eclipse CON 2004

Rapid plug-in development and integration using the Eclipse Modeling Framework (EMF)

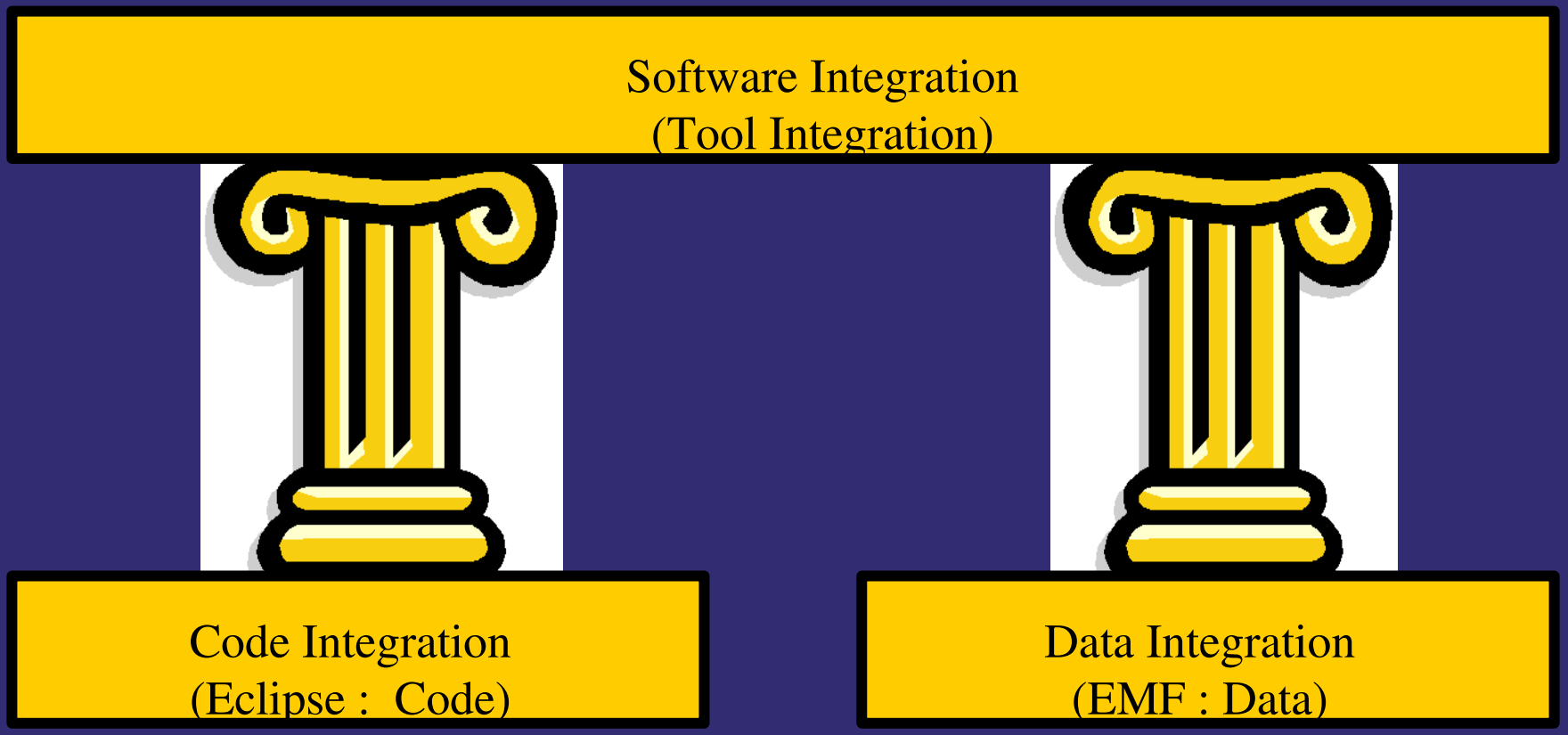
Sridhar Iyengar, IBM Distinguished Engineer
Ed Merks, Eclipse XSD Project Lead

 February 2004

Topics

- EMF Overview
- Developing plug-ins using EMF
- EMF, MDA, and building an open, integrated tools platform

Role of EMF in Tool Integration



For simplicity we don't discuss 'process integration' which is not addressed by EMF today

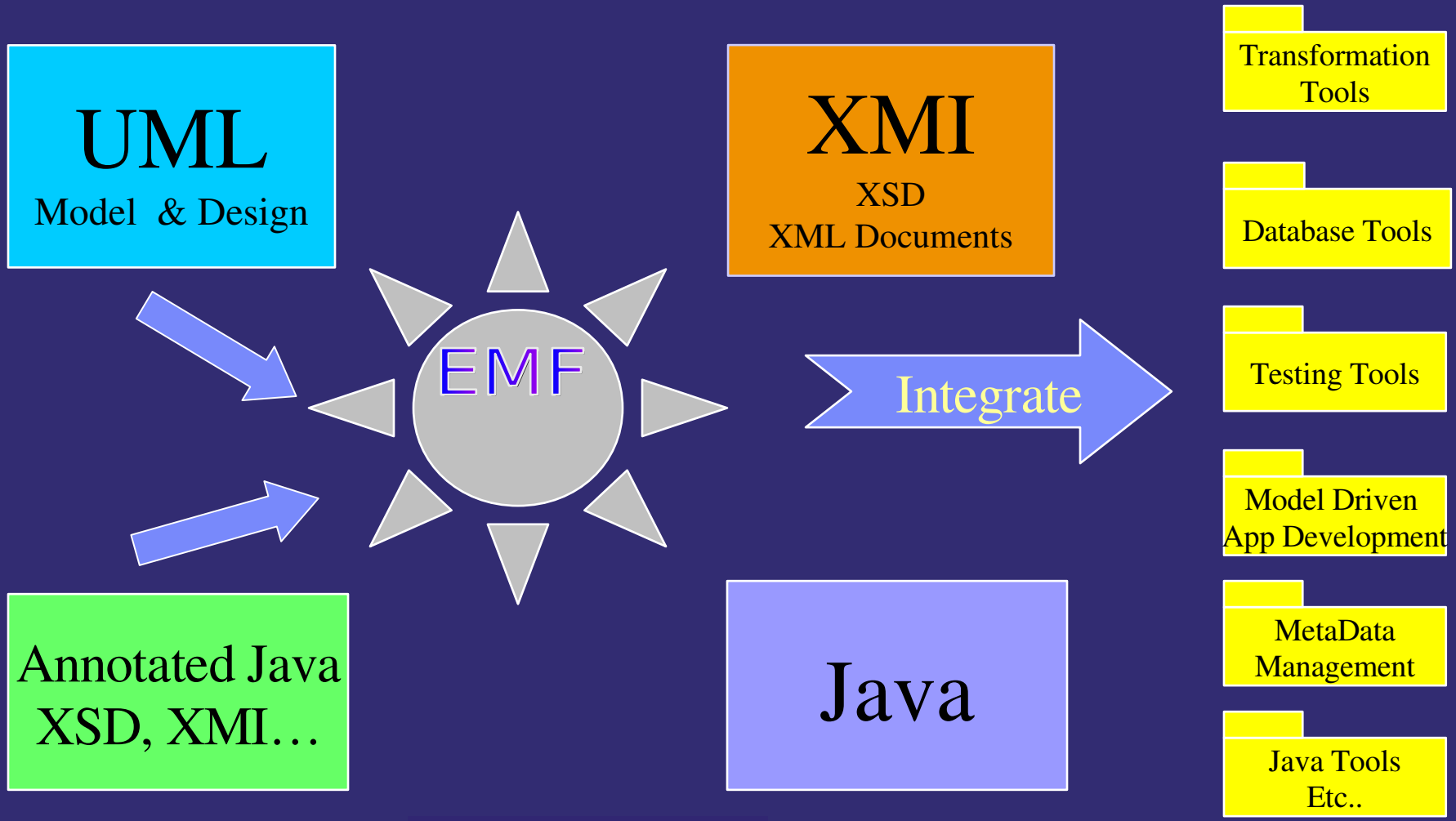
What is EMF?

- A modeling & data Integration framework for Eclipse
- What do we mean by an EMF “model”?
  - Specification of an application’s data
    - Object attributes
    - Relationships (Associations) between objects
    - Operations available on each object
    - Simple constraints (ex: cardinality) on objects and relationships
  - Essentially the Class Diagram subset of UML
- From a model definition, EMF can generate efficient, correct, and easily customizable implementation code
  - Java code, XML documents, XML Schemas…

What is EMF? (cont)

- EMF takes a pragmatic middle ground in the modeling vs. programming world
- Very low cost of entry
  - Full scale graphical modeling tool not required
  - Can be used with richer modeling and XML development tools
- Designed to make modeling practical and useful to the mainstream Java programmer
- Pragmatic unification of key technologies
  - Java (Model implementation)
  - XML (Model persistence)
  - UML & MOF (Model definition)
- Influence of & by OMG MOF, XMI, and UML

EMF - The Data Integration Foundation of Eclipse



www.eclipse.org/emf

EMF History

- Originally based on MOF (Meta Object Facility)
  - From OMG (Object Management Group)
  - Abstract language and framework for specifying, constructing, and managing technology neutral meta-models
- EMF evolved based on experience supporting a large set of tools
  - Efficient Java implementation of a practical subset of the MOF API
  - To avoid confusion, the MOF-like core meta model in EMF is called **Ecore** instead of MOF
- Foundation for model based WebSphere Studio family product set
  - Example: J2EE model in WebSphere Studio Application Developer
- 2003: EMOF defined (Essential MOF)
  - Part of MOF 2 specification; UML2 based; recently approved by OMG
  - EMF is approximately the same functionality
    - Significant contributor to the spec; adapting to it
    - Supports MOF2::EMOF XMI import now

What is Model Driven Development (MDD)?

- A discipline that commonly uses
  - Models (UML class models, Data models, XML Schemas, etc.) to represent application concepts
  - Code generation patterns to translate artifacts in these models to more detailed (not necessarily complete) implementation artifacts such as Java code, XML documents, XML Schemas, etc.
    - Patterns derived after years of experience in specific usage scenarios
  - The generated code (usually CRUD operations, simple events, referential integrity for relationships, persistence, etc.) is complemented by programmer developed code
- The Object Management Group (OMG) is defining a set of standards that enable MDD using the Model Driven Architecture (MDA) initiative
  - More on this initiative and how to use EMF to bootstrap an MDD tools platform after we delve into EMF

A Simple Example

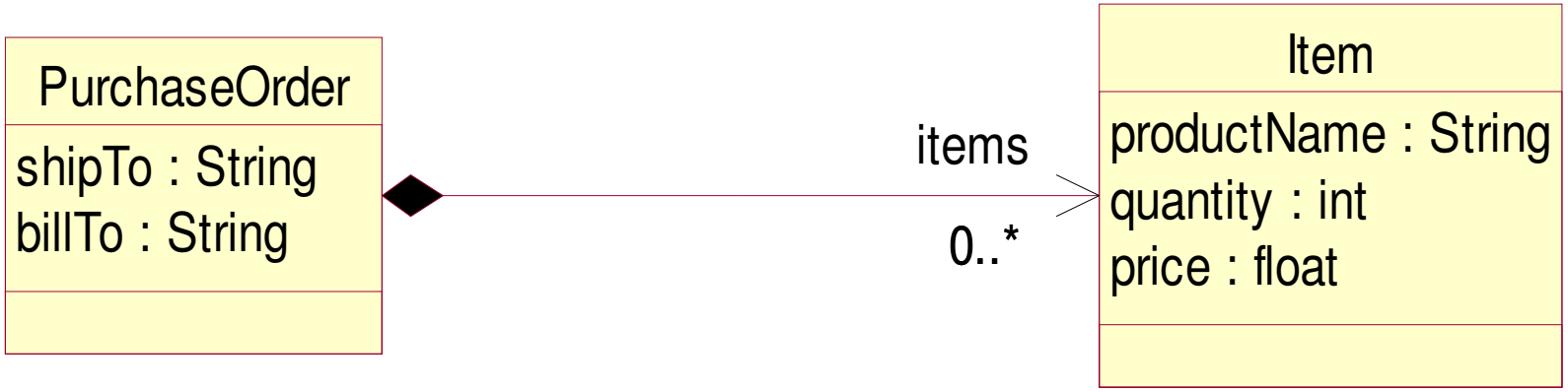
- Let's say you want to write a program to manage purchase orders for some store or supplier
  - Purchase orders include "bill to" and "ship to" addresses, and a collection of (purchase) items
  - Items include a product name, quantity, and price
- Depending on your perspective and skills, you could start by defining the "model" in one of three ways:
  1. Java Interfaces
  2. UML Class Diagram
  3. XML Schema

A Simple Example (cont) – 1. Java Interfaces

```
public interface PurchaseOrder
{
  String getShipTo();
  void setShipTo(String value);
  String getBillTo();
  void setBillTo(String value);
  List getItems(); // List of Item
}

public interface Item
{
  String getProductName();
  void setProductName(String value);
  int getQuantity();
  void setQuantity(int value);
  float getPrice();
  void setPrice(float value);
}
```

A Simple Example (cont) – 2. UML Class Diagram



A Simple Example (cont) – 3. XML Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"

  targetNamespace="http://www.example.com/SimplePO"
            xmlns:PO="http://www.example.com/SimplePO">
  <xsd:complexType name="PurchaseOrder">
    <xsd:sequence>
      <xsd:element name="shipTo" type="xsd:string"/>
      <xsd:element name="billTo" type="xsd:string"/>
      <xsd:element name="items"  type="PO:Item"
                   minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="Item">
    <xsd:sequence>
      <xsd:element name="productName" type="xsd:string"/>
      <xsd:element name="quantity" type="xsd:int"/>
      <xsd:element name="price" type="xsd:float"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:schema>
```

A Simple Example (cont)

- All three forms provide the same information
  - Different visualizations/representation
- The application's "Model" of the structure
  - EMF Models are not intended to be complete (i.e., no detailed behavior – this is coded in Java)
  - UML models can be used for more complete application generation
    - www.eclipse.org/uml2

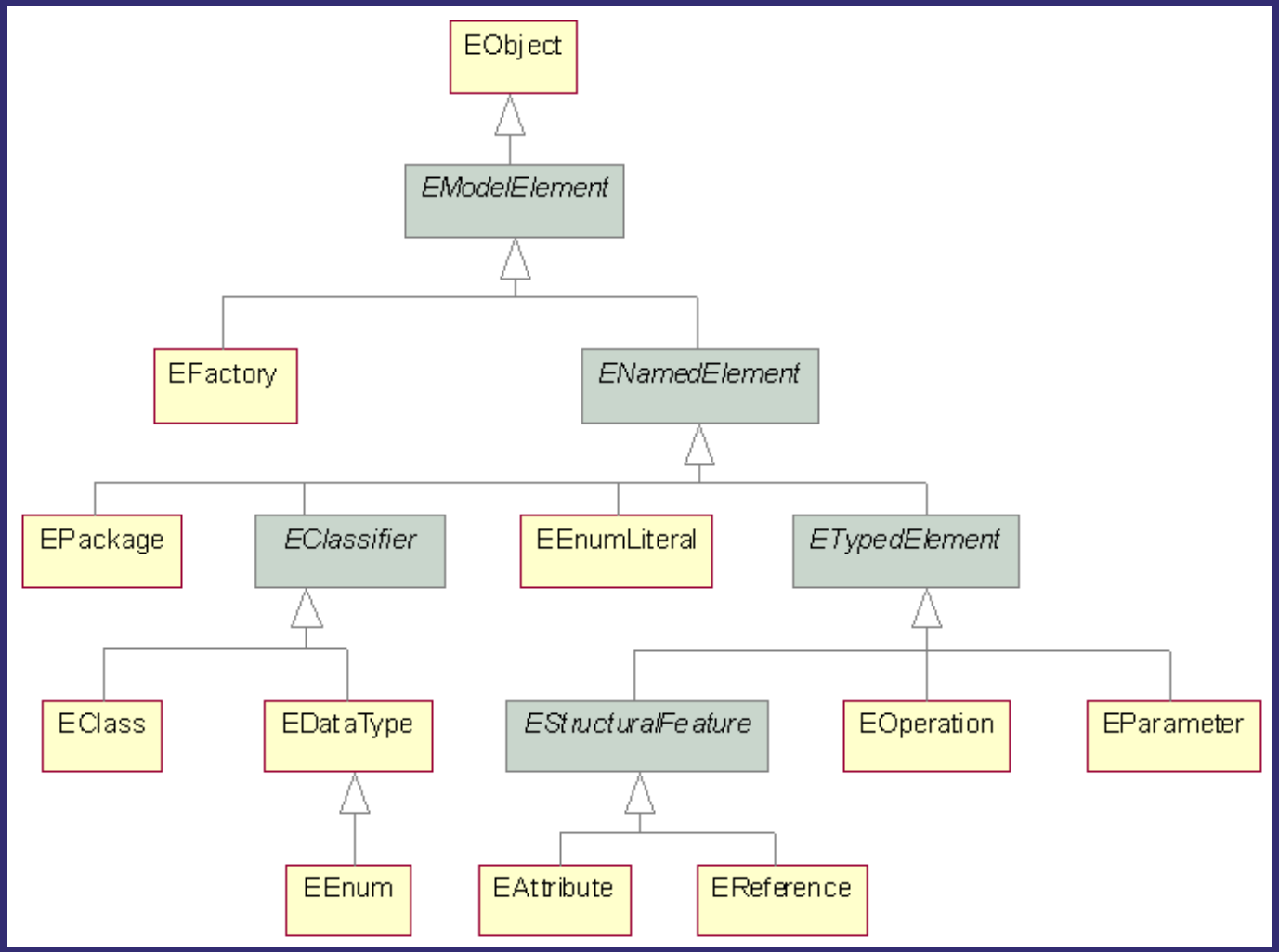
Generating a Model Implementation

- EMF can import any form of a model (Java, XSD, UML representations), and generate the others, as well as Java implementation code
- Generated Java code includes:
  - Model change notification
  - Persistence/serialization of model data
    - XML by default, but other persistent formats possible
  - Bidirectional reference handshaking (referential integrity)
  - Proxy resolution and demand-load
- EMF also automatically creates/generates other Eclipse parts (projects, plugin.xml, etc.)

The Ecore Meta-model

- EMF canonical representation of a data model is called Ecore
- The persistent form of Ecore is XMI
  - XMI : XML Metadata Interchange – an OMG standard for serializing models and metadata in XML
- Ecore model supports a superset of the modeling concepts in EMOF (Essential MOF), part of the OMG MOF 2.0 Specification
  - EMF supports EMOF definition of a model
  - EMOF/XMI2 is the supported “standard” interchange format
  - UML tools can be used to design EMOF/Ecore models

The Ecore Meta-model (cont)



EObject is the model root. Equivalent to java.lang.Object

EMF Runtime Framework

- All EMF models (generated or not) can be accessed using an efficient EMF reflective API (in interface EObject)
  - eGet(), eSet(), etc.
  - A standard common API that can be used to access models generically
- This is the key to integrating tools and applications built using EMF
- Also the key to providing generic editing commands (see next slide)

Generating Viewers/Editors for an EMF Model

- From an EMF Model, EMF can also generate view adapters and even a working Eclipse Editor for the model
  - Optionally in additional Plug-ins/Projects
- EMF.Edit provides a framework for easily building Eclipse editors for EMF-based applications
  - Content and label providers, property source support, and other convenience classes to display EMF models using standard desktop (JFace) viewers and property sheets
  - Command framework, including generic command implementation classes for building editors that support fully automatic undo and redo

EMF Demo

- In the Following Demo you will see:

1. Importing a simple model definition into EMF
2. Generating code – model, adapters, and an editor
3. Running the generated editor
4. Hand modifying the generated Java code
5. Modifying the model and regenerating the code

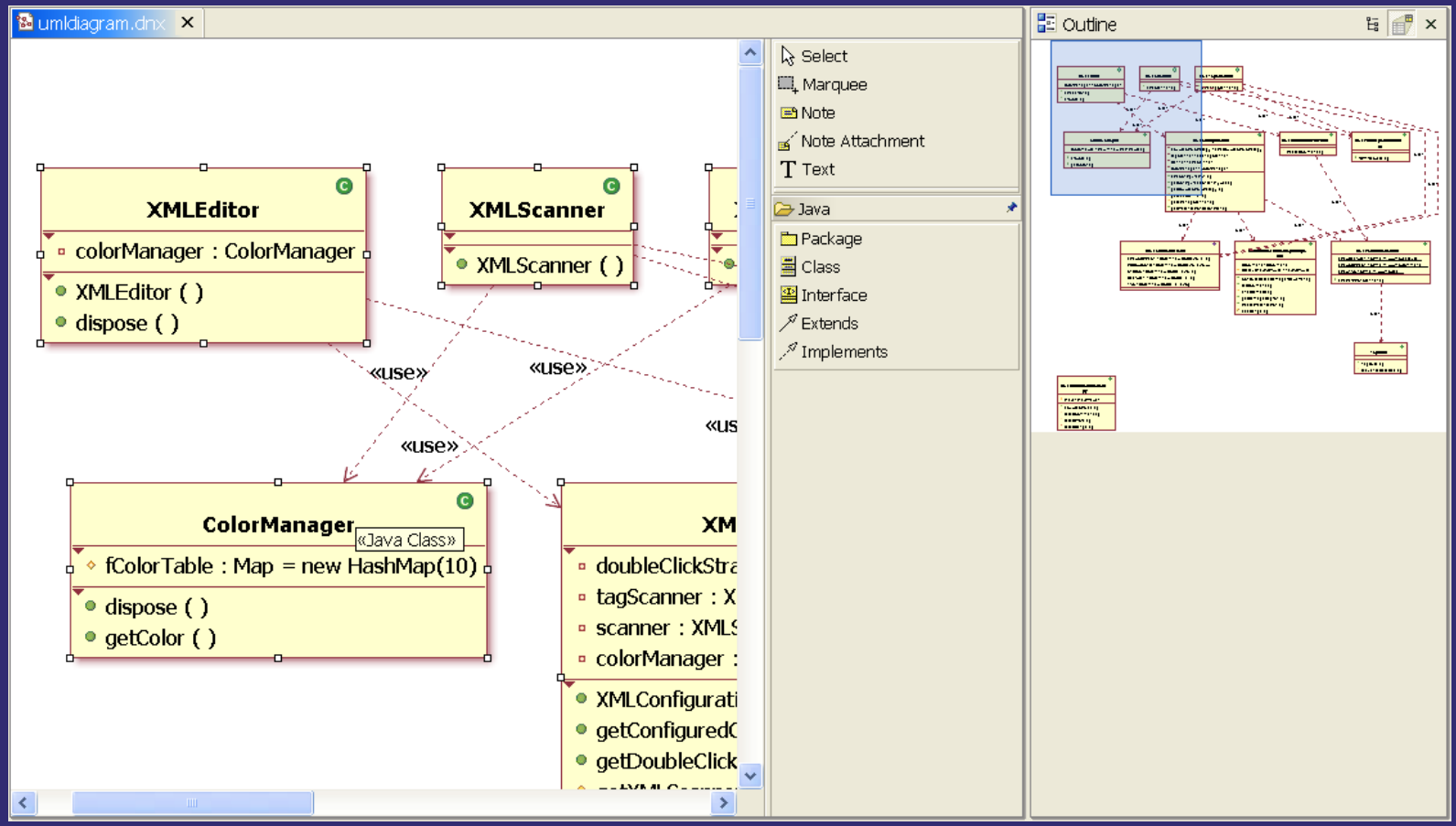
EMF Summary

- EMF provides low-cost modeling for the Java mainstream
- Boosts productivity and facilitates integration
- Mixes modeling with programming to maximize the effectiveness of both
- EMF is the foundation for fine-grain data integration in Eclipse

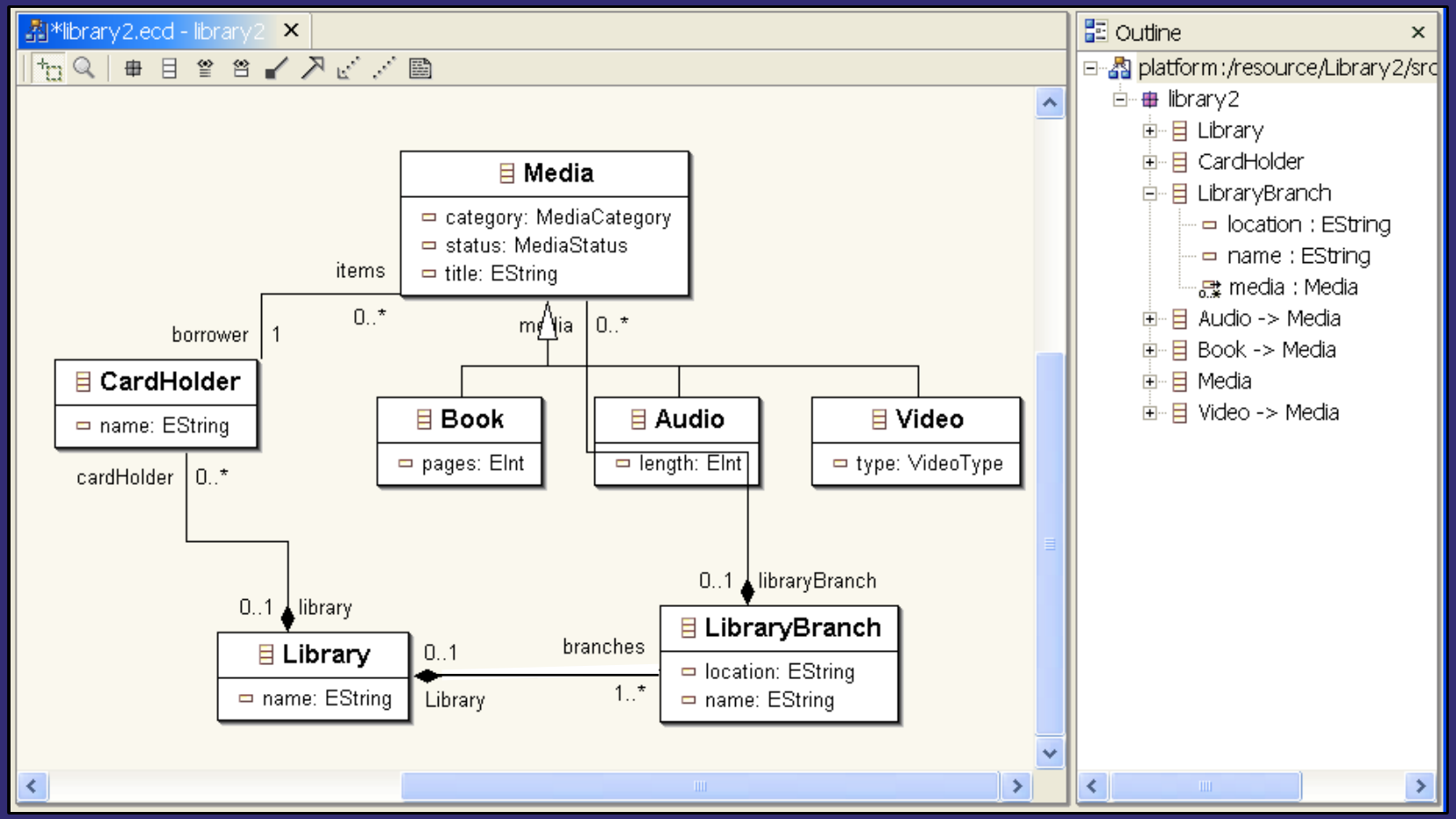
Who is using EMF today?

- IBM WebSphere Studio product family
- IBM WebSphere Application Server
- IBM Rational product family (in development)
- SDO Reference Implementation
- Eclipse Subprojects
  - Hyades Project (framework for testing and logging tools)
  - UML2 Project (implementation of the UML 2.0 metamodel)
  - VE Project (Visual Editor framework for creating GUI builders)
  - XSD Project (Java API for manipulating XML Schemas)
  - EMF Project itself (of course!)
- ISV's
  - TogetherSoft (UML editor and code generation)
  - Ensemble (support for Weblogic servers)
  - Versata (extend J2EE to capture their business rules)
  - Omondo (UML editor tightly coupled to EMF tools)

WebSphere Studio Application Developer: EMF based UML Visualization/Editing Tool for J2EE



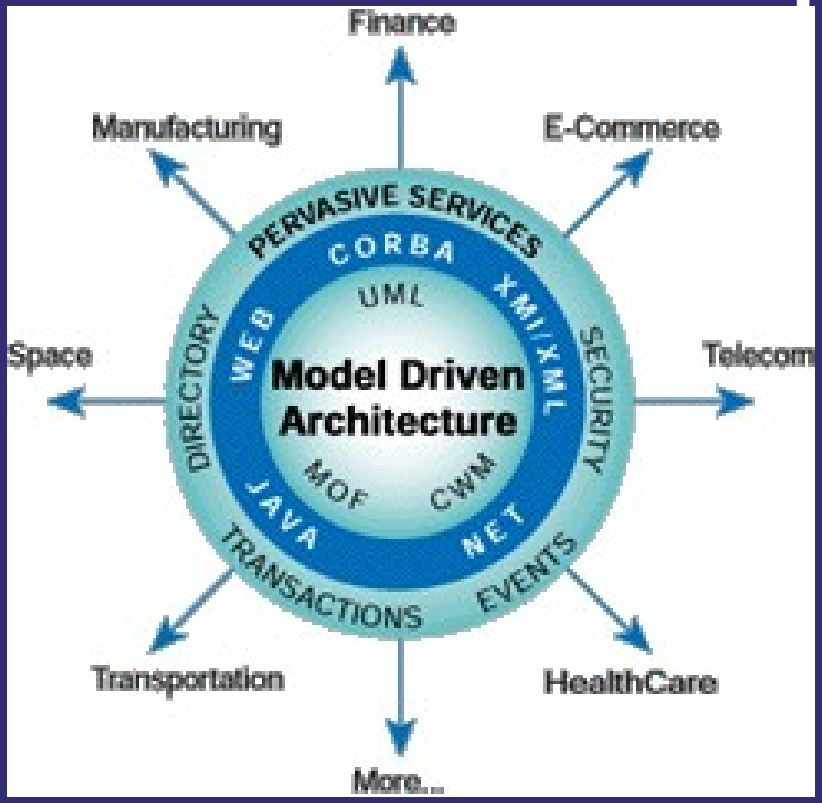
Omondo UML EMF Ecore Graphical Editor



eclipse CON 2004

EMF, MDA, and building an open, integrated tools platform

An Open Model Driven Integration Framework OMG ™ Model Driven Architecture (MDA)™



MDA is a framework, process and a set of standards

Source: OMG

An integration of best practices in Modeling, Middleware, Metadata, and Software Architecture

Model Driven (UML, MOF, CWM…)

- Platform Independent Models (PIM) – Technology or increasingly Business Models
- Platform Specific Models (PSM) - J2EE, .Net, SQL
- Mappings : PIM<->PIM, PSM<->PSM, PIM<->PSM
- Applies across the software life cycle

Key Benefits

- Improved Productivity for Architects, Designers, Developers and Administrators
- Lower cost of Application Development and Management
- Enhanced Portability and Interoperability
- Business Models and Technologies evolve at their own pace on platform(s) of choice

Open Modeling Standards: The Foundation of MDA

- Unified Modeling Language (UML)
  - For describing the problem domain and the solution architecture
- Meta Object Facility (MOF)
  - For describing and manipulating models
- XML Model Interchange (XMI)
  - For exchanging model information in XML format and generating XSD
- Common Warehouse Model (CWM)
  - For describing data mappings and database schemas
- And more
- Supported by UML profiles
  - For customizing UML to specific domains (e.g., J2EE, EAI, Real-time, Systems Engineering…)

IBM & the Industry is driving the core MDA standards
Eclipse.org is driving pragmatic integration of MDA standards using Eclipse

Model Transformations using OMG Model Driven Architecture (MDA)

Mappings
PIM – PSM
PIM – PIM…

Business Models

Auto
Color : String
Door : Integer
Engine : Integer

Engine
Cyl : integer
Fuel : String
Comp : Float

Computation Independent Model View

XMI
XSD,DTD

JMI, EMF4J..

JOLAP, JDM
UML4EJB…

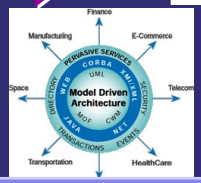
Platform Specific Implementation Model

Java, C#
Class Auto
{public String color;
public int Door;
public int Engine}

XML Doc.
<Auto>
<Color> Red </Color>
<Door> 4 </Door>
<Engine> 1 </Engine>
</Auto>

XML Schema
<xsd:ComplexType name = “Auto”>
<xsd:sequence>
<xsd:element name = “Color” type = “xsd:string”/>
<xsd:element name = “Door” type = xsd:integer”/>
<xsd:element name = “Engine” type = xsd:integer”/>
<xsd:/sequence>
<xsd:/ComplexType>

Platform Independent Models (PIM)



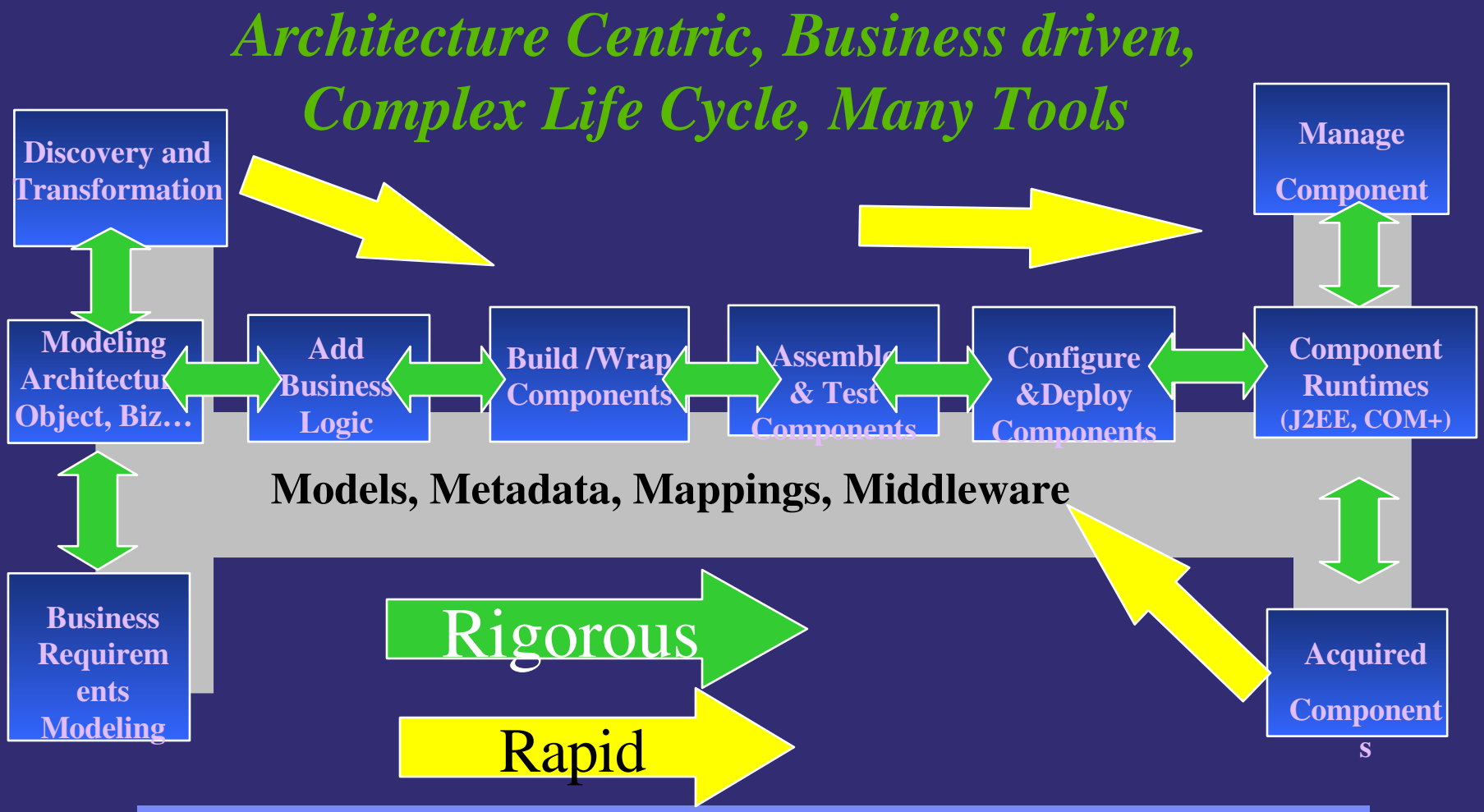
Platform/Language Specific Models (PSM)

Mappings to UDDI, WSDL, BPEL4WS being defined

MDA as an 'Architectural Style' for Model Driven Development & Integration

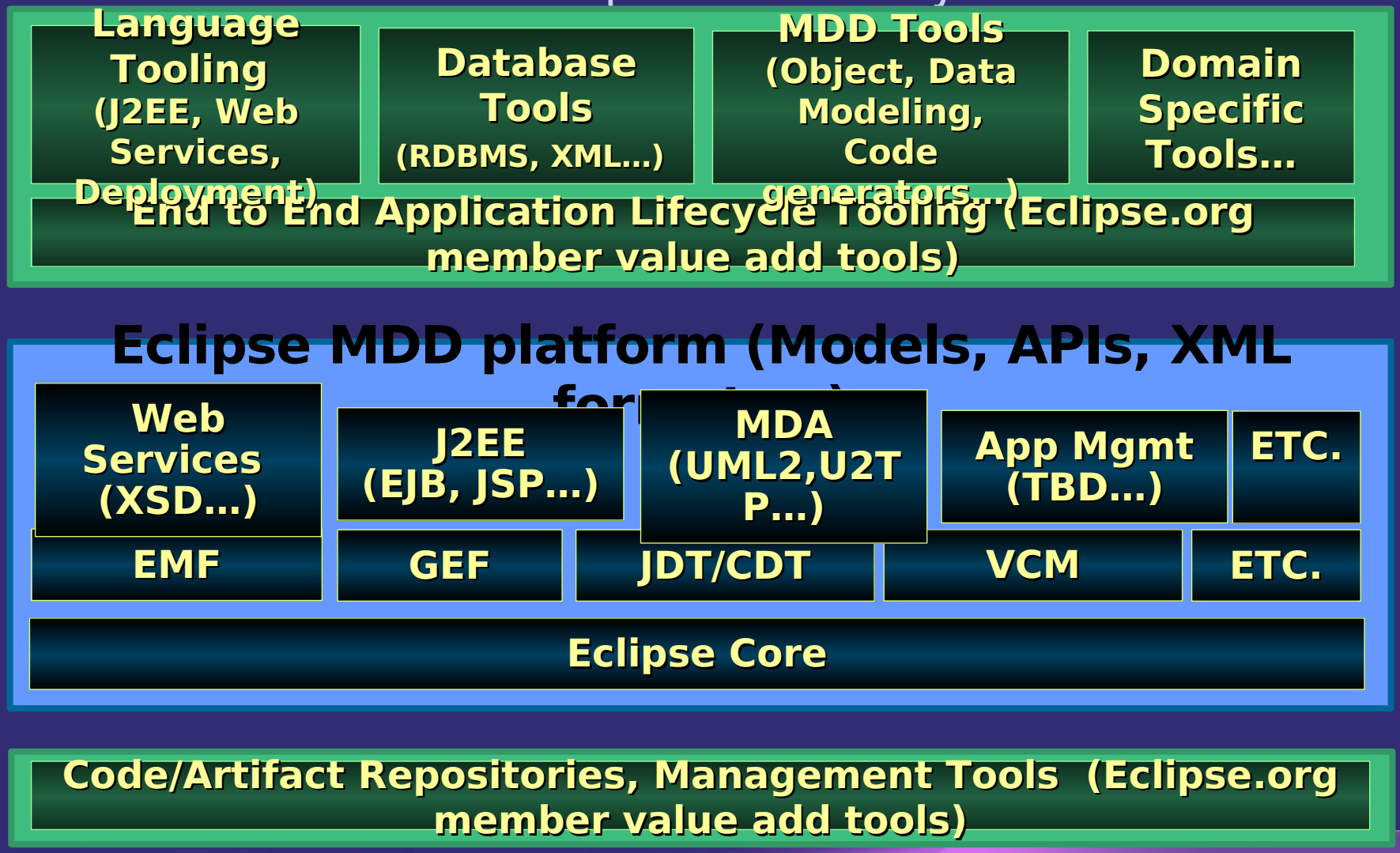
- Understand the problem domain (technology or business)
- Model the problem domain
  - Use UML for the visual modeling, analysis & design of meta model
  - Use UML profiles & MOF Models to define Domain Specific Metamodels or Languages as needed
- Formally represent the models and metadata using UML, MOF & XML
  - Simple class modeling is all you need to know
  - OCL (Object Constraint Language) can capture additional semantics
  - Reverse engineer existing DTD, XSD, XMI, Java to MOF/UML (jump start)
- Use Standard transformation (mappings & patterns) for
  - Metadata Interchange (XMI – MOF to XML, DTD, XSD)
  - Metadata Interfaces (JMI – MOF to Java, EMF4J – MOF2::EMOF to Java, MOF to WSDL, etc.)
- Use open source modeling frameworks for model integration and management
  - Eclipse EMF : www.eclipse.org/emf , Eclipse UML2 : www.eclipse.org/uml2
- Summary : Understand, Model, Map and Manage metadata to integrate

An Enterprise Application Development Life Cycle



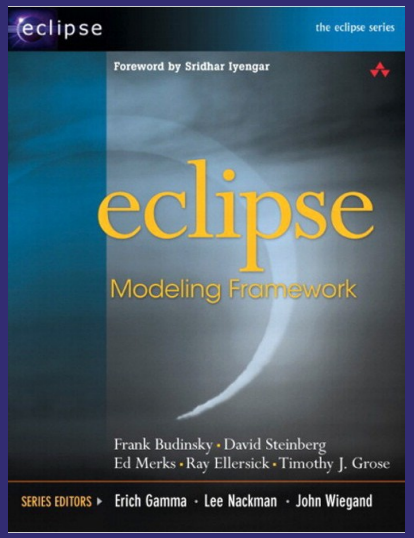
And do this with quality in a distributed environment

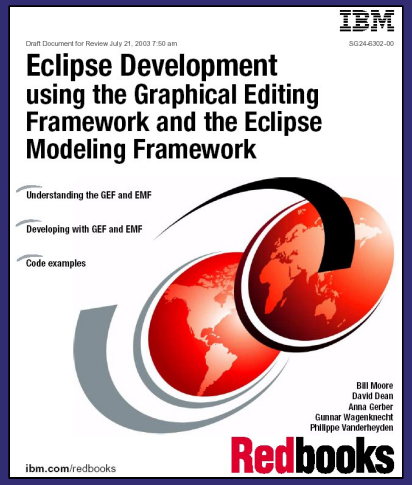
Possible Application Life Cycle Integration Platform
A call to action to the Eclipse Community



Additional Information Sources

- EMF documentation in Eclipse help
  - Documents, tutorials, API reference
- EMF project web site
  - http://www.eclipse.org/emf/
  - Documentation, newsgroup, mailing list, Bugzilla
- *Eclipse Modeling Framework* by Frank Budinsky et al. *
  - Addison-Wesley; 1st edition (August 13, 2003), ISBN: 0131425420
- Eclipse Development using the GEF and EMF
  - IBM Redbook
  - Publication number: SG24-6302-00
- www.omg.org/mda





* Material and images derived from this book

Trademarks

- OMG™, MDA™, UML™, MOF™, and XMI™ are trademarks of the Object Management Group
- IBM™ is a trademark of the IBM Corporation
- Java™ and J2EE™ are trademark of SUN Microsystems
- XML™ is a trademark of W3C
- All other trademarks belong to their respective organizations