Using RDF to Derive Schema Mappings

Brian Matthews

Contents

- Semantic Web
  - SWAD-Europe
- Motivations
- Meaning in XML documents
- Automatic generation of RDF
- A more pragmatic approach.

CCLRC

- Council for the Central Laboratory of the Research Councils (CCLRC)
- Big Science
  - Synchrotron Radiation Sources
  - Lasers
  - Pulsed Neutron Source
- Large-scale IT demands: tera-scale data, computation
- Strong IT R&D programme
- BITD: Business and Information Technology Department

Semantic Web

- Current Web:
  - Web of information for humans
- Semantic Web:
  - Web of data for computers
- Why?
  - Automation, organisation, search, integration
- Enabling technologies:
  - **RDF**: Resource Description Framework
    - Data linking, graph semantics
  - **OWL**: Web Ontology Language
    - Description Logic semantics, inference

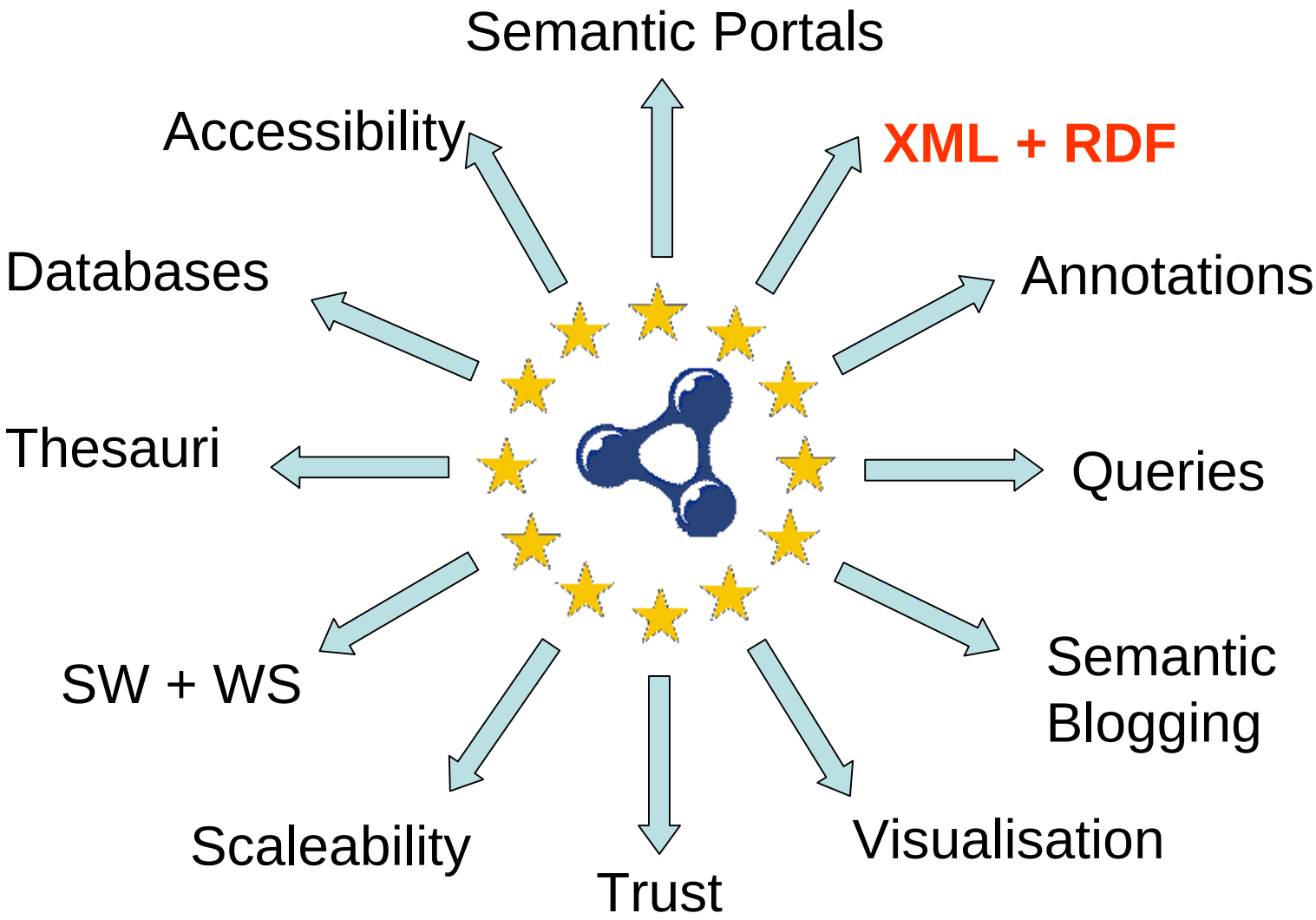
Semantic Web Advanced Development in Europe

- Purpose is to encourage the use of Semantic Web tools and techniques now:
  - By an outreach programme
  - By developing practical demonstrators
  - By providing tools and standards

- Partners:
  - Univ. of Bristol, W3C-ERCIM, CCLRC, HP Labs, Stilo

http://www.w3.org/2001/sw/Europe/

SWAD-Europe


Semantic Portals
Accessibility
XML + RDF
Databases
Annotations
Thesauri
Queries
SW + WS
Semantic Blogging
Scaleability
Trust
Visualisation

Motivation

- Lots of legacy XML in the world
- Can we bring it into the semantic web:
  - XML provides a syntactic representation
  - Defined using XSD/RNG/Schematron etc.
  - Has an implied semantics – the intent of the Schema developer
  - Can we bring this out and use it?
- Use current XML as a carrier of semantics
- Using Semantic Web as “glue” for systems integration.
  - ease the use of XML as a communication mechanism

Designing Web Systems

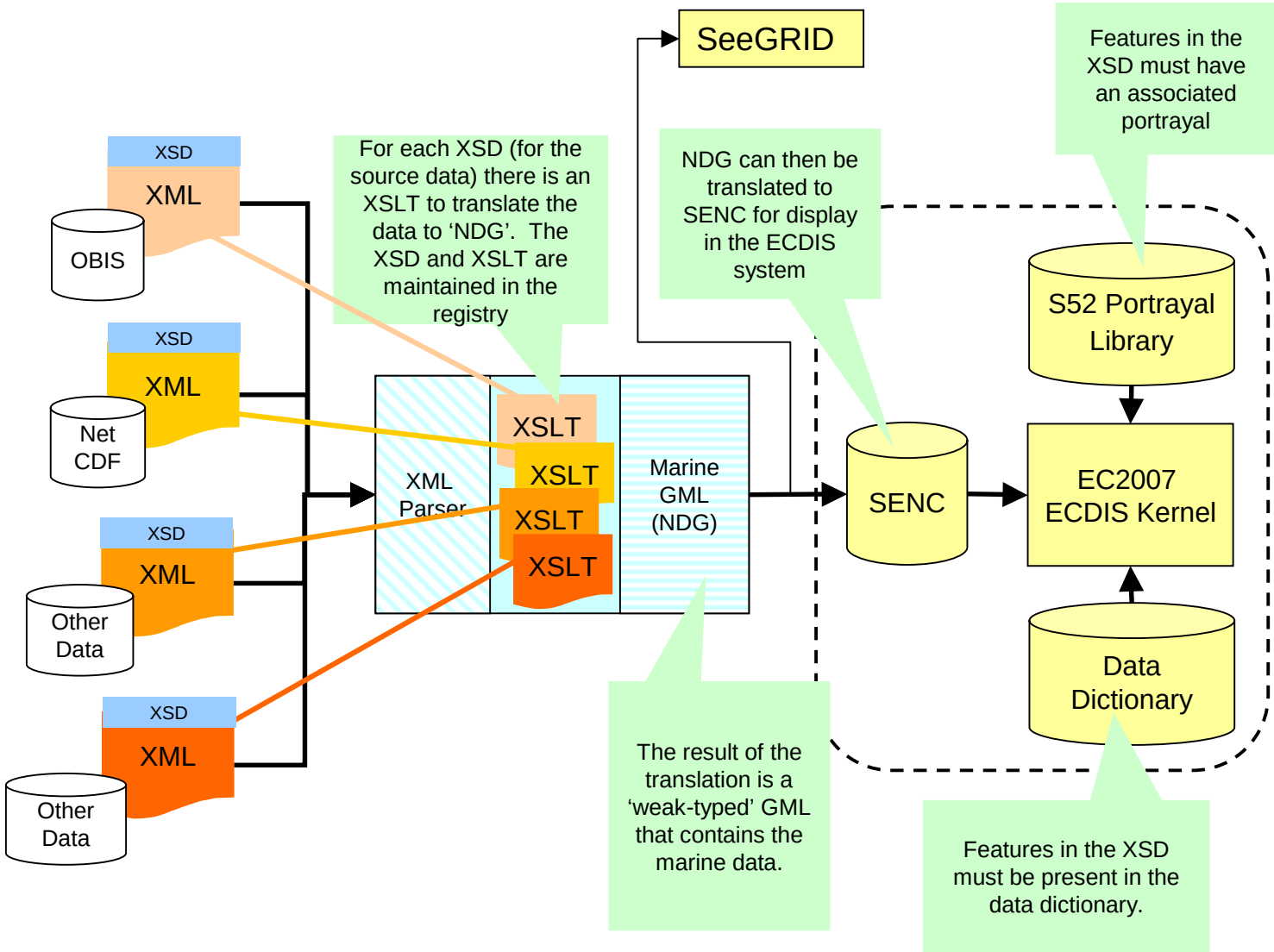
- Designing an XML base representation for use in "semantics aware" systems.
  - Using an ontology as part of the design process of generating data formats to capture the information.
- Ontology comes first,
  - Different XML formats for different purposes.
  - Derive XML schemas logically from the data model
  - Don't need the ontology at runtime.
- All the information in the schemas would be represented in the ontologies,
  - but not necessarily have all the information in the schemas that is in the ontology.
- In the software engineering design process
  - a different talk

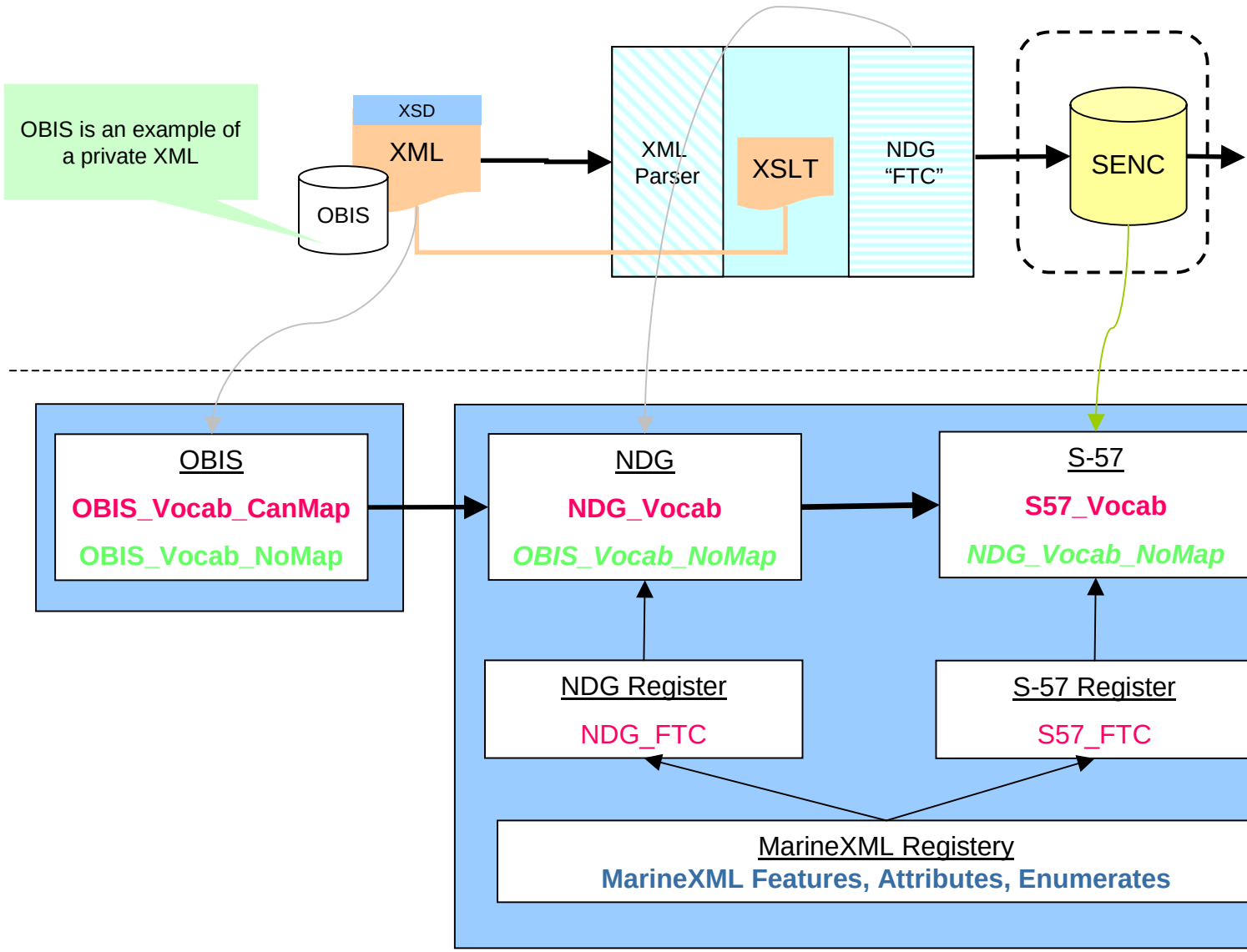
Data Exchange

- Need to input information from a different legacy system
  - Not been built on the same data model.
  - Different systems have been constructed independently and
  - Often by different organisations.
- The recipient would receive data conforming to a legacy XML Schema.
  - Need to convert this to semantically rich information.
  - RDF triples.
  - Need to produce a mapping from their XML data into your data model.
  - The mismatch could then be quite severe.
  - Some information could be ignored.
- Need a conversion script
  - extracts the RDF triples conforming to the information which is meaningful to the recipient.

Converting between XML formats

- Converting from one XML Schema format to another from a different user
  - for data exchange or combination is likely to be a common use. In this case,
  - the "semantic" stage in the process can be ignored at conversion stage
  - this analysis would have taken place solely in the derivation of the mapping,
- Control by a master ontology.
- Mappings between the master ontology and the two XML Schemas to control the derivation of an XSLT script.


SeeGRID
Features in the XSD must have an associated portrayal
For each XSD (for the source data) there is an XSLT to translate the data to 'NDG'. The XSD and XSLT are maintained in the registry
NDG can then be translated to SENC for display in the ECDIS system
XSD
XML
OBIS
XSD
XML
Net CDF
XSD
XML
Other Data
XSD
XML
Other Data
XML Parser
XSLT
XSLT
XSLT
XSLT
Marine GML (NDG)
SENC
S52 Portrayal Library
EC2007 ECDIS Kernel
Data Dictionary
The result of the translation is a 'weak-typed' GML that contains the marine data.
Features in the XSD must be present in the data dictionary.


OBIS is an example of a private XML
XSD
XML
OBIS
XML Parser
XSLT
NDG "FTC"
SENC
OBIS
OBIS_Vocab_CanMap
OBIS_Vocab_NoMap
NDG
NDG_Vocab
OBIS_Vocab_NoMap
S-57
S57_Vocab
NDG_Vocab_NoMap
NDG Register
NDG_FTC
S-57 Register
S57_FTC
MarineXML Registery
MarineXML Features, Attributes, Enumerates

A Simple Model of Meanings

- Broadly speaking, when we describe our universe of discourse, we make statements of three types:
  - Existence/Type statements "an object X of type T exists"; "there is a winegrower"; "there are two wines"
  - Attribute/Value statements "the colour of this wine is red"
  - Relationship statements "this wine is produced by that winegrower"
- Found in many modelling paradigms
  - from UML to Entity-Relationship diagrams.
- In RDF Schema the single concept 'property' covers both 'Attribute/Value' and 'Relationship'.
- OWL distinguishes between Datatype and Object properties.
- 'Structural' XML does not explicitly encode the information in this way.
  - However, we can see some patterns.

The Implied Semantics of XML Structures

- **Objects and Instances**
  - In general, objects are represented by XML elements:

```
<winegrower name="Chateau Verpriced" > </winegrower>
```

implies the existence of a winegrower object. Alternatively

```
<organisation orgtype = "winegrower" name="Chateau Verpriced" > </organisation >
```

or :

```
<organisation name="Chateau Verpriced" > <orgtype >winegrower</orgtype > </organisation >
```

  - Not every element corresponds necessarily to an object.

```
<winegrower > <name>Chateau Verpriced</name> </winegrower>
```

  - The relation between elements and objects may be context-dependent.

```
<winegrowers>
   <organisation name="Chateau Verpriced" >
   <organisation name="Chateau Verdrawn" >
</winegrowers> <winemerchants>
   <organisation name="Cheap+Cheerful" >
    <organisation name="Rough+Ready" >
</winemerchants >
```

  - So in general we can say something like

"An element with name E represents an object of type T “

where this may be further qualified by

- Context - a particular XPath within the document
- Subselection - dependency of a particular value.

The Implied Semantics of XML Structures

- **Attribute Values**
  - Object attribute values are often represented by the contents of XML attributes or subelements.

```
<wine> <name>Vielles Bottes</name > </wine>

<wine name = "Vielles Bottes" > </wine>
```

  - There may a level of conditionality,

```
<wine>
   <wine-prop prop-name = "name" prop-value = "Vielles Bottes" / >
  <wine-prop prop-name = "colour" prop-value = "noir" / >
</wine>
```

  Here the meaning of the 'prop-value' attribute depends on the contents of 'prop-name' attribute.

- **Relationships**
  - Relationships are represented in XML structures in various ways.

```
<winemerchant name = "Bristol Bottlers" >
   <wine> <name>Vielles Bottes</name> <colour>black</colour> </wine>
   <wine> <name>Weston's Finest</name> <colour>red</colour> </wine>
</winemerchant>
```

  Here the nesting establishes a relationship between the "Bristol Bottlers" winemerchant and the wines they sell.

  - alternatively

```
<wine>
 <name>Vielles Bottes</name>
<colour>black</colour>
<winemerchant name = "Bristol Bottlers" />
<winemerchant name = "Bath Brewers" />
</wine>
```

- **Cannot rely purely on syntax of the source document**
  - some higher-level interpretation is required.

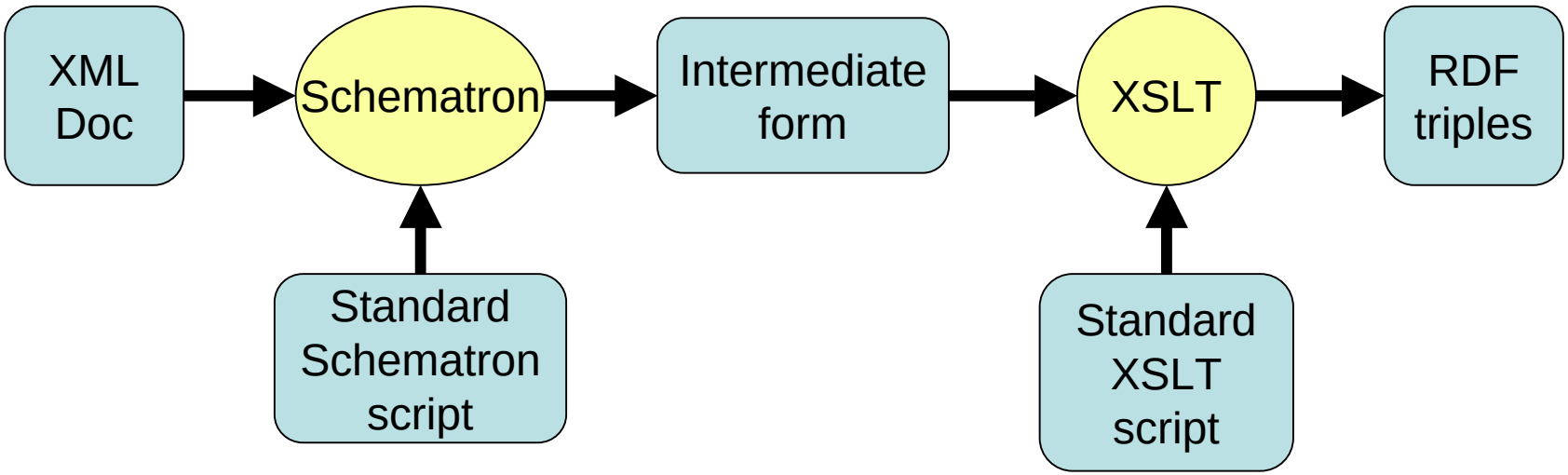
Normal Forms

- By writing the XML format in a constrained manner, the interpretation of the elements can be made systematic.
- e.g. Alternating Normal Form (Henry Thompson)
- Alternate elements representing objects and attributes/relations

```
<PurchaseOrder>
<orderDate>1999-10-20</orderDate>
 <shipTo>
  <Address>
   <country>US</country>
   <name>Alice Smith</name>
   <street>123 Maple Street</street>
   <city>Mill Valley</city>
   <state>CA</state>
   <zip>90952</zip>
  </Address>
 </shipTo>
 <item>
   <Item>
    <partNum>872-AA</partNum>
   <productName>Lawnmower</productName>
    <quantity>1</quantity>
    <USPrice>148.95</USPrice>
   </Item>
  </item>
</PurchaseOrder>
```

Automated approach

- If the XML document is in alternating normal form it is straightforward to convert to RDF triples.
- Translation by Stephen Buswell, Stilo:



A Pragmatic Approach

- Automated approach works, but is limited
- In general, for an arbitrary XML structure, the automated approach is impractical.
- A pragmatic approach:
  - hand-craft mappings between OWL ontologies and XML Schema
  - Use the mapping to construct translators between XML Documents and RDF

A Mapping Language

- A simple mapping language maps from OWL classes to SML Schema components:

```
Purchase_Order ------> purchaseOrder
```

- Need to add more context of the mapping of XPaths into the XML Schema:

```
Purchase_Order ------> xsd:element[@name="purchaseOrder"]

class(US_Address) ------> xsd:complexType [@name="USAddress]

X:class(Item) ------> xsd:element [@name=" items”]/xsd:sequence/xsd:element[@name=“item”]
```

- We may also need a *conditional* mapping rule.

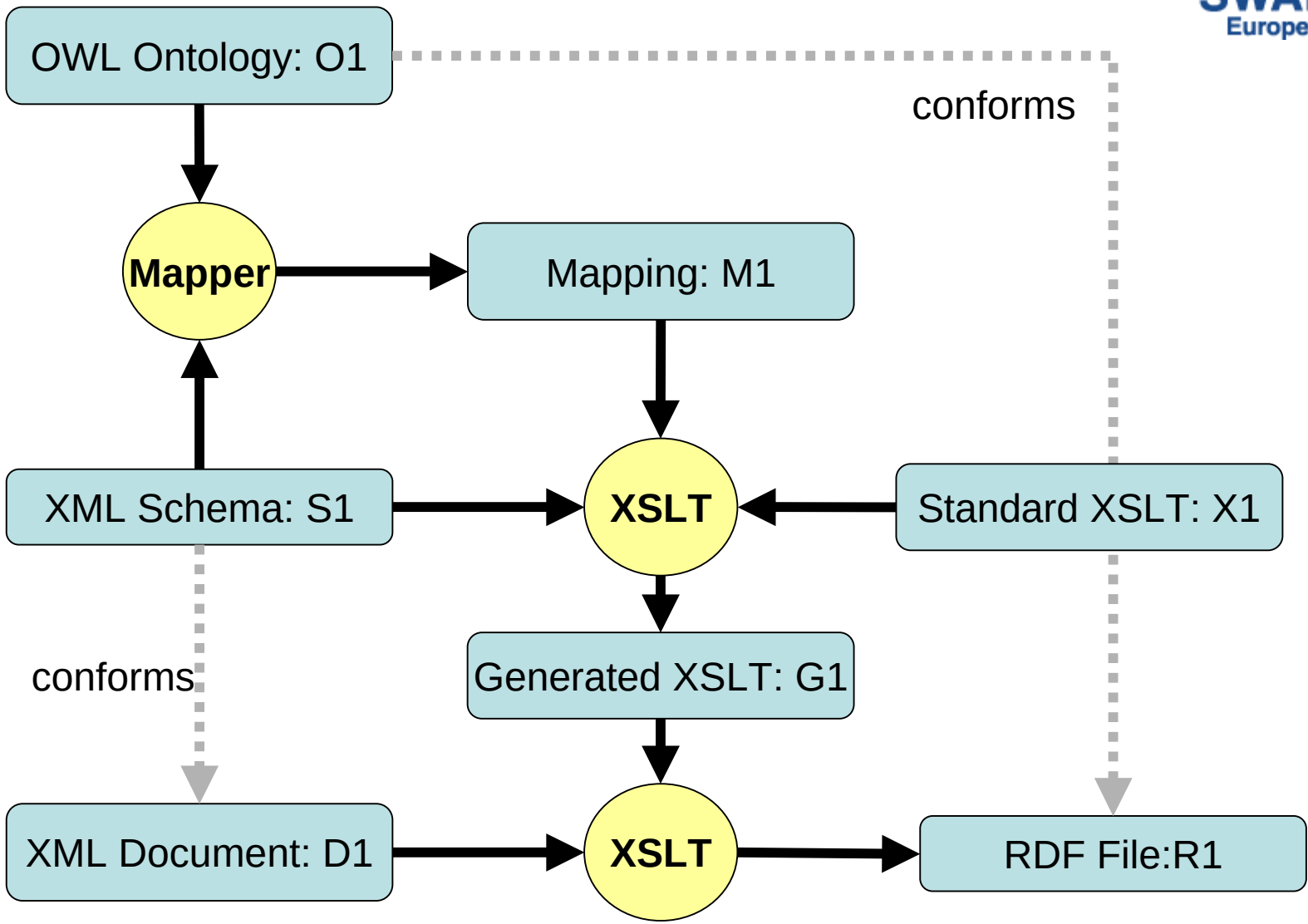
```
X.class(Address) ------> X.xsd:complexType[@name=“USAddress”] if X.class(US_Address)
```

- **Mapping the Properties**
  - It is necessary to include information on the domain and range instances of the property.

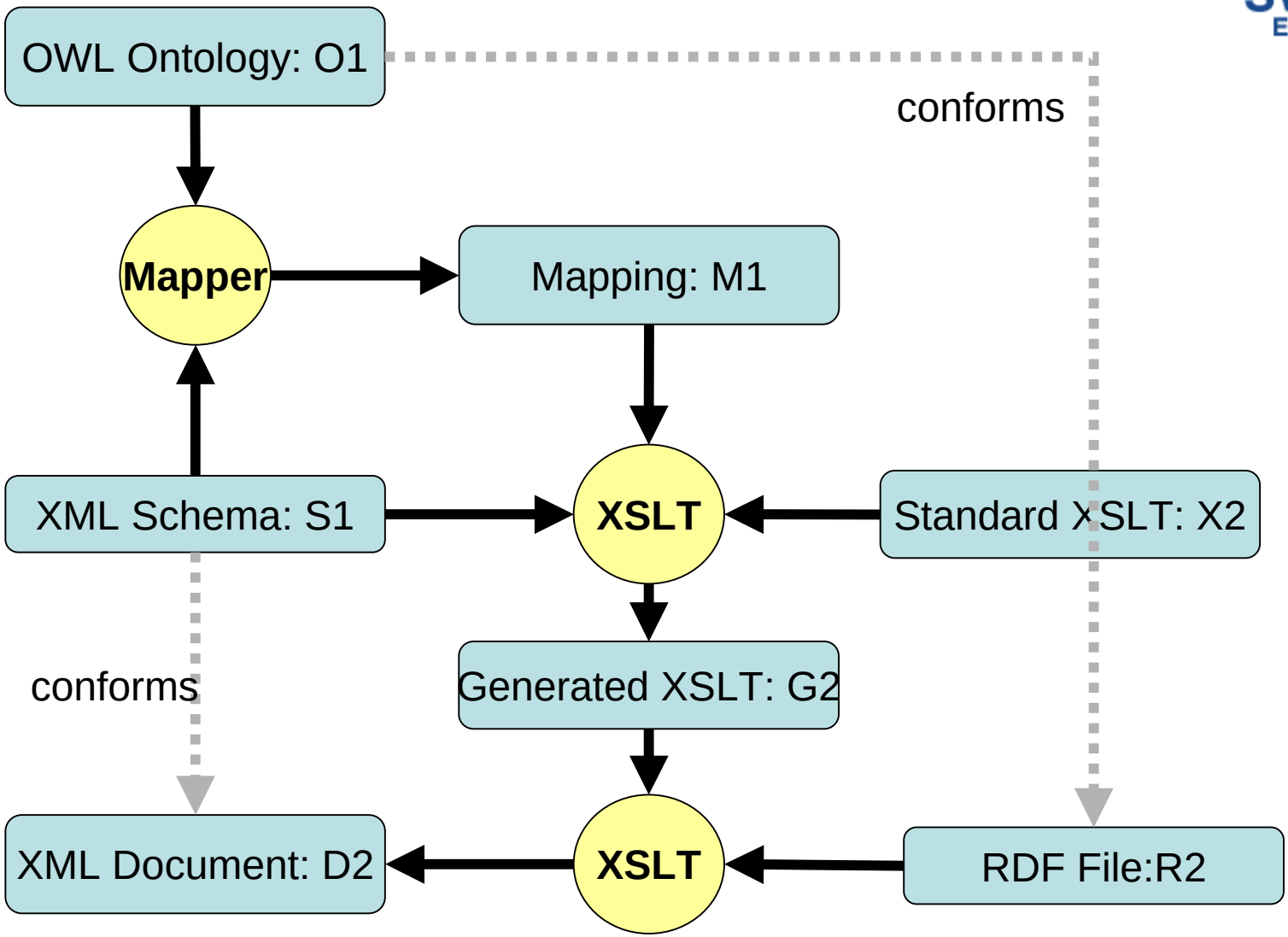
```
objectProperty(Billing) ------>
   xsd:complexType[@name=“PurchaseOrderType”]/xsd:sequence/xsd:element[@name=“billTo”]
Dom ../purchaseOrder
Rng ./*

objectProperty(Shipping) ------>
   xsd:complexType[@name=“PurchaseOrderType”]/xsd:sequence/xsd:element[@name=“shipsTo”]
Dom ../purchaseOrder
Rng ./*
```

- Embodied in an RDF Schema

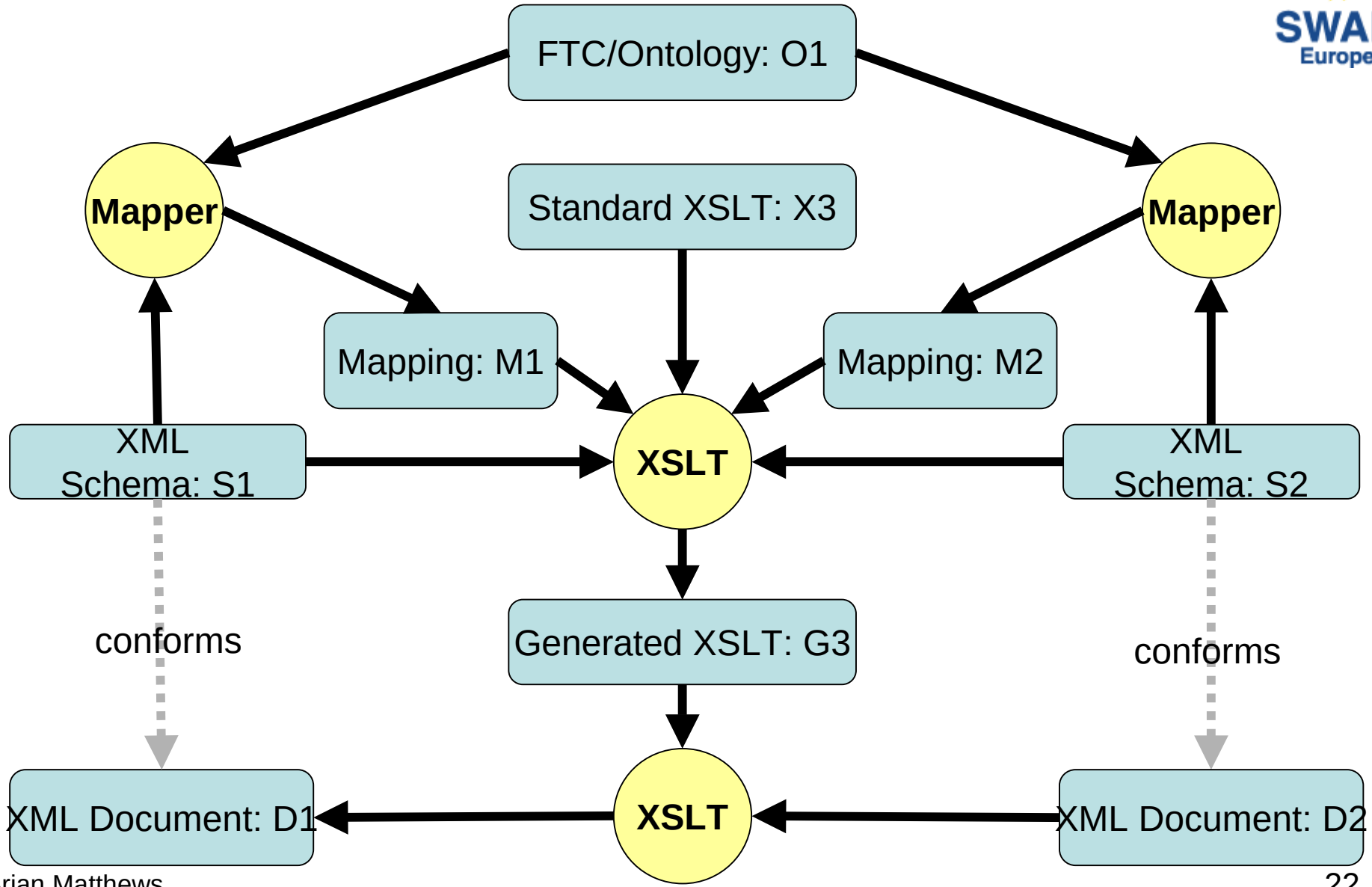
Using the Mapping 1



Using the Mapping 2

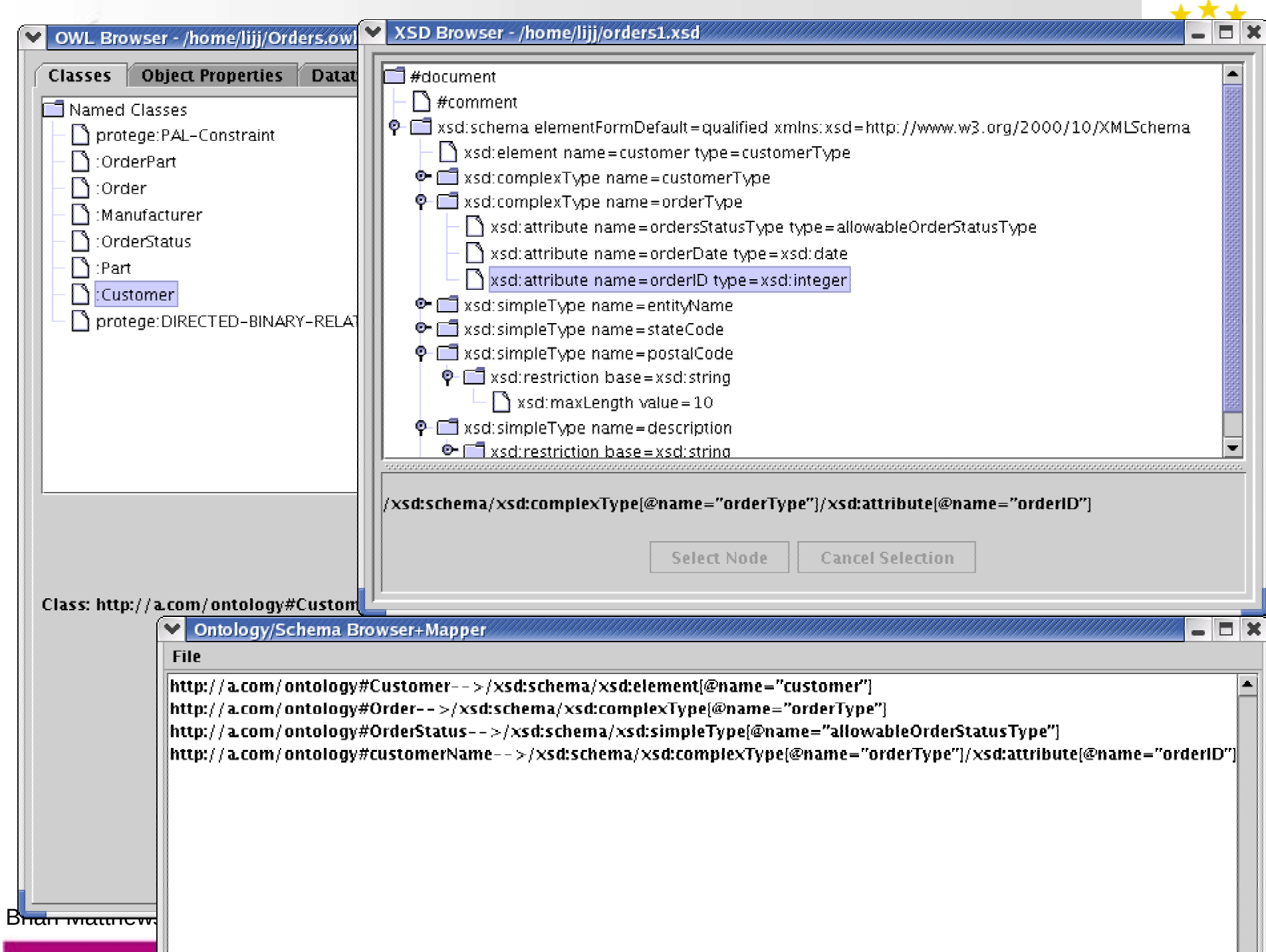


Using the Mapping 3



A Mapping Tool

- Developing mappings is a pretty tedious task
  - error prone
  - generating the right XPath
- Produced a tool to help support the generation of the mapping
  - Browse the Ontology and Schema
  - select and drag'n'drop to relates components
  - generate mapping


OWL Browser - /home/lijj/Orders.owl
Classes
Object Properties
Datat
Named Classes
protege:PAL-Constraint
:OrderPart
:Order
:Manufacturer
:OrderStatus
:Part
:Customer
protege:DIRECTED-BINARY-RELAT
XSD Browser - /home/lijj/orders1.xsd
#document
#comment
xsd:schema elementFormDefault=qualified xmlns:xsd=http://www.w3.org/2000/10/XMLSchema
xsd:element name=customer type=customerType
xsd:complexType name=customerType
xsd:complexType name=orderType
xsd:attribute name=ordersStatusType type=allowableOrderStatusType
xsd:attribute name=orderDate type=xsd:date
xsd:attribute name=orderID type=xsd:integer
xsd:simpleType name=entityName
xsd:simpleType name=stateCode
xsd:simpleType name=postalCode
xsd:restriction base=xsd:string
xsd:maxLength value=10
xsd:simpleType name=description
xsd:restriction base=xsd:string
/xsd:schema/xsd:complexType[@name="orderType"]/xsd:attribute[@name="orderID"]
Select Node
Cancel Selection
Class: http://a.com/ontology#Custom
Ontology/Schema Browser+Mapper
File
http://a.com/ontology#Customer-->/xsd:schema/xsd:element[@name="customer"]
http://a.com/ontology#Order-->/xsd:schema/xsd:complexType[@name="orderType"]
http://a.com/ontology#OrderStatus-->/xsd:schema/xsd:simpleType[@name="allowableOrderStatusType"]
http://a.com/ontology#customerName-->/xsd:schema/xsd:complexType[@name="orderType"]/xsd:attribute[@name="orderID"]

Conclusions

- Needs testing in practise:
  - using in MarineXML project
  - exchanging oceanographic data (navigational and scientific).
  - Automation?
  - At least heuristics, guidelines and documentation
- Part of the good software engineering process
- Using the Semantic Web can smooth the task of providing a framework for data exchange.
  - Help provide the glue in the design process
- XML and the Semantic Web are complementary.