CWMI RFP
Final Submission Presentations
(ADTF Denver meeting, 3/8/00)

• CWM Overview (10 mins, Dan Chang)
• CWM Metamodel, Part 1 (20 mins, Doug Tolbert)
• CWM Metamodel, Part 2 (20 mins, John Poole)
• CWM Generation, Validation, and Extensions (20 mins, David Mellor)
• CWM Summary and Actions (10 mins, Dan Chang)

• Vote to recommend CWM Adoption & CWM FTF Formation (10 mins, Jim Odell)
CWM
(Common Warehouse Metamodel)
Overview

Dan Chang (dtchang@us.ibm.com)
CWM

• A complete specification of the syntax and semantics needed to export/import shared warehouse metadata and the common warehouse metamodel, including:
  – The CWM Metamodel (Volume 1)
  – Interchange format for shared warehouse metadata (CWM DTD, Volume 2)
  – Interchange format for the CWM Metamodel (CWM XML, Volume 2)
  – Access API for shared warehouse metadata (CWM IDL, Volume 2)
CWM Co-submitting Companies

- **IBM** (*Dan Chang, J. J. Daudenarde, Debra LaVergne, Christoph Lingenfelder*)
- **Unisys** (*Sridhar Iyengar, Don Baisley, Doug Tolbert*)
- **NCR** (*Vilhelm Rosenqvist, Bruce McLean*)
- **Hyperion** (*John Poole, David Zhang*)
- **Oracle** (*Gordon Callan, David Last, David Mellor, Mark Hornick*)
- **UBS** (*Hans-Peter Hoidn, Jeffrey Peckham*)
- **Genesis** (*David Frankel, Phil Longden*)
- **Dimension EDI** (*Chris Nelson, Anders Tornqvist*)

Expertise in UML, XML, metadata repository, databases, data warehousing, and business intelligence (OLAP, data mining)
CWM Supporting Companies

- **Deere** *(Dave Smith)*
- **Sun** *(Chuck Mosher, Karsten Riemer, Nidhi Rao)*
- **HP** *(Jishnu Mukerji)*
- **Data Access** *(Cory Casanave)*
- **InLine Software** *(Jack Greenfield)*
- **Aonix** *(Charles Simon)*
- **Hitachi** *(Yuichi Yugawa)*

Primary expertise in using databases, data warehouses, and business intelligence tools
CWM Design Basis

• OMG Metamodelling Architecture
  – **UML** as the standard language for defining metamodels
  – **XMI** as the standard mechanism for interchanging metadata and metamodels in **XML**
  – **MOF to IDL Mapping** as the standard mechanism for accessing metadata through APIs (independent of programming languages and object models)
OMG Metamodeling Architecture

- User Data/Object Layer (M0)
- Metadata/Model Layer (M1)
- Metamodel Layer (M2)
- Meta-metamodel Layer (M3)

Stock: name, price

UML: Class, Attribute
CWM: Table, Column
ElementType, Attribute

MOF: Class, Attribute,
Operation,
Association

<Stock name="IBM" price="112"/>
OMG Modeling Architecture

Applications, Tools, Repositories

Metamodels (UML, CWM, …)

Meta Object Facility (MOF Model, MOF-IDL)
XML Metadata Interchange (XMI)
Roles of UML in CWM

• The *metamodeling language* (as in the MOF Model)
  – UML Semantics, UML Notation, OCL

• The *foundation metamodel*
  – UML Foundation, Common_Behavior, and Model_Management packages

• The *object (resource) metamodel*
  – Same as above
The CWM Metamodel

<table>
<thead>
<tr>
<th>Management</th>
<th>Warehouse Process</th>
<th>Warehouse Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analysis</td>
<td>Transformation</td>
<td>OLAP</td>
</tr>
<tr>
<td>Resource</td>
<td>Object (UML)</td>
<td>Relational</td>
</tr>
<tr>
<td>Foundation</td>
<td>Business Information</td>
<td>Data Types</td>
</tr>
</tbody>
</table>

**UML 1.3**
(Foundation, Behavioral_Elements, Model_Management)
CWM Metamodel, Part 1

Doug Tolbert
(doug.tolbert@unisys.com)
Package Architecture

Modular Design

– Minimum dependencies
  • Cross package services provided by links to UML
– Avoid subpackages
– Reduced complexity, improved understanding
– Use only the packages you need
Base

The modeling environment

- **UML notation** used as diagramming technique
- **UML metamodel** extended to support warehouse concepts

<table>
<thead>
<tr>
<th>Management</th>
<th>Warehouse Process</th>
<th>Warehouse Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analysis</td>
<td>Transformation</td>
<td>OLAP</td>
</tr>
<tr>
<td>Resource</td>
<td>Data Mining</td>
<td>Information Visualization</td>
</tr>
<tr>
<td>Foundation</td>
<td>Information</td>
<td>Business Nomenclature</td>
</tr>
<tr>
<td></td>
<td>Visualization</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Object (UML)</th>
<th>Relational</th>
<th>Record</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Multi-Dimensional</td>
<td>XML</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Business Information</th>
<th>Data Types</th>
<th>Expressions</th>
<th>Keys Index</th>
<th>Type Mapping</th>
<th>Software Deployment</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**UML 1.3**

*(Foundation, Behavioral_Elements, Model_Management)*
**Foundation**

Metamodels shared by other packages

- **Foundation**
  - Business Information
  - Data Types
  - Expressions
  - Keys & Indexes
  - Software Deployment
  - Type Mapping

<table>
<thead>
<tr>
<th>Management</th>
<th>Warehouse Process</th>
<th>Warehouse Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analysis</td>
<td>Transformation</td>
<td>OLAP</td>
</tr>
<tr>
<td>Resource</td>
<td>OLAP</td>
<td>Data Mining</td>
</tr>
<tr>
<td>Foundation</td>
<td>Data Mining</td>
<td>Information Visualization</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Business Nomenclature</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Object (UML)</th>
<th>Relational</th>
<th>Record</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multi-Dimensional</td>
<td>XML</td>
<td></td>
</tr>
</tbody>
</table>

- **UML 1.3** (Foundation, Behavioral_Elements, Model_Management)
• Business Information
  – Responsible parties & their contact information
  – Documentation and general commentary
  – Hierarchies of business types can be constructed
• Data Types
  – Supporting classes for modeling of data type systems
  – Extent class represents collections of instances
Foundation

- **Keys & Indexes**
  - Shared by several data resource models
  - Promotes similar representation across models
Foundation

• Expressions
  – Tree-structured, functional model of expressions
  – Full access to all CWM objects
  – Supports both “black box” and “white box” expressions
  – Use “white box” expressions for interchange and lineage
Foundation

- **Software Deployment**
  - **SoftwareSystem**
    - A software package on a CD
  - A Deployed Software System is a set of DeployedComponents
    - An installed SoftwareSystem
  - Each Deployed Component is on a specific Machine
    - An installed program
Foundation

• Software Deployment
  – DataManagers provide access to data
    • A deployed DBMS
  – DataProviders wrap other DataManagers as specified by ProviderConnections
    • ODBC and JDBC drivers
Foundation

• Type Mapping
  – Map types to corresponding types in other systems
  – Designed for simple data type exchanges
  – Use Transformations for more complex mappings
Describe logical and physical data containers

- Operational sources
- Warehouse targets
- Logical models

<table>
<thead>
<tr>
<th>Management</th>
<th>Warehouse Process</th>
<th>Warehouse Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analysis</td>
<td>Transformation</td>
<td>OLAP</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Data Mining</td>
</tr>
<tr>
<td>Resource</td>
<td></td>
<td>Information Visualization</td>
</tr>
<tr>
<td>Foundation</td>
<td></td>
<td>Business Nomenclature</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Object (UML)</th>
<th>Relational</th>
<th>Record</th>
<th>Multi-Dimensional</th>
<th>XML</th>
</tr>
</thead>
<tbody>
<tr>
<td>Business Information</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data Types</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Expressions</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Keys</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Index</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Type Mapping</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Software Deployment</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

UML 1.3
(Foundation, Behavioral_Elements, Model_Management)
## Data Resource Matrix

<table>
<thead>
<tr>
<th>Resource</th>
<th>Intent</th>
<th>Extent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object Oriented</td>
<td>Package</td>
<td>Extent</td>
</tr>
<tr>
<td>Relational</td>
<td>Catalog/Schema</td>
<td>Object</td>
</tr>
<tr>
<td>Record</td>
<td>RecordFile</td>
<td>Column Value</td>
</tr>
<tr>
<td>Multi-dimensional</td>
<td>Schema</td>
<td></td>
</tr>
<tr>
<td>XML</td>
<td>Schema</td>
<td></td>
</tr>
</tbody>
</table>

**Note:** The matrix illustrates the mapping of resources to intents and extents in various data models.
Data Resource Packages

- Relational
  - RDBMS catalogs & ODBC/JDBC client catalog views
  - SQL-99 compliant
Data Resource Packages

• Record
  – Basis for traditional databases & files
  – Self-describing, delimited, & fixed-offset supported
Data Resource Packages

• Multidimensional
  – Physical representation of multidimensional databases
Data Resource Packages

- **XML**
  - Supports XML 1.0
  - Basis for XML documents
  - Allows use as sources and as targets
CWM Metamodel, Part 2

John Poole
(john_poole@hyperion.com)
Data Analysis

Describes production & analysis of warehouses

- Describe analytical & deployment structures
- Design data movement & transformations
- Deployable on a number of data resources

<table>
<thead>
<tr>
<th>Management</th>
<th>Warehouse Process</th>
<th>Warehouse Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analysis</td>
<td>Transformation</td>
<td>OLAP</td>
</tr>
<tr>
<td>Resource</td>
<td>Data Mining</td>
<td>Information</td>
</tr>
<tr>
<td>Foundation</td>
<td></td>
<td>Visualization</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Business Nomenclature</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Object (UML)</th>
<th>Relational</th>
<th>Record</th>
<th>Multi-Dimensional</th>
<th>XML</th>
</tr>
</thead>
<tbody>
<tr>
<td>Business Information</td>
<td>Data Types</td>
<td>Expressions</td>
<td>Keys</td>
<td>Type Mapping</td>
<td>Software Deployment</td>
</tr>
</tbody>
</table>

UML 1.3
(Foundation, Behavioral_Elements, Model_Management)
Transformation

- General mechanism describing data movement and lineage
- Generic transformations from any physical object to any other physical object
- Maps logical structures and concepts in the warehouse onto physical implementation
- Provides for multiple physical implementations of logical structures and concepts
Transformation

- Support for “White-Box” transformation mappings

- Leverages the UML hierarchy: Mapping of Classifier-to-Classifier and Feature-to-Feature

- Mapping of Classifier-to-Feature
Transformation

- Transformations can be specified for arbitrary model elements
- Reification of the transformation “process”
- Relates to Warehouse Process and Warehouse Operation metamodels
OLAP

- Analytical model: Cubes, measures, dimensions, attributes, levels, and hierarchies
- Cubes contain multiple measures and are implemented via cube regions
OLAP

- Dimensions: Multiple levels, attributes, and hierarchies

- Levels are used in multiple hierarchies; support a subset of the dimension attributes
Data Mining

- Models the fundamental meta data necessary for constructing and managing Data Mining models
- Three conceptual areas: Model, Settings and Attributes
Data Mining

- Models major settings types: Statistical, Clustering, Association Rules, Supervised

- Relates settings to specification and attributes
Information Visualization

- CWM core metamodel for information visualization and publishing
- Separation of "logical" rendered object from rendering "transformation"
- Recursive/composite structuring
Business Nomenclature

• CWM metamodel for “Business Metadata”
• Intended for Data Warehousing and Business Intelligence domains
• Common business terms and concepts
• Used in conjunction with analysis and reporting tools
Business Nomenclature

- Taxonomies consist of concepts
- Glossaries consist of terms
- Taxonomies and Glossaries can be associated
- All are relevant to some “business domain”
Warehouse Management

Orchestrates warehouse activities

- **Warehouse Process** relates
  - Transformations
  - Triggering events

- **Warehouse Operation** logs
  - Transformation activity
  - Metrics

<table>
<thead>
<tr>
<th>Warehouse Process</th>
<th>Warehouse Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transformation</td>
<td>OLAP</td>
</tr>
<tr>
<td></td>
<td>Data Mining</td>
</tr>
<tr>
<td></td>
<td>Information Visualization</td>
</tr>
<tr>
<td></td>
<td>Business Nomenclature</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Resource</th>
<th>Analysis</th>
<th>Foundation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object (UML)</td>
<td>Relational</td>
<td>Record</td>
</tr>
<tr>
<td></td>
<td>Multi-Dimensional</td>
<td>XML</td>
</tr>
</tbody>
</table>

| Management        | | |
|-------------------| | |
| Business Information | Data Types | Expressions |
|                   | Keys       | Type Mapping |
|                   | Mapping    | Software Deployment |

| | |
| UML 1.3            | |
| (Foundation, Behavioral_Elements, Model_Management) | |
Warehouse Process

- A Warehouse Process identifies warehouse tasks and the events that trigger them (“what gets done”)
- Relates Transformation Activity to Event types
Warehouse Operation

- Activity and Step Execution track details of executions of Transformations (tracks “what got done when”)
- Step Execution may be related to a UML CallAction model element
Warehouse Operation

- ChangeRequest objects represent a proposed change or one that has been implemented or rejected.
- Measurement objects can hold values for any object (such as volumetric details)
CWM Generation, Validation, and Extension

David Mellor
(dmellor@us.oracle.com)
CWM Generation

- **Metamodel**
  - Single logical source
  - Multiple packages
- **Generated for each package**
  - XML document
  - CWM DTD
  - CWM IDL
CWM Development

• Metamodel Proposal
  – An information metamodel is proposed to be included in the submission
  – Group analysis to determine if the metamodel should become part of CWM

• Model Team
  – A team is formed to develop a metamodel as part of the CWM submission
CWM Development

- Metamodel Development
  - Metamodels constructed by domain knowledgeable modelers
  - A CWM Metamodel is constructed by analyzing common portions of existing Warehouse tool models as well as public reference models
  - Tool logical models can be re-constructed using CWM as common starting point
CWM Validation

• Use Case Scenarios

  – Develop use cases of the individual models

  – Develop representative use cases that involve several packages used in combination

  – Develop use cases that represent entire target tools
CWM Validation

Validation

- Each Use Case defines a “slice” through the model being validated
- Use Cases represent “typical” problems that the model must solve
CWM Extensions

• CWM provides interchange of the common portions of warehouse tool meta models
• CWM should be used as the new foundation of a warehouse tool model
• Volume 3 of the Specification contains examples of CWM Extensions
CWM Extension Packages

- **Olap** (from Analysis)
- **InformationSet** (from Analysis)
- **Information Reporting** (from Analysis)
- **Record** (from Resource)
- **Multidimensional** (from Resource)
- **COBOLDData** (from Resource)
- **IMSDatabase** (from Resource)
- **Essbase** (from Resource)
- **Express** (from Resource)
- **DMSII** (from Resource)
- **UML** (from org.omg)
- **ER** (from Foundation)

[Diagram of CWM Extension Packages]
CWM Extensions

- CWM Extensions
  - Extensions
    - Published tool specific information for the purpose of interchange
    - Physical characteristics of databases not common in general but interchangeable in a heterogeneous environment
CWM Extensions

- CWM Extensions
  - Extensions
    - Common ancestry in base metamodel
    - Tool-specific default structures (e.g., Database)
Extending CWM

• Extending the model to redefine a working tool metamodel without generating a new DTD.
  
  – Categorized into three generic types:

  • Proprietary Attributes: Tool specific definitions not intended for interchange

  • Proprietary Associations and or Classes: Tool specific areas not common and not intended for interchange

  • Sharable Extensions: Tool specific definitions proposed for interchange via tagged value pairs
Extending CWM

• Extending the model to redefine a working tool metamodel without generating a new DTD.
  • **Proprietary Attributes: Tool specific definitions not intended for interchange**
  • Proprietary Associations and or Classes: Tool specific areas not common and not intended for interchange
  • Sharable Extensions: Tool specific definitions proposed for interchange via tagged value pairs
Extending CWM

- Extending the model to redefine a working tool metamodel without generating a new DTD.
  - Proprietary Attributes: Tool specific definitions not intended for interchange
  - Proprietary Associations and or Classes: Tool specific areas not common and not intended for interchange
  - Sharable Extensions: Tool specific definitions proposed for interchange via tagged value pairs
Extending CWM

- Extending the model to redefine a working tool metamodel without generating a new DTD.
  - Proprietary Attributes: Tool specific definitions not intended for interchange
  - Proprietary Associations and or Classes: Tool specific areas not common and not intended for interchange
  - **Sharable Extensions:** Tool specific definitions proposed for interchange via tagged value pairs

```
CWM - Relational
  - Table
     - Column

ToolX - Relational
  - Column
     - DisplayName

ToolX - Relational.Column
  - Inherited Class Attributes
  - Leaf Class Attributes

Interchanged Via CWM
Interchanged Via CWM Tagged ValuePairs
```
CWM
Summary and Actions

Dan Chang (dtchang@us.ibm.com)
CWM

• A common specification that defines, in UML, the structure and semantics of shared metadata in data warehousing and business intelligence
  – Resource: Object, Relational, Record, Multidimensional, XML
  – Analysis: Transformation, OLAP, Data Mining, Information Visualization, Business Nomenclature

• A common specification that defines, in XML, the interchange format and, in IDL, the access API for such shared metadata
CWM Specification:
CWM XML, CWM DTD, CWM IDL

- CWM Metamodel (in UML Notation)
- CWM XML
- CWM DTD
- CWM IDL
- MOF DTD
- CWM Metadata Interchange (in XML)
- CWM Metadata Access
CWM Extensions (CWMX)

• Published vendor specific metamodel for the purpose of metadata interchange (Volume 3 & Volume 4, non-normative)
  – Common ancestry in the CWM metamodel
  – Demonstrates the validity of the CWM metamodel
  – Demonstrates the extensibility of the CWM metamodel
## Data Resource Matrix

<table>
<thead>
<tr>
<th>Resource</th>
<th>Intent</th>
<th>Extent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object Oriented</td>
<td>Package</td>
<td>Extent</td>
</tr>
<tr>
<td>Relational</td>
<td>Catalog/Schema</td>
<td>Object</td>
</tr>
<tr>
<td>Record</td>
<td>RecordFile</td>
<td>Data Value</td>
</tr>
<tr>
<td>Multi-dimensional</td>
<td>Schema</td>
<td>RecordSet</td>
</tr>
<tr>
<td>XML</td>
<td>Schema</td>
<td>Record</td>
</tr>
<tr>
<td></td>
<td>ElementType</td>
<td>MemberSet</td>
</tr>
<tr>
<td></td>
<td>Attribute</td>
<td>Member</td>
</tr>
<tr>
<td></td>
<td>Intent Extent</td>
<td>Member Value</td>
</tr>
<tr>
<td></td>
<td>Value</td>
<td>Data Value</td>
</tr>
</tbody>
</table>
CWM

- Enables interchange and access of shared metadata at *three abstraction levels*

![Diagram](image)
CWM: Past, Present and Future

• Past
  – Initial submission: 9/17/99
  – OMG Demo: 11/99
  – Evaluation: 9/99 - 1/00

• Present
  – Final submission: 2/11/00
  – Evaluation: 2/00 - present

• Future
  – Adopted specification: 3/10/00 (we hope!)
  – CWM FTF : 3/10/00 (we hope!)
  – Available specification: 9/00 (we plan)
  – Product enablement and interoperability showcase
November 1999 OMG Demo

Common Warehouse Metamodel

Metadata Interchange Flow

DB2/UDB
Unisys
Hyperion
Oracle
Dimension EDI

IBM
UREP
Essbase
Express RAA
PolyVal XML Mediator