Modeling Semantic Web Services with OPM/S – A Human and Machine-Interpretable Language

Dov Dori **Eran Toch** **Iris Reinhartz-Berger**

Technion – Israel Institute of Technology
Faculty of Industrial Engineering and Management

MIS Department
University of Haifa

3rd International Workshop on Web Dynamics
WWW2004

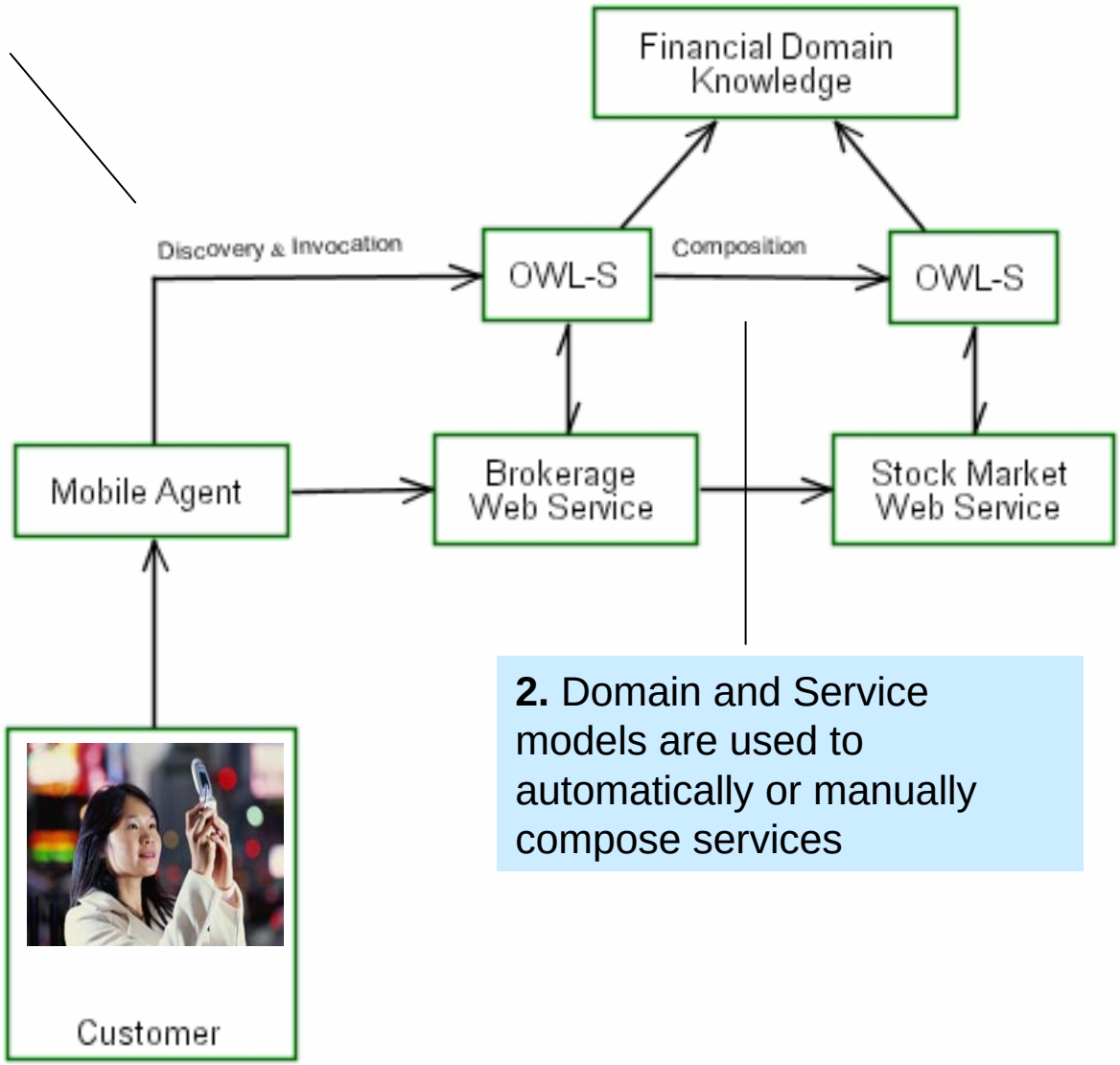
Agenda

1. Background and Challenges
2. OPM/S Modeling Framework
3. Dynamic Modeling
4. Conclusions and Future Research

The Semantic Web Services Vision

1. The customer's agent automatically locates and invokes the brokerage firm's Web service

3. Developers can use domain knowledge and service models (OWL-S) to enhance the engineering process by reusing existing services and validating artifacts



2. Domain and Service models are used to automatically or manually compose services

Ontology for Web Services (OWL-S)

- OWL-S is an upper ontology for a semantic description of Web services.
  - E.g. an input message to the service can be typed as the concept “Customer”, rather than just a String.
  - The words “customer” and “client” may have the same semantics in a specific domain, but they are not syntactically equivalent.
- OWL-S can be used for:
  - Automatic Discovery and Invocation.
  - Composition and interoperation.
  - Enhanced specification and development.
  - More…

OWL-S – cont'd

- Based on the Web Ontology Language (OWL) – a W3C Recommendation.
- OWL-S describes the service by using three sub-ontologies:
  - Profile model: What it does (inputs, outputs, preconditions…)
  - Process model: How it works
  - Grounding model: Grounding to an invocation method (e.g., WSDL)

Challenges of OWL-S Modeling

- Semantic Web Services did not cross the boundary between Industry and Academia.
- Why?
  1. OWL-S specifications are powerful, but complex to model and understand.
  2. Modeling requires knowledge in both OWL-S and Domain descriptions (OWL, RDFS, DAML+OIL…).
  3. Dynamic nature of OWL-S Models (next slide…).
  4. More issues…

Challenges of Dynamic Models

- Semantic Web services rely on external domain knowledge descriptions.
  - Domain knowledge descriptions form the basis of the definition of the service elements.
  - Changes in the domain knowledge can change the meaning of the elements.
- Modeling composite services that include external OWL-S models
  - External OWL-S models can change dynamically and without warning, violating the functionality of the model.

Agenda

1. Background and Challenges
2. OPM/S Modeling Framework
3. Dynamic Modeling
4. Conclusions and Future Research

OPM/S: OWL-S Modeling

- **Objective:** Streamline OWL-S Modeling
- Wrapping OWL-S with a higher-level modeling language and framework.
- Based on Object-Process Methodology (**OPM**):
  - Supported by the OPM Case Tool (OPCAT).
  - A single language for both the model and the domain knowledge.
- Provides a framework for dynamic modeling through meta-libraries and transparent reuse.

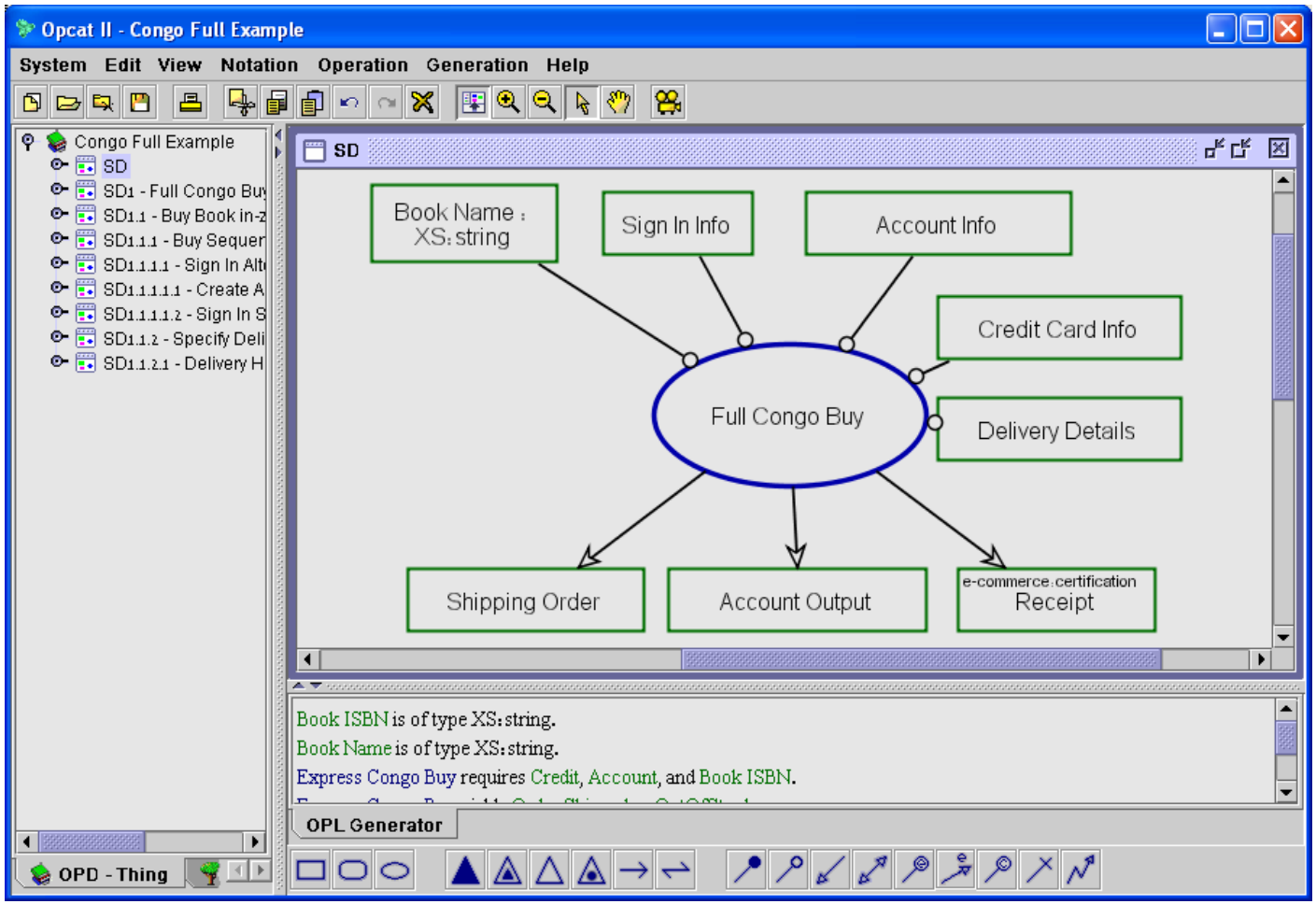
What is Object-Process Methodology?

- A comprehensive systems modeling, engineering, and lifecycle support paradigm.
- Two major features:
  - Unification of function, structure and behavior in a **single** view.
  - Bi-modal expression of the model via intuitive yet formal **graphics** and equivalent **natural language.**

OPM Elements

- Entity types:
  - **Object**: A thing that exists for some time
  - **State**: A situation at which an object can be
  - **Process**: A thing that transforms at least one object
- Link types:
  - **Structural link**: A link denoting a persistent relation between things
  - **Procedural link**: A link denoting transformation of an object by a process, enabling of a process by an object, or triggering a process

OPCAT – OPM Case Tool



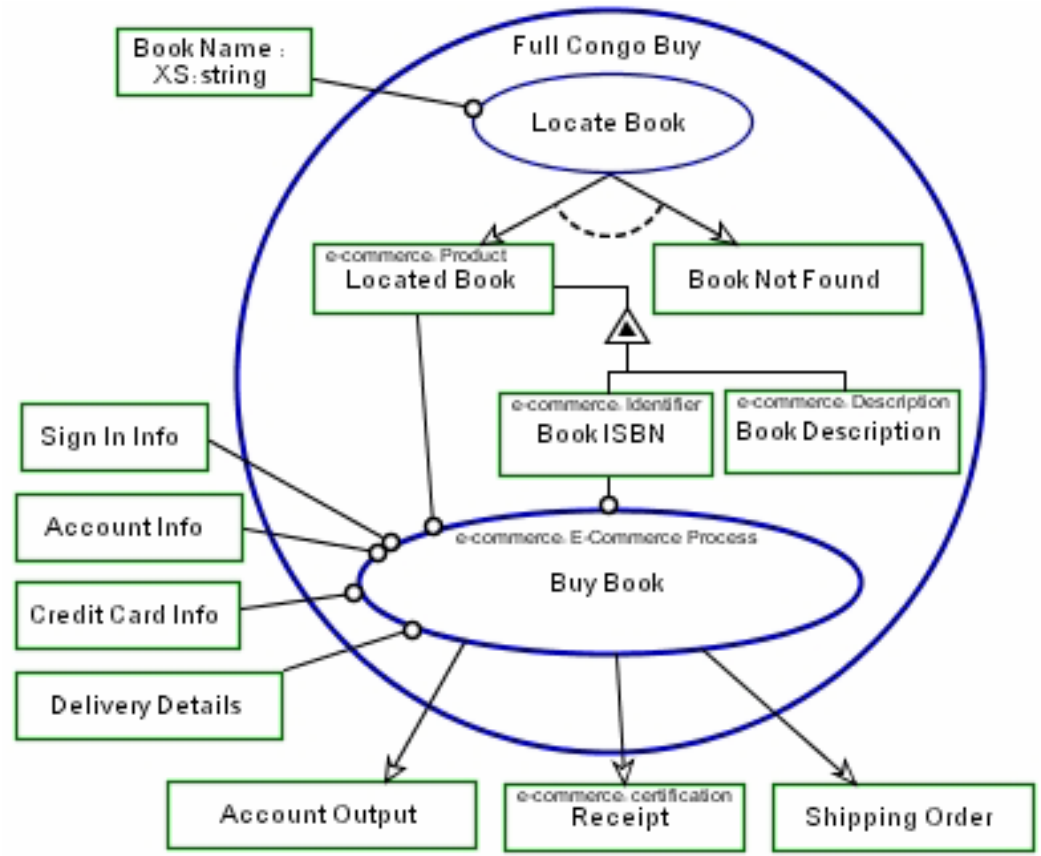
Download from: http://www.objectprocess.org

Why is OPM suitable for OWL-S Modeling?

- Unifies the process model, profile model and grounding model of OWL-S into a single frame of reference.
- Proven ease of use.
- Complexity management through scaling mechanisms.
- a visual-textual notation to increase comprehension.

OWL-S Model in OPM/S – An Example

- An OPM/S representation of "*Full Congo Buy*" (an example of an OWL-S Specification):



To the Meta-Library >>

An Example – cont'd

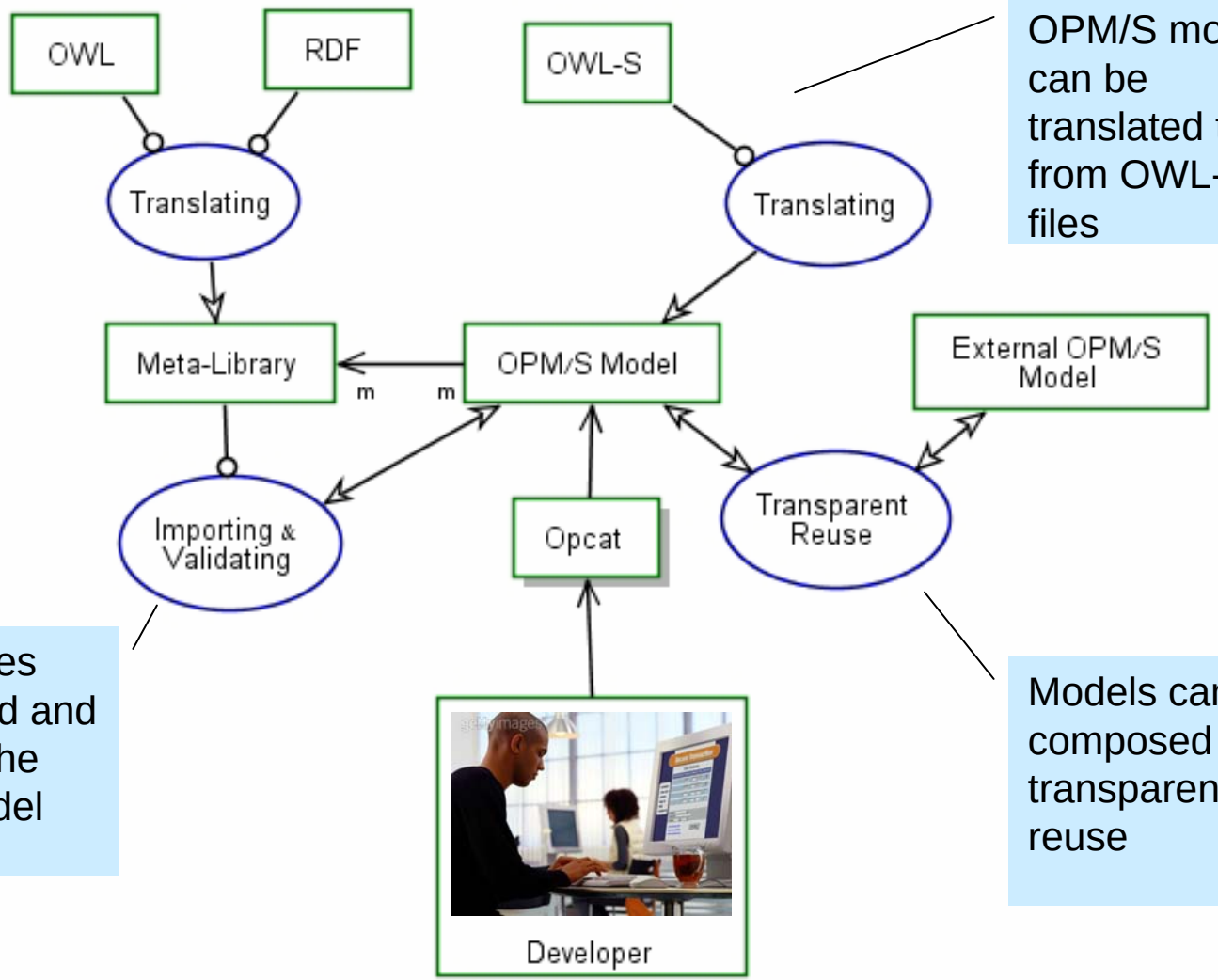
- OPL (Object-Process Language) View:

Book Name is of type XS:string**.**
Full Congo Buy zooms into **Locate Book** and **Buy Book**, as well as **Book Not Found** and **Located Book**.
Located Book plays the role of **Product** of **e-commerce.**
Located Book exhibits **Book ISBN** and **Book Description**
Book ISBN plays the role of **Identifier** of **e-commerce.**
Book Description plays the role of **Description e-commerce.**
Locate Book requires **Book Name.**
Locate Book yields either **Located Book** or **Book Not Found.**
Buy Book plays the role of **E-Commerce Process** of **e-commerce.**
Buy Book occurs if **Located Book** exists**.**
Buy Book requires **Book ISBN, Credit Card Info, Sign In Info, Account Info,** and **Delivery Details.**
Buy Book yields **Receipt, Shipping Order,** and **Account Output.**
Receipt plays the role of **Confirmation** of **e-commerce.**

Agenda

1. Background and Challenges
2. OPM/S Modeling Framework
3. Dynamic Modeling
4. Conclusions and Future Research

OPM/S Dynamic Framework Architecture



OWL and RDF can be translated into an OPM meta-library

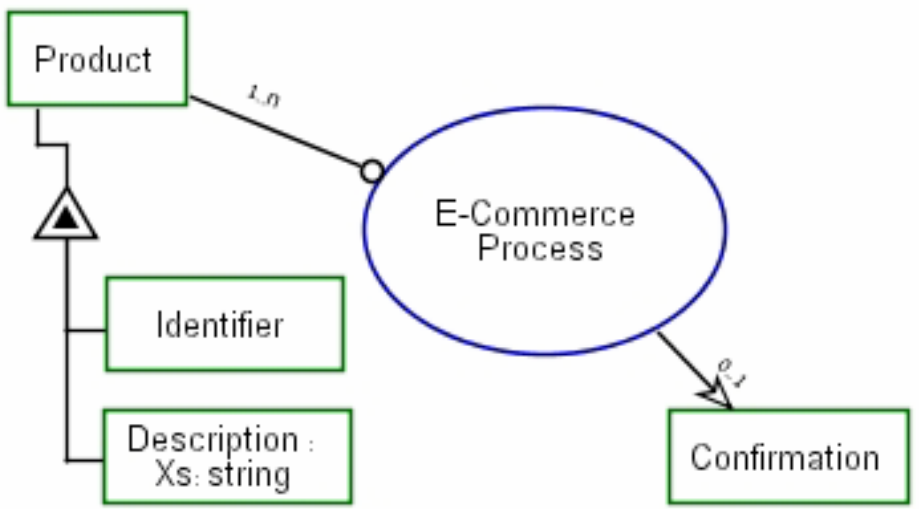
OPM/S models can be translated to and from OWL-S files

Meta-libraries are imported and utilized by the OPM/S model

Models can be composed using transparent reuse

Meta-Libraries

- Represent dynamic, distributed and usable domain knowledge.
- Define constraints to be validated on specific models.
- Can be translated from OWL or RDFS.
- A simple e-commerce meta-library:



<< to the model

Meta-Libraries – Dynamic Aspects

- Meta-Libraries are dynamically referenced through URLs by each model that utilize it.
- The references are refreshed each time a design session is initiated.
- The Web service model is validated against the library, insuring ongoing reliability of the library.

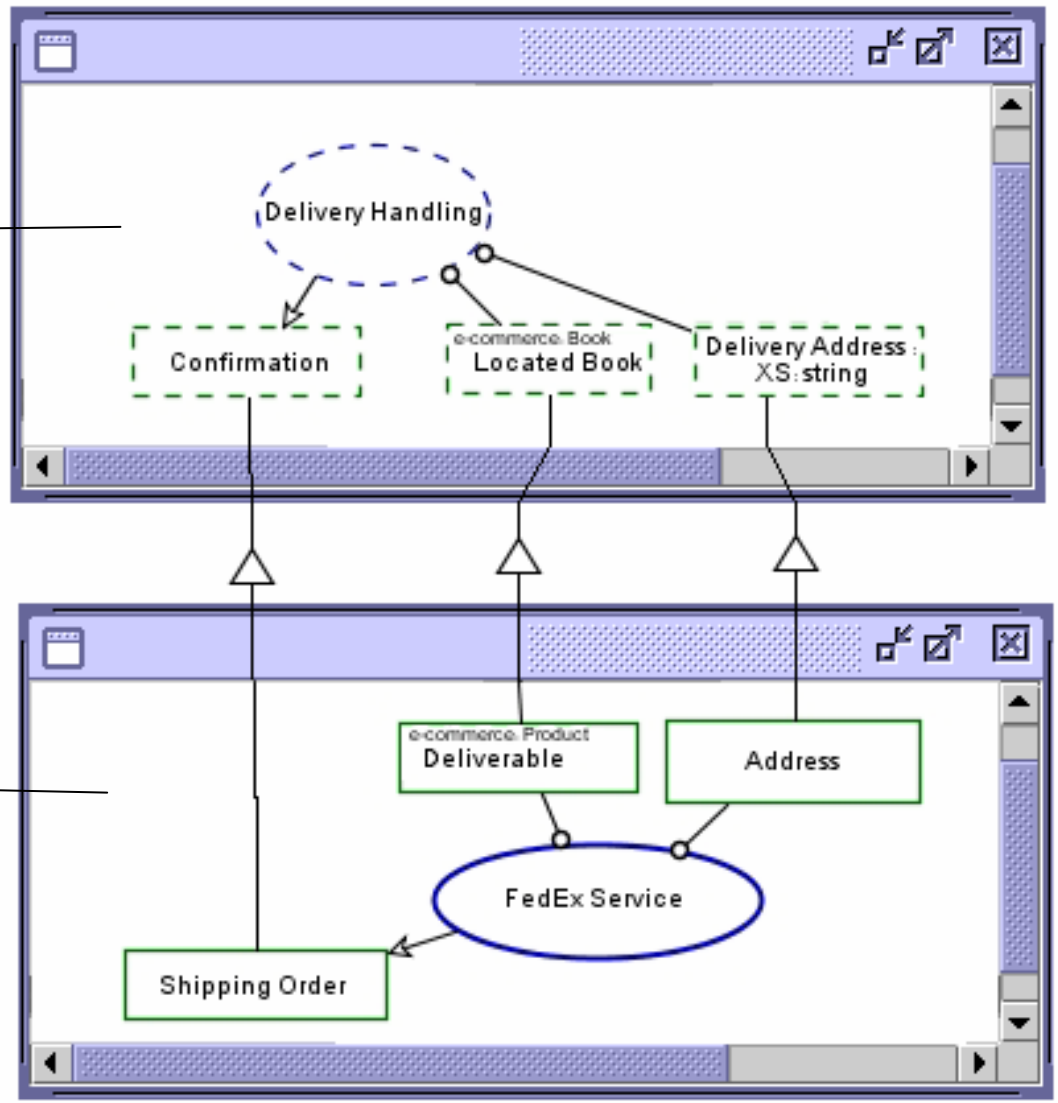
Transparent Reuse in Depth

- Enables ongoing modeling of interconnected Web services.
- The reused models are not black boxes, and reflect dynamic changes.
- A **supervision mechanism** handles dynamic changes in related models, and automatically suggests possible solutions, based on shared semantics.

Transparent Reuse – An Example

Parts of the local system are annotated as “environmental”, serving as a stub for the external service

The external system is visually linked to the local system by “is-a” relations



Transparent Reuse - Supervision

- A supervision mechanism is employed in order to detect changes in related models.
- Alternative substitutions for broken links are suggested by matching semantically related entities.
  - For example, *Deliverable* can be automatically linked to *Located Book* if a semantic connection exists between their roles: *e*-commerce:Book and e-commerce:Product.

Agenda

1. Background and Challenges
2. OPM/S Modeling Framework
3. Dynamic Modeling
4. Conclusions and Future Research

Conclusions

- Due to its dynamic and semantic nature, modeling of OWL-S specifications poses special challenges.
- Based on OPM, OPM/S aims to streamline modeling of OWL-S specifications.
- Meta-libraries and transparent reuse fit the dynamic nature of Web service modeling:
  - Using a distributed architecture, which is
  - Enhanced by supervision and validation mechanisms

Current Development State

Tool	Availability
OPCAT 2	✓
Meta-Libraries (+validation)	✓
Transparent Reuse	July 2004
OWL-S to OPM/S translation	October 2004
OWL and RDF translation to OPM	October 2004
Supervision Mechanism for Transparent Reuse	January 2005

Future Research

- Matching and discovery for semantic Web services, based on the OPM/S notation.
- Measuring Cognitive perception, readability and usability of models of semantic Web services.
- Using semantic meta-data to speed up modeling tasks.
- More…

Thank You

http://www.objectprocess.org