web major assumption:

- Humans and machines must each use a different format of knowledge representation.

- **OWL Introduction:** “...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.*” – NOT TRUE!
Graphic Knowledge Representations: Concept Maps

Spider

Flowchart

Hierarchy

System

Concept map of concept map (Adapted from http://classes.aces.uiuc.edu/ACES100/Mind/CMap.html)
Graphic Knowledge Representations: Semantic Network

Live Young \rightarrow \text{gives birth to} \rightarrow \text{Mammal} \rightarrow \text{Whale} \rightarrow \text{Blue Whale} \rightarrow \text{is the largest animal}

\text{Largest Animal} \rightarrow \text{is a} \rightarrow \text{Whale}

\text{Blue Whale} \rightarrow \text{is a} \rightarrow \text{Mammal}

\text{Mammal} \rightarrow \text{is a} \rightarrow \text{Animal}

\text{Animal} \rightarrow \text{is a} \rightarrow \text{Live Young}

\text{Martin} \rightarrow \text{lives in} \rightarrow \text{Pacific Ocean}
Graphic Knowledge Representations: Conceptual Graphs

CG Display Form (DF) for "John is going to Boston by bus."

[Go]-
(Agent)->[Person: John]
(Dest)->[City: Boston]
(Inst)->[Bus].

CG Linear Form (LF) for "John is going to Boston by bus."
Graphic Knowledge Representations: The OPM equivalent

The Person John exhibits the Location City.
City is Boston.
John handles Going.
Going requires Bus.
Going changes City to Boston.
Graphic Knowledge Representations: Conceptual Graphs: Context

CG Display Form (DF) for "Tom believes that Mary wants to marry a sailor".

[Person: Tom]←(Expr)→[Believe]→(Thme)←
[Proposition: [Person: Mary *x] ←(Expr)←[Want]→(Thme)←
[Situation: [?x] ←(Agnt) ←[Marry] →(Thme) →[Sailor]].

CG Linear Form (LF) for "Tom believes that Mary wants to marry a sailor".
Graphic Knowledge Representations: The OPM equivalent

OPD:
Graphic Knowledge Representations: RDF

"http://www.w3.org/Home/Lassila has creator Ora Lassila",
and in general

"<subject> HAS <predicate> <object>".
RDF – more realistic with RDF/XML Validation Service
http://www.w3.org/RDF/Validator/

<?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>
ViSWeb: An OPM-Based Visual Semantic Web Spec Alternative

Ora Lasilla is the creator of WWW.w3.org/Home/Lassila.
The ViSWeb Schema: Adding Class Information

Person is the creator of URI.

Person is the creator of URI.
Instantiating the ViSWeb Schema

Person is the creator of URI.
WWW.w3.org/Home/Lassila is an instance of URI.
Ora Lasilla is an instance of Person.
The Person Ora Lasilla is the creator of the URI WWW.w3.org/Home/Lassila.
The namespace **Semantic Web** is at URL [WWW.SemanticWeb.org/definitions](http://WWW.SemanticWeb.org/definitions). The namespace **Documents** is at URL [WWW.Documents.org/definitions](http://WWW.Documents.org/definitions).

(Namespace declaration sentences)

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

(Class definition sentences)

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

(Relation definition sentence)
The default namespace **Semantic Web** is at [WWW.SemanticWeb.org/definitions](http://WWW.SemanticWeb.org/definitions). The namespace **Documents** is at [WWW.Documents.org/definitions](http://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](http://WWW.w3.org/Home/Lassila).
Adding Attributes

An identified property with structured value
Adding Attributes

The corresponding OPD
A better representation

The default namespace **Semantic Web** is at [WWW.SemanticWeb.org/definitions](http://WWW.SemanticWeb.org/definitions).

The **Person Ora Lasilla** is the creator of the **Document** [WWW.w3.org/Home/Lassila](http://WWW.w3.org/Home/Lassila).

The **Person Ora Lasilla** exhibits the **Employee ID** [WWW.w3.org/staffid/85740](http://WWW.w3.org/staffid/85740) and the **Email** Lasilla@w3.org.
A final RDF and OPM example
A final RDF and OPM example

Namespace: rdfs
Http://www.w3.org/2000/01/rdf-schema#

Namespace: eg
Http://www.eg.org/egSpec#

Resource

Work

Agent

Document
Http://.../proposal/
is the author of

Person

Title
Information Management - A Proposal

Author
Tim Berners Lee
A final RDF and OPM example

The default namespace rdfs is at WWW.w3.org/2000/01/rdf-schema#.

The namespace eg is at WWW.eg.org/egSpecs#.

The namespace eg defines Work and Agent.

Work and Agent are Resources.

Document is a Work.

Person is an Agent.

Author is a Person.

Document exhibits Author and Title.

The Document Http://.../Proposal exhibits the Author Tim Berners Lee and the Title Information Management – A Proposal.
Advantages of the VisWeb Paradigm

- **Graphic-text knowledge representation:**
  - The powerful graphic-text bimodal representation of OPM is extended to the Visual Semantic Web paradigm.
  - Rather than mentally parsing cryptic XML scripts, knowledge is presented to the user in a subset of natural language as well as diagrammatically.
  - Puts to work the "two sides of the human brain," the visual and the lingual.
Advantages of the VisWeb Paradigm

- **Visual navigability:**
  - Diagrammatic display enables users to surf and navigate the Web in a visual way in search for knowledge.
  - Objects, processes, classes and links can be hyperlinked to pertinent Web sites, which themselves may contain VSW or any other multimedia knowledge representations.
Advantages of the VisWeb Paradigm

- **Semantic sentence interpretation:**
  - 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.
  - VisWeb is based on a sound ontology of *objects* with *states* and *processes*:
Advantages of the VisWeb Paradigm

- **Semantic sentence interpretation (cont.):**
  - VisWeb is based on a sound ontology of *objects* with *states* and *processes*:
  - Objects are things that exist, (possibly at some state)
  - Processes are things that happen to objects and transform them:
    - create or destroy them, or
    - change their state
  - Based on this ontology, sentences can be interpreted semantically rather than syntactically.
Advantages of the VisWeb Paradigm

- **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.
  - A major part of the knowledge about a system is functional (what is its purpose) and dynamic (how it operates).
  - 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.
Advantages of the VisWeb Paradigm

- **Complexity management:**
  - A major problem in real-life systems is their complexity due to the sheer amount of knowledge details.
  - 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.
Summary

- The Visual Semantic Web (VisWeb) paradigm proposes to unify human and machine representations of knowledge.
- The foundation for this unification is OPM.
- OPM advocates the integration of a system's structure and behavior is a single, graphic and textual model.
- Like OPM, the VisWeb model enables the representation of static and dynamic knowledge.
Summary 2

- VisWeb uses 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:
  - graphic-text knowledge representation,
  - visual navigability,
  - semantic sentence interpretation,
  - specification of system dynamics, and
  - complexity management.
- As noted in [7], "It is also important to understand that this XML syntax is only one possible syntax for"
Summary 3

- As noted in W3C RDF Documentation
- "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, the OPM-based approach to representing the Semantic Web on top of the RDF data model, is expressed
  - graphically, using OPDs, and
  - textually in Object-Process Language
Future Work – Theoretical

- Proceed in both the theoretical and practical paths.
- The theory will focus on extending the idea behind the VisWeb paradigm to other knowledge and system representation aspects.
- VisWeb should be able to also handle procedural, dynamic behavioral aspects, as well as functional ones.
Future Work – Practical

- The practical work will augment the current capabilities of OPCAT so it can
  - model the various VisWeb requirements
  - provide the services of bi-directional RDF-VSW compilation.

- Design and build a Web crawler which will automatically generate VSW representations of knowledge stored in Web pages.

- Accomplishing even some of these goals will greatly benefit the huge World Wide Web user community.