

ICSE 2000
Limerick, Ireland



**Workshop on
Standard Exchange Format**



Topic : *MOF/XMI/UML & CDIF*

By

St-Pierre, L. "Using the UML and the XML Metamodel Interchange format (XMI) with COBOL Systems"

Tichelaar, S., Ducasse, S., & Demeyer, S. "FAMIX: Exchange Experience with CDIF and XMI"

Dirckze, R., Baisley, D., & Iyengar, S. "XMI – A Model Driven XML Interchange Format"

June 06, 2000

WoSEF: MOF/XMI/UML & CDIF

2

The first text shows the versatility of the UML which can be applied, with restrictions, for non-design activities and in a non-OO paradigm.

The second text shows why the UML is not enough and would at the very least need to be extended for extensive non-design related activities.

The third text is a thorough presentation of the enormous strengths of XMI and the MOF as a common framework to achieve and expand an independent exchange mechanism.



MOF/XMI/UML & CDIF

- Why Standard Exchange Formats?
- The OMG Model Driven Approach
- XML Metadata Interchange (XMI) format
- Experiences with CDIF & XMI

June 06, 2000

WoSEF: MOF/XMI/UML & CDIF

Why Standard Exchange Formats?

- Today's computing model is
 - A Cooperating Distributed Model
 - Heterogeneous
 - Autonomous
 - Internet based
 - Cooperation is achieved by
 - APIs - for fined grained cooperation & integration
 - IDL, Java, etc.
 - Stream based - for coarse grained information exchange
 - HTTP, XML, CDIF, etc.

June 06, 2000

WoSEF: MOF/XMI/UML & CDIF

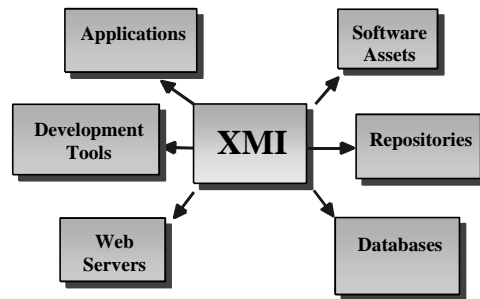
Information about software components is usually complex (many different types of elements are involved and they may have multiple relationship permutations). The API approach is not the best option.

One can publish its own interfaces and data format to allow others to communicate with it (ex: UML Rose repository).

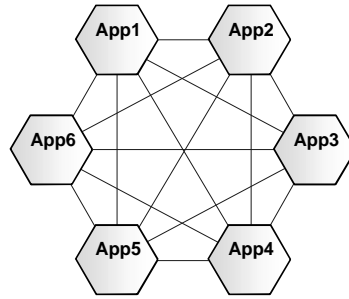
Or

Everybody can reach an agreement on an independent data format for the exchange.

Why "SEF" Contd.



6 bridges written by 6 vendors.



$N*N-N = 30$ bridges written
by $N = 6$ vendors.

June 06, 2000

WoSEF: MOF/XMI/UML & CDIF

At the extreme the publish approach can lead to a multiplication of the effort required to publish OR import the information unless a prevalent format arises in which case the owner of the format controls the exchange.

At the extreme the consensus approach can lead to a difficult and heavy process of adoption and revision of the information model(s). One way to avoid that is by having a modular framework.



The OMG Model Driven Approach

- Meta-metamodel (M3), metamodels (M2), and models (M1)
 - The MOF is M3
 - Describes other M2 models including
 - Modeling Languages, Technology models, Middleware models, etc.
 - Rules for API generation for accessing models
 - IDL and Java
 - Rules for metadata interchange format generation
 - XML Metadata Interchange (XMI) format
 - XML DTDs for content verification

June 06, 2000

WoSEF: MOF/XMI/UML & CDIF

The layered architecture of the MOF intrinsically provides an extendible environment to develop various metamodels for different needs and share metamodels elements between these metamodels.

For example, the Common Warehouse Model re-uses packages from the UML metamodel.

The Model Driven Approach contd.

- M2 describes (meta) models
 - Modeling Languages – UML, FAMIX, etc.
 - Technology models - CWM, EDOC, etc.
 - Business object models – CCM, EJB, etc.
 - Workflow models
 - Services – JNDI, CosNaming, JTS, etc.

- M1 Describes application models
 - Finance application model, etc.

June 06, 2000

WoSEF: MOF/XMI/UML & CDIF

In the MOF family the challenge is in the integration of the different metamodels to avoid metamodel overlap (multiple definitions of the same types of elements) or refactoring (definition of an element inside the wrong metamodel).



Why M3 Approach?

- **Abstraction**
 - The model abstracts functionality from implementation
 - Architecture/implementation independence
 - Reuse of functionality not code
 - Easy to manage/understand
 - Metadata enabled
- **Semantic Interoperability**
 - Semantic integration of applications/components
 - Normative APIs
 - Information interchange
 - XMI

June 06, 2000

WoSEF: MOF/XMI/UML & CDIF

XMI

- Requirements for interchange format
 - Need to be “understood” by systems that exchange data
 - Data needs to be augmented with metadata
 - Machine consumable
 - Human Readable (for debugging, etc.)
 - Convenient for querying
 - Supports industry standards

June 06, 2000

WoSEF: MOF/XMI/UML & CDIF

Introduction to XMI

- XML that conforms to the MOF rules is XMI
 - XMI is subset of XML
- Middleware neutral standard for model interchange
 - Exchange the MOF (M3)
 - Exchange metamodels (M2)
 - Exchange models (M1)

Note : The MOF specifies the DTD generation rules for any MOF compliant metamodel, as well as the XMI stream for any instance data of MOF compliant metamodels.

June 06, 2000

WoSEF: MOF/XMI/UML & CDIF

XMI can be describe as a set of rules for translating MOF compliant metamodels into an XML description (DTD).

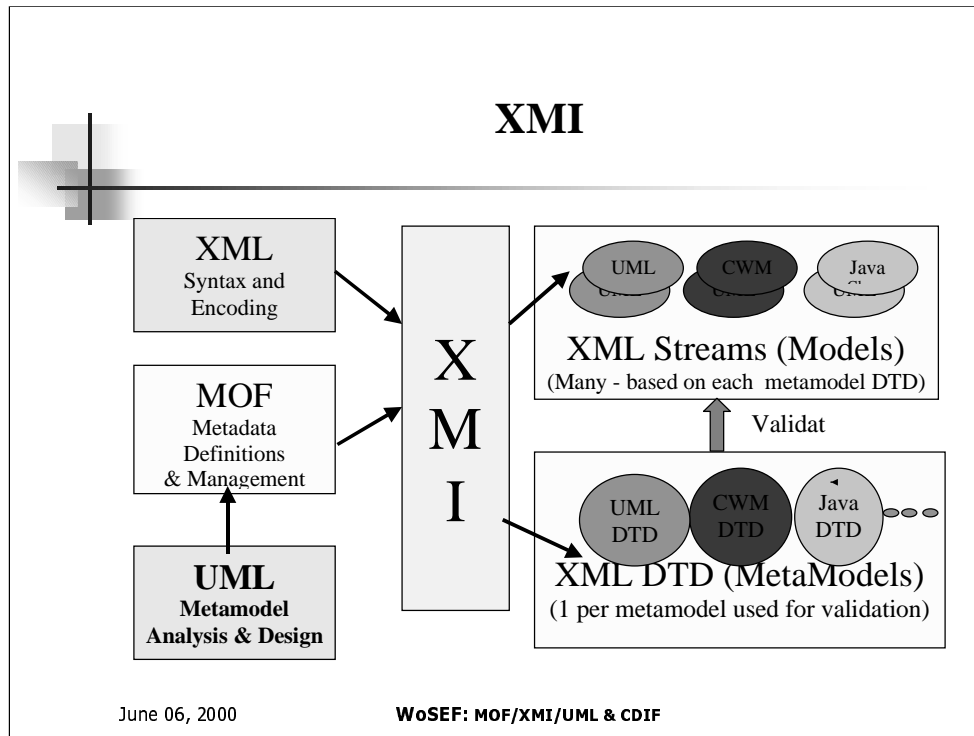
All XMI files are XML files that are valid against a DTD generated automatically from the metamodel (ex: the UML DTD).

XMI Status

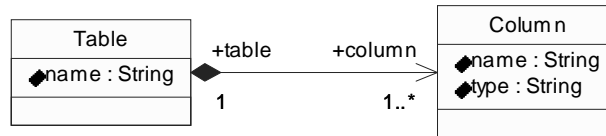
- XMI 1.0
 - OMG standard on 23rd March, 1999
- XMI 1.1
 - OMG standard on 2nd February 2000
 - Includes X-links and XML Namespaces
- XMI 1.2
 - XML schemas coming soon!

June 06, 2000

WoSEF: MOF/XMI/UML & CDIF



XMI Example



- Simple RDB metamodel
 - Table has “name” of type String
 - Column has “name” and “type” of type String
 - Table has one or more columns
 - Column belongs to one table

June 06, 2000

WoSEF: MOF/XMI/UML & CDIF

RDB.DTD (XMI1.1)

```
<!-- All XMI DTDs must reference the fixed part -->
<!-- XmiFixed.dtd -->
<!ENTITY % fixedDTD SYSTEM "XmiFixed.dtd">
%fixedDTD;
<!ATTLIST XMI xmlns:RDB CDATA #IMPLIED >
<!-- PACKAGE: RDB:SimpleRDB -->
<!-- ***** RDB:tableHasColumn ***** -->
<!ELEMENT RDB:Table.column ( RDB:Column)* >
<!-- CLASS: RDB:Table -->
<!ELEMENT RDB:Table.name (#PCDATA|XMI.reference)*>
<!ENTITY % RDB:TableProperties '((RDB:Table.name)?)' >
<!ENTITY % RDB:TableCompositions '(RDB:Table.column*)' >
<!ENTITY % RDB:TableAttPropsList 'name CDATA #IMPLIED' >
```

June 06, 2000

WoSEF: MOF/XMI/UML & CDIF

RDB.DTD *Contd.*

```
<!ELEMENT RDB:Table ( %RDB:TableProperties;
    ,(XML:extension* )
    , %RDB:TableCompositions; )?>
<!ATTLIST RDB:Table %RDB:TableAttPropsList; %XML:element.att; %XML:link.att;>
%XML:link.att ; >
<!-- CLASS: RDB:Column -->
<!ELEMENT RDB:Column.name (#PCDATA|XML:reference)*>
<!ELEMENT RDB:Column.type (#PCDATA|XML:reference)*>
<!ENTITY % RDB:ColumnProperties '((RDB:Column.name)?
    ,(RDB:Column.type)?)' >
<!ENTITY % RDB:ColumnAttPropsList 'name CDATA #IMPLIED
    type CDATA #IMPLIED' >
<!ELEMENT RDB:Column ( %RDB:ColumnProperties;
    ,(XML:extension* ) )?>
<!ATTLIST RDB:Column %RDB:ColumnAttPropsList; %XML:element.att; %XML:link.att; >
```

June 06, 2000

WoSEF: MOF/XML/UML & CDIF

RDB Instance Data

```
<?xml version="1.0" encoding="UTF-8" ?>
<XMI xmi.version='1.1'>
  <XMI.header>
    <XMI.metamodel xmi.name='SimpleRDB' xmi.version='1.0'/>
  </XMI.header>
  <XMI.content>
    <RDB:Table name='PERSON'>
      <RDB:Table.column>
        <RDB:Column name='name' type='String'>
          <RDB:Column name='socialSecurityNumber' type='CHAR(9)'>
        </RDB:Table.column>
      </RDB:Table>
    </XMI.content>
  </XMI>
```

June 06, 2000

WoSEF: MOF/XMI/UML & CDIF

Experiences with CDIF & XMI

- CDIF is good
 - Poor support for explicit relationships, no multi-valued strings, etc.
 - Development halted – helped shape XMI
- XMI is the future
 - XML based, model driven, middleware independent, etc.
 - Supports industry standards – MOF, UML, etc.
 - XMI (1.0) is verbose
 - XML Namespace support will cut document size

June 06, 2000

WoSEF: MOF/XMI/UML & CDIF

The rapidity with which UML, XML and XMI have been adopted as standard and the rapidity by which they have been accepted and more importantly implemented in various tools suggest that they are our best options at the moment.

UML and Source Code Representation

Problem: UML is a model for OO Analysis and Design

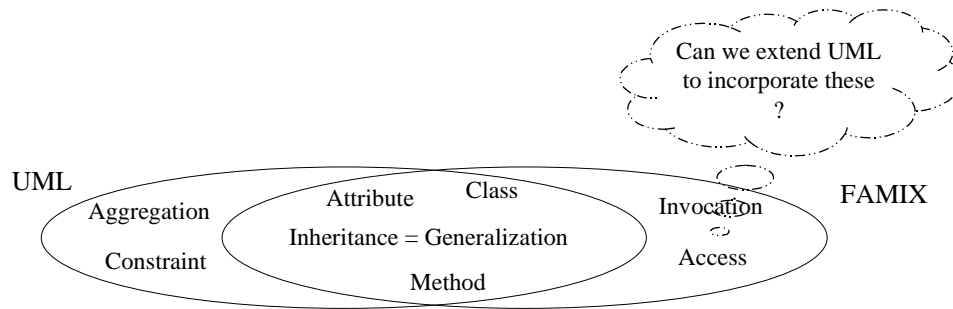
- Mapping COBOL to UML
 - Detail left out
 - COBOL specifics (e.g. PICTURE clause)
 - less useful constructs (e.g. expression composition)
 - Information outside the scope of UML
 - Metrics
 - Software Management Elements

June 06, 2000

WoSEF: MOF/XMI/UML & CDIF

UML and Source Code Representation (2)

■ Mapping Object-Oriented Source Code to UML



June 06, 2000

WoSEF: MOF/XMI/UML & CDIF

FAMIX represents object-oriented source code.

We didn't take UML as our metamodel because it is not fit to represent object oriented source code.

Although classes, methods and attributes nicely map to UML constructs,

- Inheritance has to be interpreted as being UML Generalization, although in source code it is not always generalization, but implementation inheritance.

- Aggregations and Constraints are not implementation concepts

- Although UML is extendible, Invocations and Accesses are difficult to introduce in UML in an optimal way for all purposes.

Conclusion

- There is much value in the MOF/XMI because of the extensible and layered model approach and also because of the general availability of the technology (UML, XML).
- With the adoption of more metamodels, MOF/XMI is being positioned as a model driven middleware architecture.

June 06, 2000

WoSEF: MOF/XMI/UML & CDIF