Analysis & Design of XML Vocabularies with UML

XMLmodeling.com

David Carlson
Ontogenics Corp.
dcarlson@ontogenics.com
http://XMLmodeling.com

XMLmodeling.com

Agenda

- Conceptual analysis of XML vocabularies
- Designing XML schemas with UML
- UML Profile for XML schemas
- Supporting alternative XML schemas:
  - W3C XML Schema, RELAX NG, SOX, DTD, ...
- Case studies and lessons learned
  - FpML financial products schema
  - xCBL e-commerce vocabulary
  - XMI 2.0 schema reverse-engineered to UML
  - TREX schema grammar

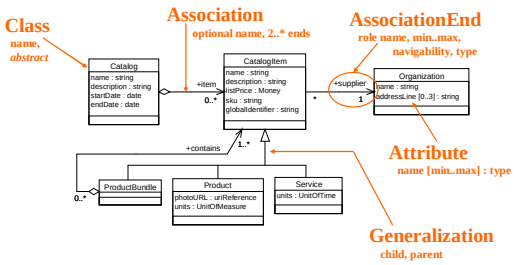
Mapping UML to XML

XMLmodeling.com

- Forward and Reverse engineering UML models to and from XML schemas
- Basic mapping rules

XMLmodeling.com

UML Class Diagram for CatML



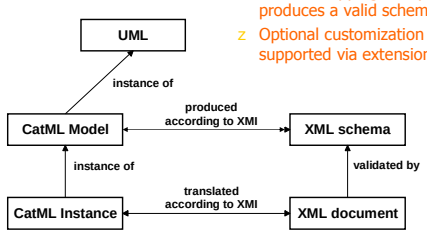
CatML as Portal Content Language

- This demo portal uses the Apache Jetspeed server
- Product Detail portlet is based on this simple CatML model
- Other portlets for price discounts and categories are from an extended model

Goals of Mapping

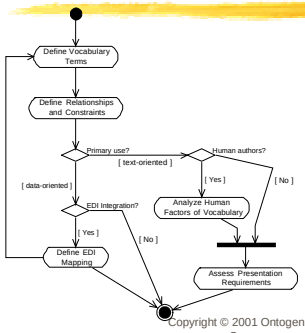
- Produce a usable XML schema from any UML class diagram, without use of extension stereotypes
- Customize schema specification by using UML profile extensions (required for most reverse-engineering)
- Start with XMI 1.1 specification as default, baseline production rules
- Possible to automate both schema generation and reverse-engineering
- UML packages should be mapped to XML namespaces and modular schema design

Mapping UML to Schemas & Instance



- Default mapping always produces a valid schema.
- Optional customization is supported via extensions

Forward Engineering Schemas



- Analysis of business vocabulary guides the development of new XML schemas.
- Requirements for data-oriented or text-oriented use determine the final design.
- This is a UML Activity diagram, often used for business process diagramming.

XMLmodeling.com

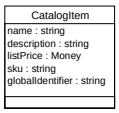
Reverse Engineering Schemas

- Integrate existing XML schema assets into UML software engineering tools and repositories
- Use the same mapping rules as forward engineering
- Support round-trip development process
- Unless the schema was written with strict adherence to XMI production rules, then UML modeling extensions are probably required
- Use reverse-engineered schemas a part of a larger new development project
  - Examples: UDDI, WSDL, XHTML, NewsML, FpML, xCBL, DocBook, xlang, TREX grammar

 9

XMLmodeling.com

UML Class and Attributes



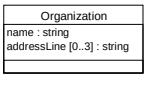

```
<xs:element name="CatalogItem" type="cml:CatalogItem"/>

<xs:complexType name="CatalogItem">
  <xs:all>
    <xs:element name="name" type="xs:string"/>
    <xs:element name="description" type="xs:string"/>
    <xs:element name="listPrice">
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="cml:Money"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="sku" type="xs:string"/>
    <xs:element name="globalIdentifier" type="xs:string"/>
  </xs:all>
</xs:complexType>
```

Allows unordered element content, but still enforces min/maxOccurs

 10

XMLmodeling.com

UML Attribute



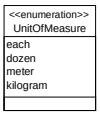

```
<xs:element name="Organization"
            type="cml:Organization"/>

<xs:complexType name="Organization">
  <xs:sequence>
    <xs:element name="name" type="xs:string"/>
    <xs:element name="addressLine" type="xs:string"
                minOccurs="0" maxOccurs="3" />
  </xs:sequence>
</xs:complexType>
```

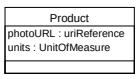
- All UML attributes are mapped to XML elements in these examples, but you can control this via UML extensions.
- UML attributes are [1..1] multiplicity by default (i.e. required)

 11

XMLmodeling.com

Enumerated Datatypes



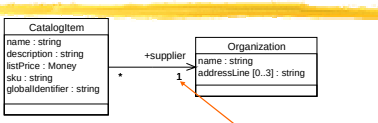



```
<xs:simpleType name="UnitOfMeasure">
  <xs:restriction base="xs:string">
    <xs:enumeration value="each"/>
    <xs:enumeration value="dozen"/>
    <xs:enumeration value="meter"/>
    <xs:enumeration value="kilogram"/>
  </xs:restriction>
</xs:simpleType>

<Product>
  <units>each</units>
</Product>

<Product units="each">
  ...
</Product>
```

 12

XMLmodeling.com

UML Association



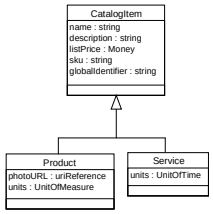

```
<xs:complexType name="CatalogItem" >
 <xs:all>
  . . .
  <xs:element name="supplier">
   <xs:complexType>
    <xs:sequence>
     <xs:element ref="cml:Organization"/>
    </xs:sequence>
   </xs:complexType>
  </xs:element>
 </xs:all>
</xs:complexType>
```

Enforces association multiplicity (1..1 in this example)

 13

XMLmodeling.com

UML Class Inheritance




```
<xs:element name="Product" type="cml:Product"
            substitutionGroup="cml:CatalogItem" />

<xs:complexType name="Product">
 <xs:complexContent>
  <xs:extension base="cml:CatalogItem">
   <xs:all>
    <xs:element name="photoURL"
                type="xs:uriReference"/>
    <xs:element name="units"
                type="cml:UnitOfMeasure"/>
   </xs:all>
  </xs:extension>
 </xs:complexContent>
</xs:complexType>
```

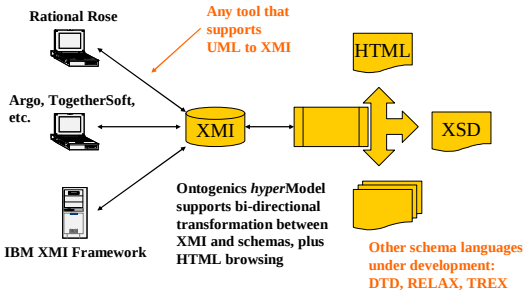
 14

Schema Generation

XMLmodeling.com

XMLmodeling.com

UML Model Transformation



 16

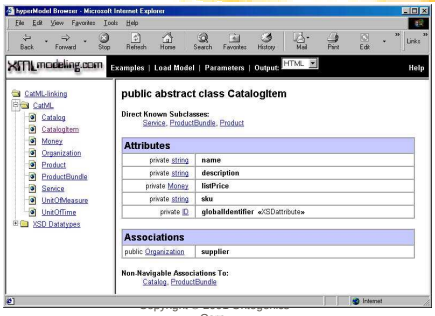
XMLmodeling.com

XML Metadata Interchange (XMI)

- Standard XML interchange format for UML
  - adopted by the Object Management Group (OMG)
  - gaining widespread acceptance
- But, XMI is more general than this…
  - create a DTD/schema for any MOF metamodel
  - serialize XML document for metamodel instance
  - UML is only one of several metamodels using XMI
    - Common Warehouse Metamodel (CWM)
    - CORBA Component Model

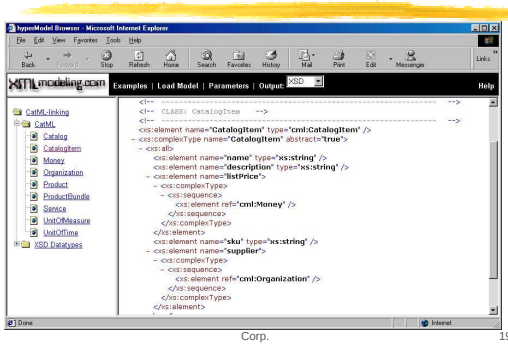
XMLmodeling.com

*hyper*Model UML Model Browsing



XMLmodeling.com

*hyper*Model XML Schema Generation



Customized Mapping

XMLmodeling.com

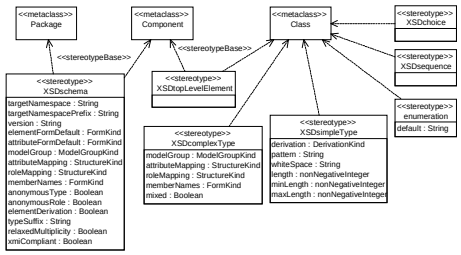
XMLmodeling.com

UML Profile for XML

- Stereotypes
  - An extension to the UML language itself. A stereotype allows a modeler to attach a new meaning to one of the UML foundational elements, e.g. Class, Attribute, Association, Package, etc. A stereotype is usually represented on a diagram as the name surrounded by guillements << >>. Alternatively, a stereotyped element can be shown as an icon on the UML diagram.
- Tagged Value
  - An extension to the attributes of a UML model element. Each model element has a standard set of attributes; for example, each UML Class has a name, visibility, etc. A tagged value defines a new attribute that is associated with a stereotyped model element. A tagged value is shown on the diagram as a name/value string: {tagName=value}.

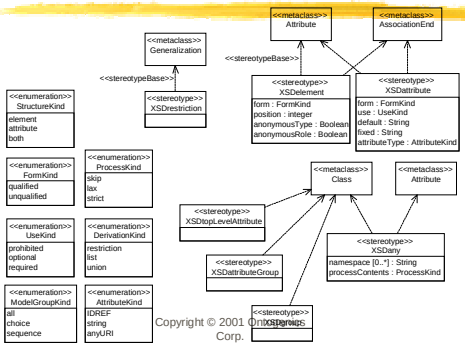
XMLmodeling.com

UML Profile for XML (part 1)



XMLmodeling.com

UML Profile for XML (part 2)



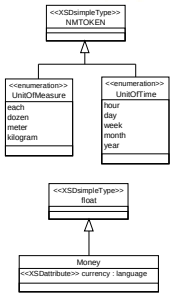
XMLmodeling.com

<<XSDattribute>>

UML Construct	Attribute AssociationEnd
Description	This stereotype may be assigned to either a UML Attribute or an AssociationEnd to indicate that the corresponding UML construct should be generated as an *attribute* definition within the parent complexType, and not generated as an *element* definition.
Tagged Values	form (qualified | unqualified) – overrides the attributeFormDefault selection set for the <<XSDschema>> containing this definition. use (prohibited | optional | required) – As defined by the XML Schema specification. default – string containing attribute default value fixed – string containing attribute fixed value

XMLmodeling.com

<<XSDsimpleType>>

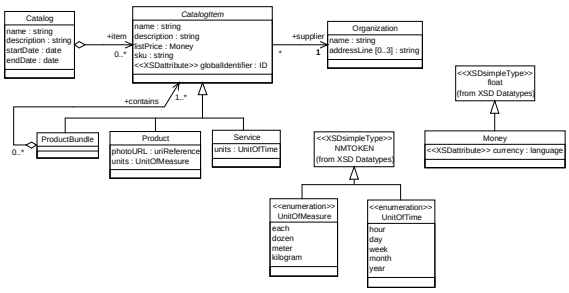


UML Construct	Class
Description	Define a new XML Schema simpleType. The UML class with this stereotype is usually a specialization of another simpleType, possibly owned by the XML Schema Datatypes namespace. If no superclass is specified, then derive this simpleType from 'string'.
Tagged Values	derivation (restriction | list | union) – select one of the three kinds of derivation, as defined by the XML Schema Datatypes specification. pattern – as defined by the XML Schema Datatypes specification. length – as defined by the XML Schema Datatypes specification. minLength – maxLength – and others…

25

XMLmodeling.com

UML Diagram with Stereotypes



26

XMLmodeling.com

Customized CatML Schema

```
<xs:complexType name="CatalogItem" abstract="true">
  <xs:all>
    <xs:element name="name" type="xs:string"/>
    ...
  </xs:all>
  <xs:attribute name="globalIdentifier" type="xs:ID" use="required"/>
</xs:complexType>

<xs:complexType name="Money">
  <xs:simpleContent>
    <xs:extension base="xs:float">
      <xs:attribute name="currency" type="xs:language" use="required"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
```


27

Reverse Engineering

XMLmodeling.com

- **Manual**
  - XHTML (represent DTD modularity)
- **Automated**
  - NewsML, FpML, xCBL, UDDI, WSDL, xlang, DocBook, TREX grammar, XMI 2.0

XMLmodeling.com

Reverse Engineering XHTML

- Divide a large schema into reusable parts
  - "modularization" in the XHTML specification
- Explore the ability to map text-oriented DTDs into UML, then into an W3C Schema
- The modularization is defined by content group parameter entities in the DTD. How well do these map to generalization and inheritance in the UML model?

 29

XMLmodeling.com

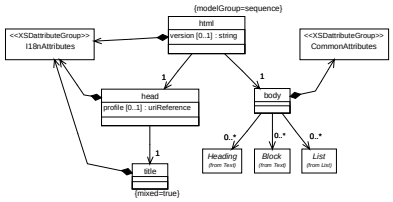
Goals of XHTML Specification

- XHTML represents the essential core of elements required for presentation of hypertext documents
  - omit the embedded presentation tags and attributes from HTML 4.0
- XHTML Basic was designed for simple devices such as mobile phones, PDAs, pagers, etc.
- XHTML Basic modules may also be included in other vocabularies
  - e.g., allow HTML text in a product description element content, but without allowing <table>

 30

XMLmodeling.com

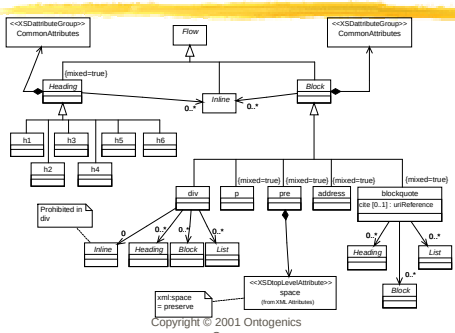
Structure Module



 31

XMLmodeling.com

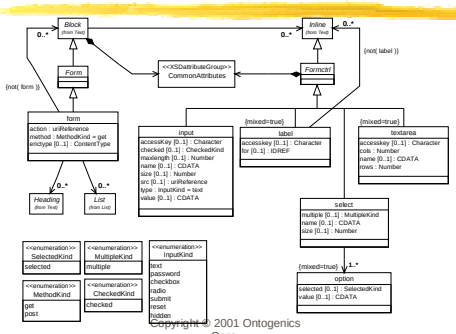
Text Module



 32

XMLmodeling.com

Basic Forms Module



XMLmodeling.com

Lessons Learned from XHTML

- Generalization in UML provides a very effective way to represent reusable content groups in XML Schema
  - Association to an abstract superclass allows all subclasses to be included in that relationship
- Shared attribute collections are sometimes difficult to represent without excessive duplication
  - However, generalization hierarchy helps to keep attributeGroup associations from cluttering the model
- It's difficult to capture all constraints in the UML model structure (e.g. cannot contain *Inline*)
  - Schema languages like Schematron (based on constraint rules) may be complementary to W3C Schema

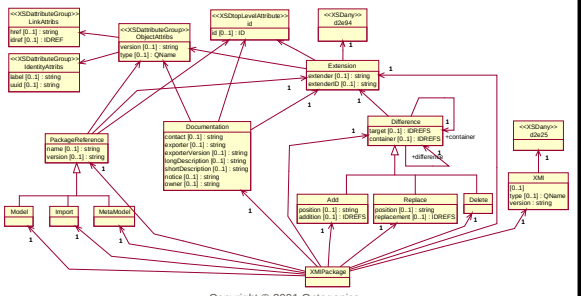
XMLmodeling.com

Reverse Engineering XMI 2.0

- UML model was reverse engineered from the XMI 2.0 schema included in the June 2001 spec
- Comparable with the original UML model includes in Chapter 4, except <<XSDany>> classes replace MOF RefBaseObject
- This model was generated on the airplane ride to Boston...

XMLmodeling.com

XMI 2.0 imported from XML Schema



XMLmodeling.com

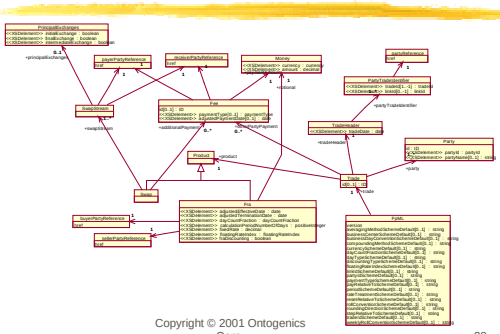
FpML Vocabulary

- Financial Products Markup Language (FpML)
- FpML Architecture document is based on object-oriented design practices, mapped into guidelines for manually authoring DTD/XSD
- Lessons:
  - manual division of model into readable diagrams
  - enhance reverse engineering stylesheet to recognize plain associations, aggregations, and composition

 37

XMLmodeling.com

FpML Model (*small* excerpt)



 38

XMLmodeling.com

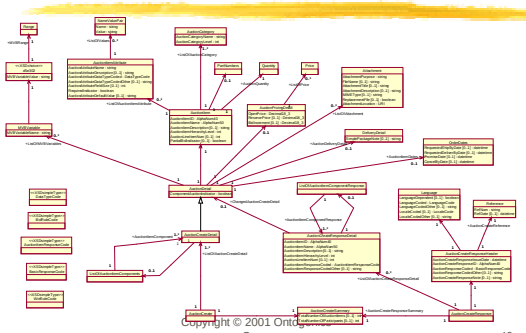
xCBL from SOX to UML to XSD

- Automated reverse engineering from SOX schema to UML model
  - (SOX is an OO schema language developed by CommerceOne, to be replaced by XML Schema)
- Then, generated XSD schema from the UML model which successfully validates original XML document instances for auction messages.
- This UML model was produced from SOX

 39

XMLmodeling.com

xCBL AuctionCreateDetail



 40

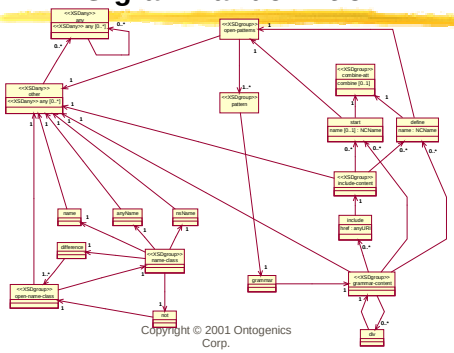
XMLmodeling.com

Reverse Engineering RELAX NG

- Preliminary work-in-progress to reverse/forward engineer TREX schemas
  - (TREX & RELAX now combined and renamed to RELAX NG, moving toward ISO standard via OASIS)

XMLmodeling.com

RELAX NG grammar definition



XMLmodeling.com

Challenges to Resolve

- Reverse engineering schemas into a good platform independent conceptual model
- Automated class diagram layout is pretty bad…
- Many UML tools do not allow display of tagged values (e.g. position & modelGroup)
- Some schema constraints don't map well to UML
- We need to separate UML model structure definition from design profile values