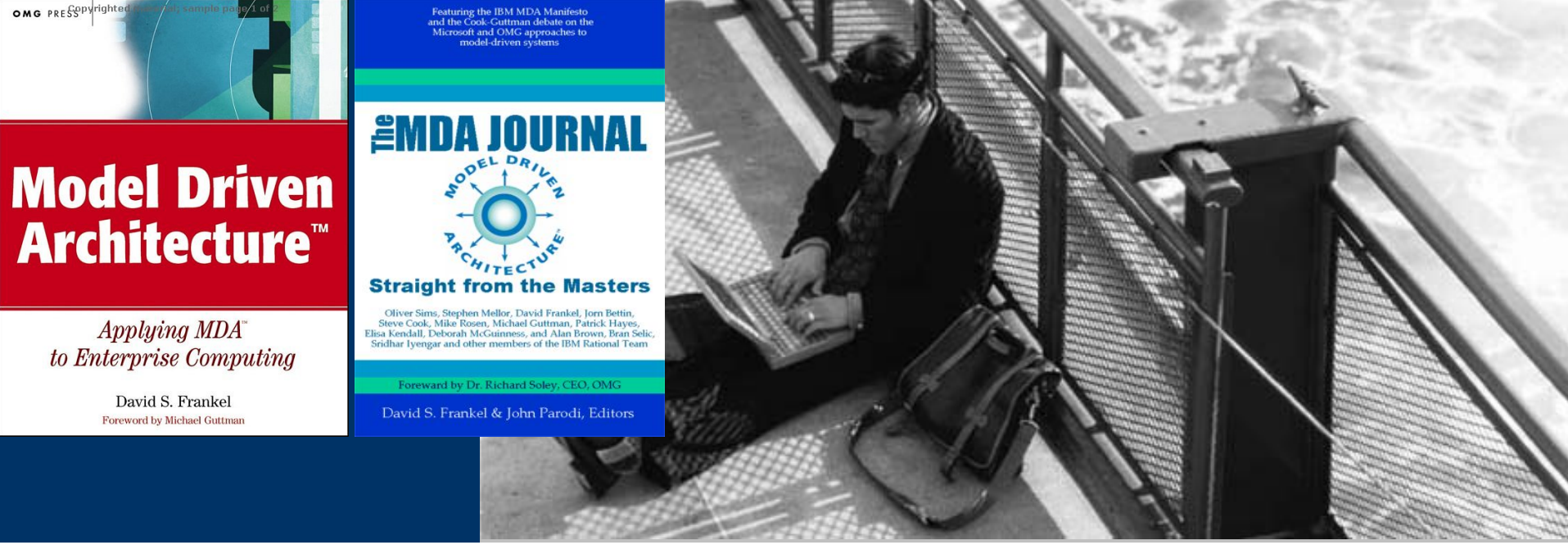


An Architectural Overview of MDA®

David S. Frankel
Lead Standards Architect – Model Driven Systems
SAP Labs

Portions adapted from the books

Model Driven Architecture: Applying MDA to Enterprise Computing

The MDA Journal: Model Driven Architecture Straight from the Masters

David S. Frankel

Agenda

- Value Chain Driven Business
- Industrializing Software
- Model-Driven Enterprise Architecture
- Informal vs. Formal Modeling
- Business Process Management
- Metadata Fragmentation
- Metadata Integration via MOF
- XMI and JMI
- UML Profiling
- PIMs and PSMs
- Model Driven Data Transformations
- The Future of MDA: Model-Driven Business Process Platforms

Agenda

- Value Chain Driven Business
- Industrializing Software
- Model-Driven Enterprise Architecture
- Informal vs. Formal Modeling
- Business Process Management
- Metadata Fragmentation
- Metadata Integration via MOF
- XMI and JMI
- UML Profiling
- PIMs and PSMs
- Model Driven Data Transformations
- The Future of MDA: Model-Driven Business Process Platforms

Value Chains Overview

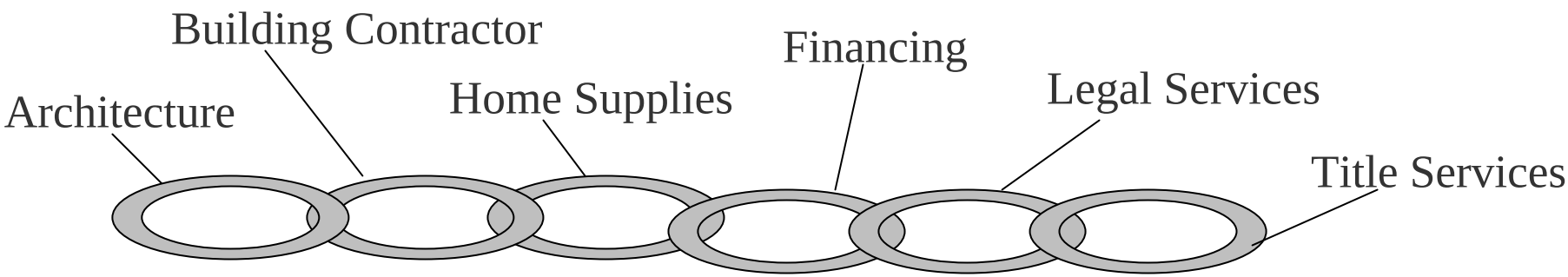
Focus on core competencies

- Rely on other organizations to provide necessary products and services outside the core

Kinds of value chains

- *Supply Chains*: Links a producer with suppliers.
- *One-Stop Shopping Chains*: Provides a customer with one address for the purchase of a set of related products and services
- *Software Development Chains*: Distributes software design, development, deployment, and management responsibilities among multiple organizations.

One-Stop Shopping Chains



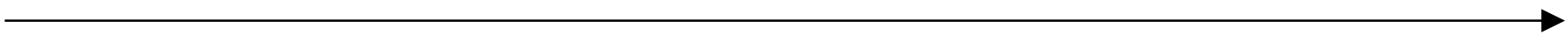
Increased Complexity Facing IT

Uniform User Devices

Fat Clients | Web Clients | Wireless Handhelds | Telephone Keypads

Support Business Functions Within the Enterprise

Support Business Functions Within the Enterprise, C2B, and B2B



Complexity

Defining and managing links in value chains, and lower-granularity services

Building, updating, and integrating complex, distributed, secure service-oriented systems is difficult

- **Labor-intensive**
- **Easy to use a good application server in an unscalable fashion**
- **Scalability and security require consistent use of architectural patterns**
- **Variability in link technology is a security vulnerability**

Agenda

- Value Chain Driven Business
- Industrializing Software
- Model-Driven Enterprise Architecture
- Informal vs. Formal Modeling
- Business Process Management
- Metadata Fragmentation
- Metadata Integration via MOF
- XMI and JMI
- UML Profiling
- PIMs and PSMs
- Model Driven Data Transformations
- The Future of MDA: Model-Driven Business Process Platforms

Applying Industrial Lessons Where Appropriate

Basic industrial principles to achieve efficiencies and automation

- **Formal blueprints (models)**
- **Components**
- **Patterns**

Crawl, walk, run…a gradual change

Take advantage of things that are unique about software

- **Rapid prototyping capabilities**

Build on what we already know about software development

Bringing Model-Centrism to Intermediate Tiers, EAI and B2Bi

Part of general trend to raise the abstraction level

Models as development artifacts

- **Not simply blueprints for humans**

Already well-established for front and back ends

- **WYSIWYG GUI modeling and data modeling**
- **Hand coding no longer predominates**
- **But tuning allowed**

SOA requires intermediate tiers

Wizards vs. models

Component-Based Development

- **Interchangeable components and scientific management were the keys to the industrial revolution**
- **More than objects: Independently deployable**
- **Excellent source: *Business Component Factory*, by Peter Herzum and Oliver Sims**
- **Crucial to Service Oriented Architecture**

Design Patterns

Patterns at the technical level

- **Such as *Java Blueprints***
- **Best practices for implementing components or a set of interacting components**

Some patterns make sense at the level of business semantics

- **Such as the *Observer* pattern (Gamma et al)**

Automatic Pattern Replication

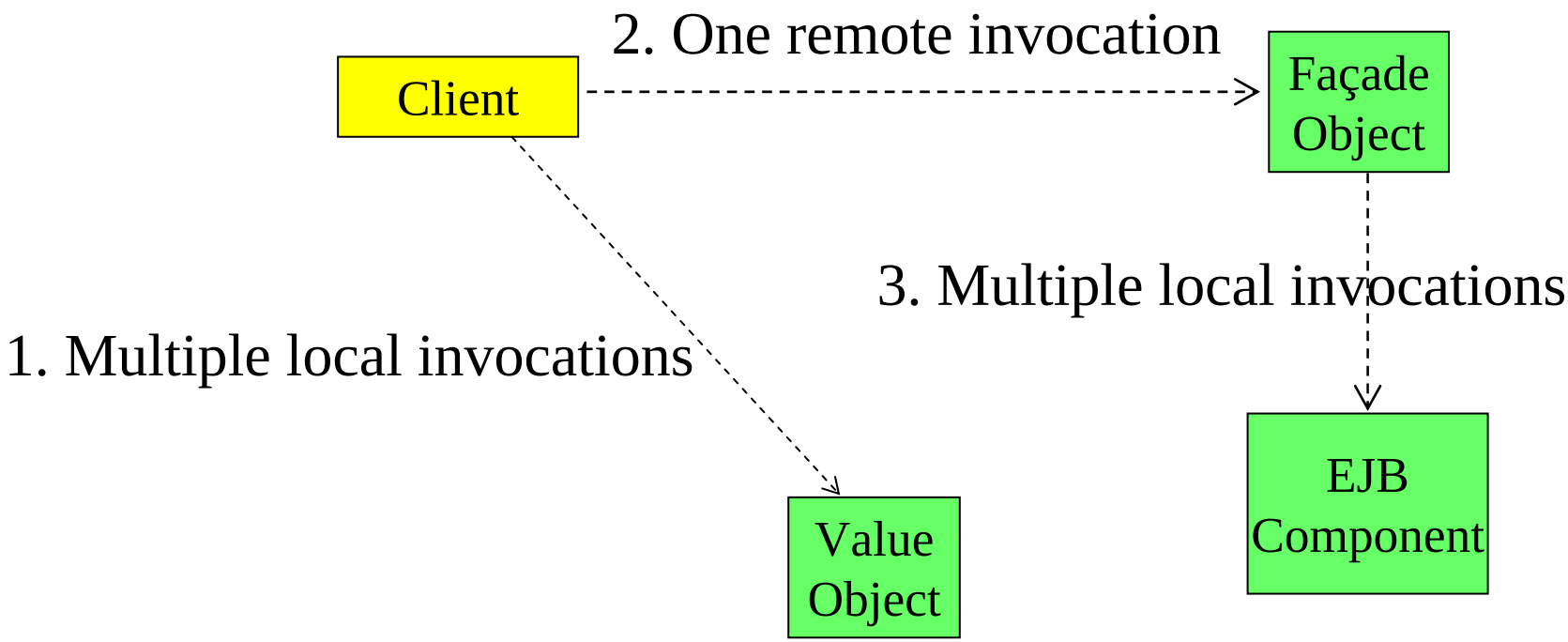
MDA generators encapsulate pattern knowledge

- **And apply patterns automatically**
- **Technical patterns are the most amenable**
- **Repetitive hand-coding of each pattern instance is inefficient**
- **Patterns community is coming around to this view**
  - **e.g. John Crupi**

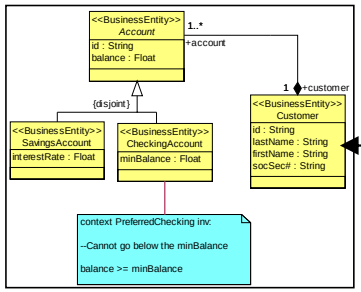
Generators can enforce large scale patterns or *architectural styles*

- **Richard Hubert, *Convergent Architecture***

Using Value Object Design Pattern to Set Attributes



A Generator Applying the Value Object Pattern



Model of Customer Entity

Generator

Façade Remote Interface

Façade Class

Value Object Class

Bean Interrface

Bean Class

= completely generated

= partially generated

Other Advances Toward Efficiency

Middleware

- Raises the abstraction level of the platform

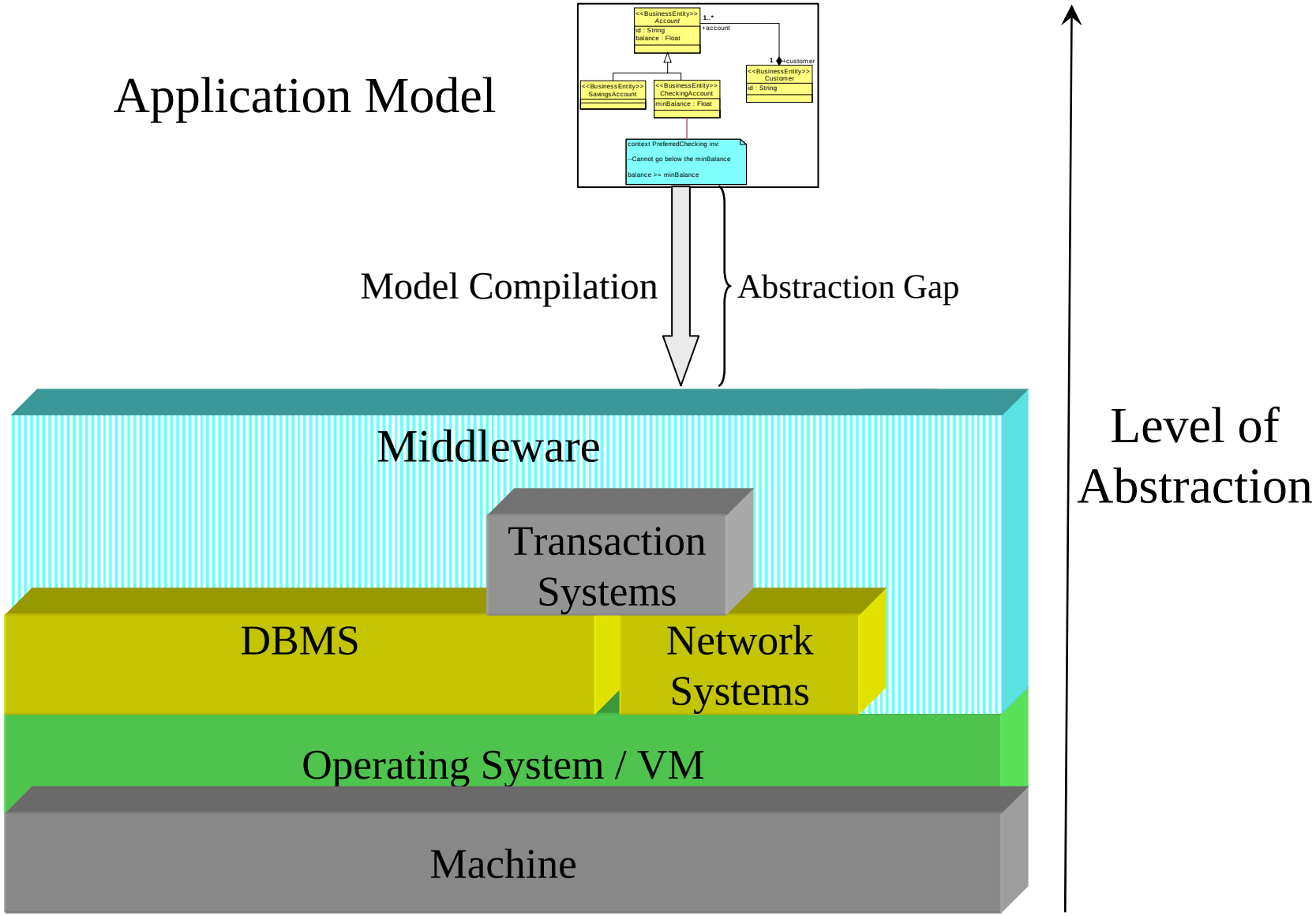
Declarative Specification

- e.g. setting transaction properties in component descriptors

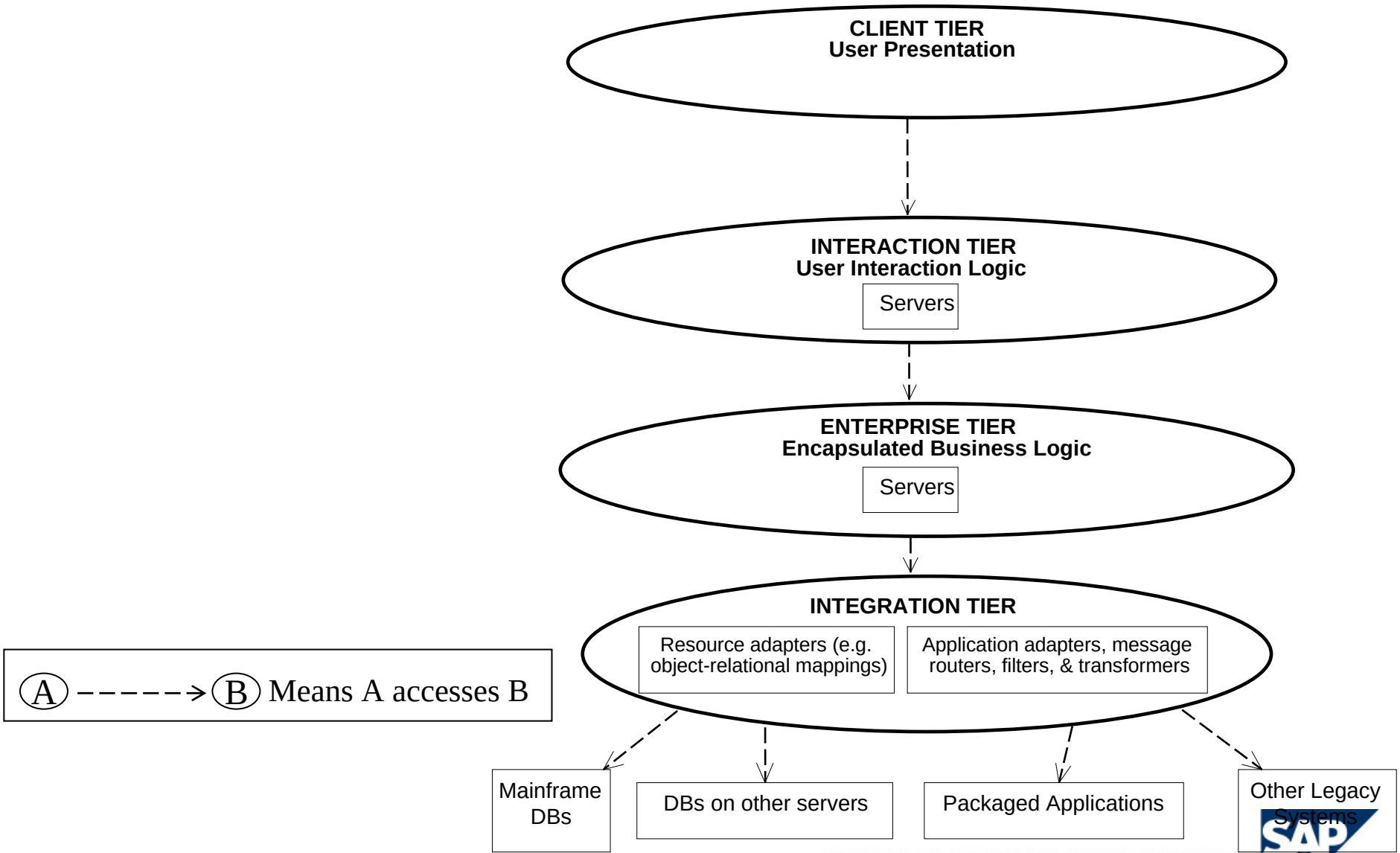
Enterprise Architecture

- Separation of concerns

Model Compilers and the Abstraction Level



Multi-Tiered Enterprise Architecture With EAI Adapters & Message Management



Architectural Separation
Application Viewpoint vs. Infrastructure

Applications

Middleware Service APIs

Middleware Configuration Languages e.g. EJB Deployment Descriptor

3GL Languages
- Editors
- Debuggers
- Programmatic and user interfaces to compilers

"Above the Line"

Water Line[1]

"Below the Line"

Middleware Implementations

3GL Languages Compiler Implementations

[1]The "above and below the line" concept was developed by Oliver Sims

Agenda

- Value Chain Driven Business
- Industrializing Software
- Model-Driven Enterprise Architecture
- Informal vs. Formal Modeling
- Business Process Management
- Metadata Fragmentation
- Metadata Integration via MOF
- XMI and JMI
- UML Profiling
- PIMs and PSMs
- Model Driven Data Transformations
- The Future of MDA: Model-Driven Business Process Platforms

Model-Driven Enterprise Architecture

UML® "out of the box" does not support modeling enterprise-centric computing

- **Tiers**
- **Middleware layers**
- **Distributed components**
- **Security**

A model-driven enterprise architecture requires modeling languages to support it

- **Distinct but coordinated**
- **For different system aspects and levels of abstraction**
  - **Different cells in the Zachman grid**
  - **Different RM-ODP viewpoints**
- **Use UML profiles and MOF to define the languages**

One Modeling Language?

Abstractions (Columns) ←→

Perspectives (Rows) ↕

The Zachman Framework	**DATA** *What (Things)*	**FUNCTION** *How (Process)*	**NETWORK** *Where (Location)*	**PEOPLE** *Who (People)*	**TIME** *When (Time)*	**MOTIVATION** *Why (Motivation)*
SCOPE (Contextual) *Planner*	List of things important to the business ***Entity = Class of business thing***	List of processes the business performs ***Function = Class of business process***	List of Locations in which the business operates ***Note = Major business location***	List of Organizations Important to the Business ***People = Major organizations***	List of Events Significant to the Business ***Time = Major business event***	List of Business Goals/Strategies ***Ends/Means = Major bus. goal/Critical success factor***
BUSINESS MODEL (Conceptual) *Owner*	Semantic Model ***Ent = Business entity*** ***Rein = Business relationship***	Business Process Model ***Proc = Business process*** ***I/O = Business resources***	Business Logistics System ***Node = Business location*** ***Link = Business linkage***	Work Flow Model ***People = Organization unit*** ***Work = Work product***	Master Schedule ***Time = Business event*** ***Cycle = Business cycle***	Business Plan ***End = Business objective*** ***Means = Business strategy***
SYSTEM MODEL (Logical) *Designer*	Logical Data Model ***Ent = Data entity*** ***Rein = Data relationship***	Application Architecture ***Proc = Application function*** ***I/O = User views***	Distributed System Architecture ***Node = I/S function (Processor, Storage, etc.)*** ***Link = Line characteristics***	Human Interface Architecture ***People = Role*** ***Work = Deliverable***	Processing Structure ***Time = System event*** ***Cycle = Processing cycle***	Business Rule Model ***End = Structural assertion*** ***Means = Action assertion***
TECHNOLOGY MODEL (Physical) *Builder*	Physical Data Model ***Ent = Segment/Table, etc.*** ***Rein = Pointer/Key***	System Design ***Proc = Computer function*** ***I/O = Data elements/ sets***	Technology Architecture ***Node = Hardware/ System software*** ***Link = Line specifications***	Presentation Architecture ***People = User*** ***Work = Screen format***	Control Structure ***Time = Execute*** ***Cycle = Component cycle***	Rule Design ***End = Condition*** ***Means = Action***
DETAILED REPRESENTATIONS (Out-of-Context) *Sub-Contractor*	Data Definition ***Ent = Filed*** ***Rein = Address***	Program ***Proc = Language statement*** ***I/O = Control block***	Network Architecture ***Node = Addresses*** ***Link = Protocols***	Security Architecture ***People = Identity*** ***Work = Job***	Timing Definition ***Time = Interrupt*** ***Cycle = Machine cycle***	Rule Specification ***End = Sub-condition*** ***Means = Step***
FUNCTIONING ENTERPRISE	Actual Business Data	Actual Application Code	Actual Physical Networks	Actual Business Organization	Acutal Business Schedule	Actual Business Strategy

Applications

Modeling Languages
- Editors (e.g. UML modeling tools)
- Programmatic and user interfaces to generators

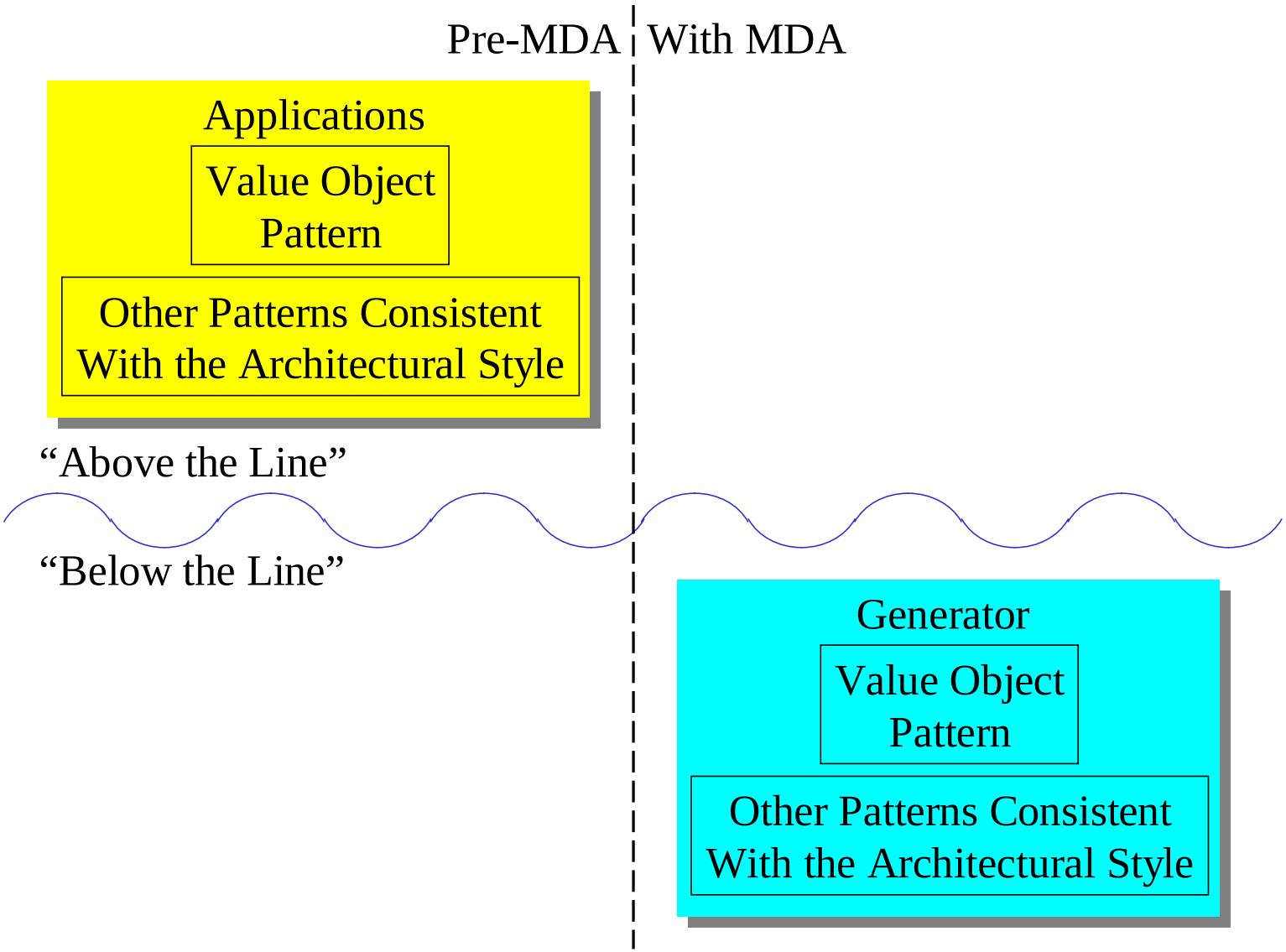
Water Line

Modeling Language Definitions (language creator's Viewpoint)

Mappings of languages to Technologies (including application of patterns)

Generator Implementations (Generators conform to mappings)

Pushing Pattern Knowledge Below the Line



Model-Driven Development vs. Model Driven Architecture

MDA includes model-driven development

Also about model-driven deployment

- **Currently deployment tools metadata is fragmented**
  - **Little standardization**

Also about model-driven management (ops)

- **Generating instrumentation from models of service-level agreements (SLAs)**
- **Java Management Specification (JSR-77) provides some standardization**

Agenda

- Value Chain Driven Business
- Industrializing Software
- Model-Driven Enterprise Architecture
- Informal vs. Formal Modeling
- Business Process Management
- Metadata Fragmentation
- Metadata Integration via MOF
- XMI and JMI
- UML Profiling
- PIMs and PSMs
- Model Driven Data Transformations
- The Future of MDA: Model-Driven Business Process Platforms

Informal Models

- **Informal modeling**
- **Used to sketch out basic concepts**
- **Advantage over typical box and line diagrams because shapes and line types have specific meanings**
- **Important, but can't drive code generators and dynamic execution engines**
  - **Analogously, informal text can't be compiled and executed like 3GL text**

Formal Models

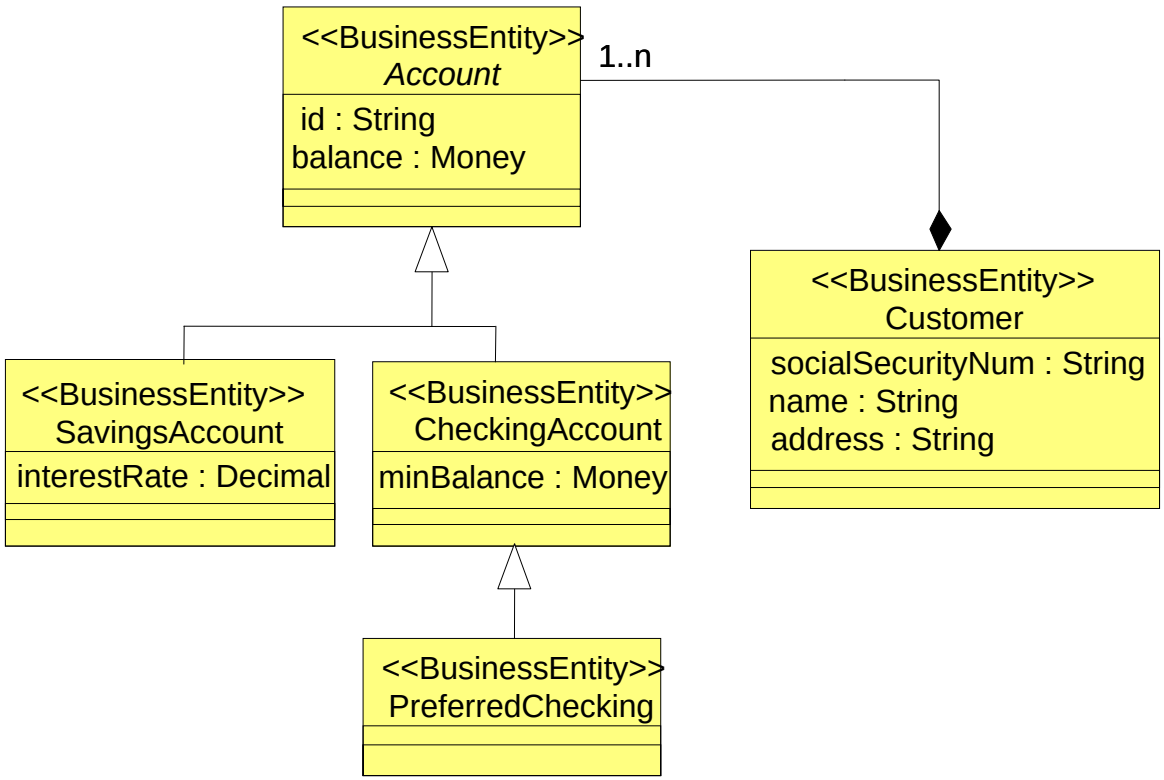
Precise

- Precision and detail are not the same!

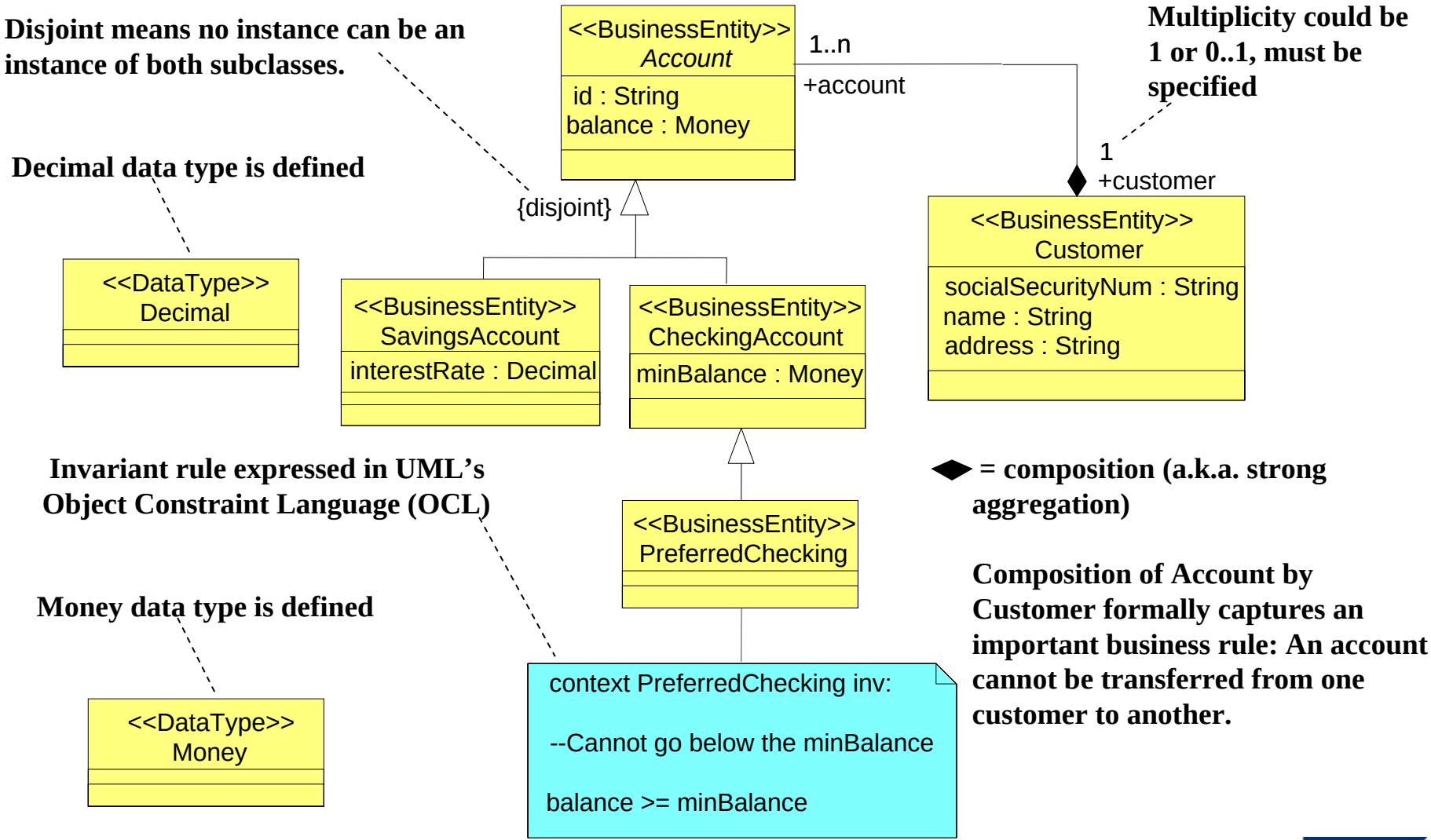
Computationally complete

- Missing properties and unresolved references not acceptable
- 3GL analogy…
  - an incomplete expression such as “a +” does not compile
  - An undeclared identifier does not compile

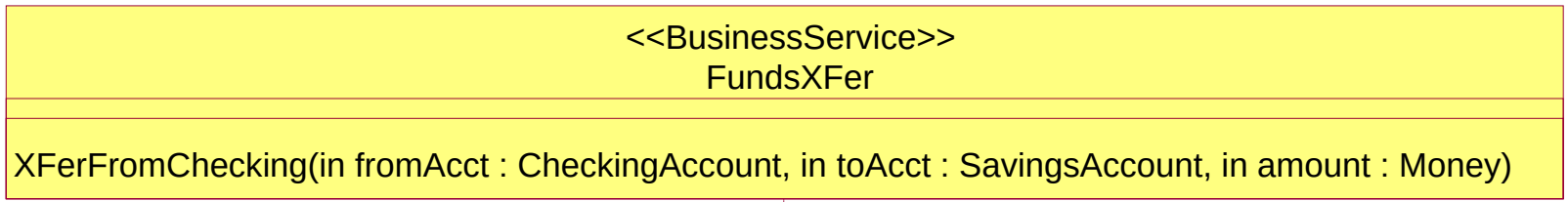
Business Information Model
Imprecise and Incomplete



Business Information Model
Precise and Complete



A Formal Model of an Abstract Business Service




```
context FundsXFer::XFerFromChecking (fromAcct : CheckingAccount, toAcct : SavingsAccount) : void
pre:
   --There must be sufficient funds in the checking account to support the transfer
   fromAcct.balance >= amount
pre:
   --The checking account and the savings account must belong to the same customer
   fromAccount.customer = toAccount.customer

post:
   --The balance of the checking account is reduced from its orginal amount by the amount of the transfer
   fromAcct.balance = fromAcct.balance@pre - amount
post:
   --The balance of the savings account is increased from its original amount by the amount of the transfer
   toAcct.balance = toAcct.balance@pre + amount
```

Contracts, Reuse, and Interoperability

"Connecting the dots"

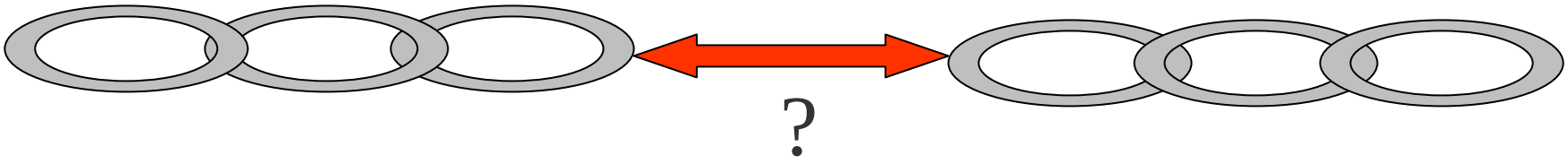
- **Makes the specification more complete**
- **Flushes out design flaws**

Interoperability among components is difficult when contract not well understood

Formal contract increases the degree of semantic interoperability

- **Regardless of whether code is generated from the contract**
- **Semantic interoperability required for B2Bi**

Provides a "gold standard" for people who speak different human languages



Intelligent Links in Value Chains

- **Formalized service level agreements (SLAs)**
- **Tools use formal SLAs to generate instrumentation that monitors compliance**
  - **Or use SLAs dynamically for compliance monitoring**
- **Bring intelligent links online more quickly and with more confidence**

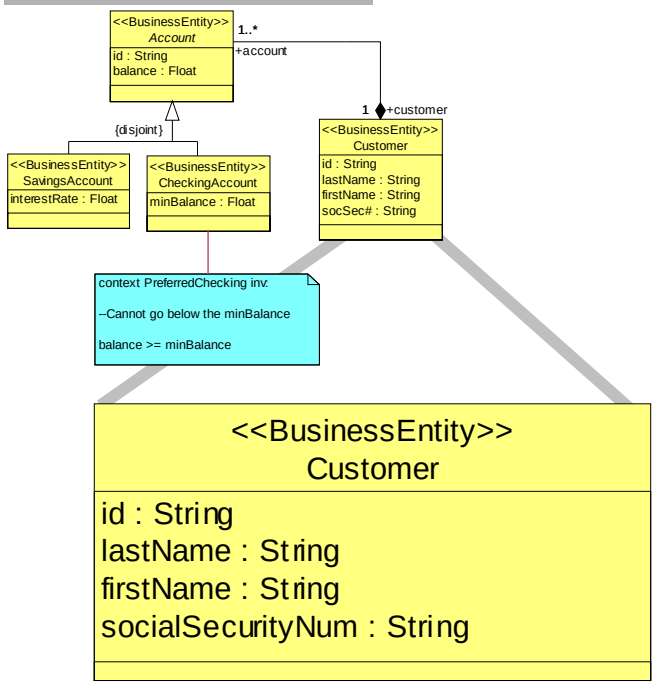
Mapping the Business Information Model to XML

Platform-Independent Model

Class Model -XML Mapping Rules

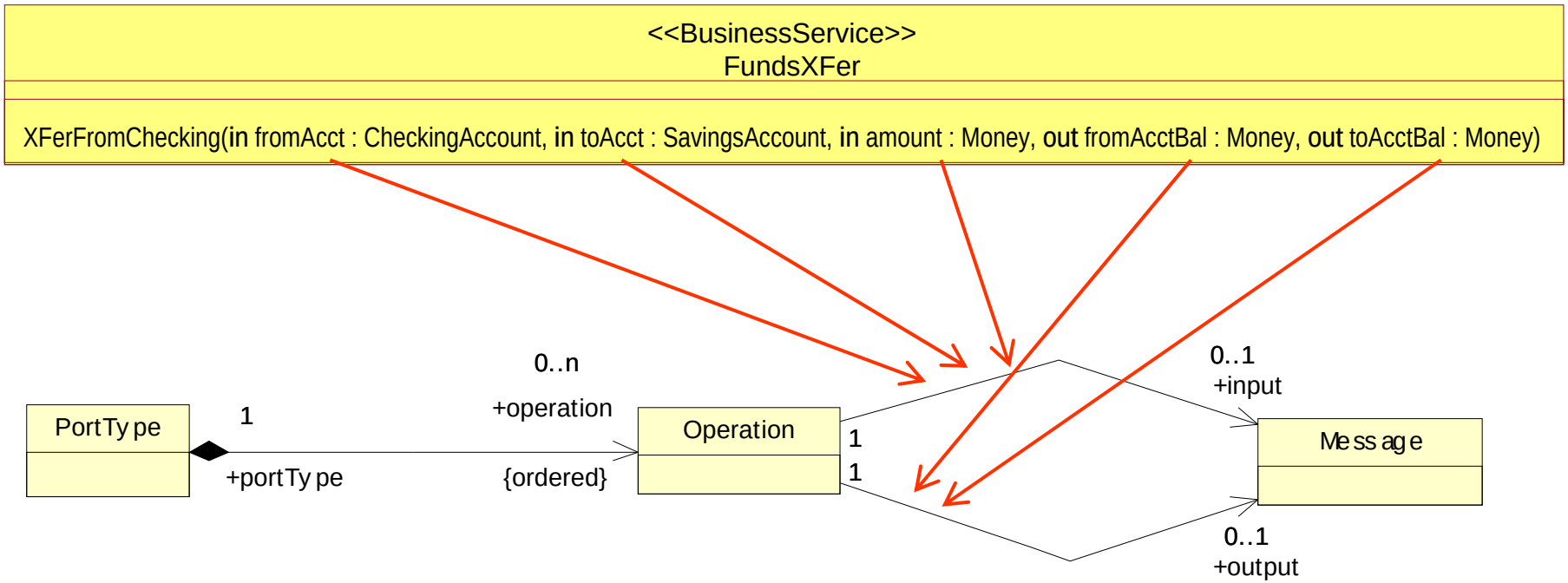
Produce

XML DTD (or Schema)

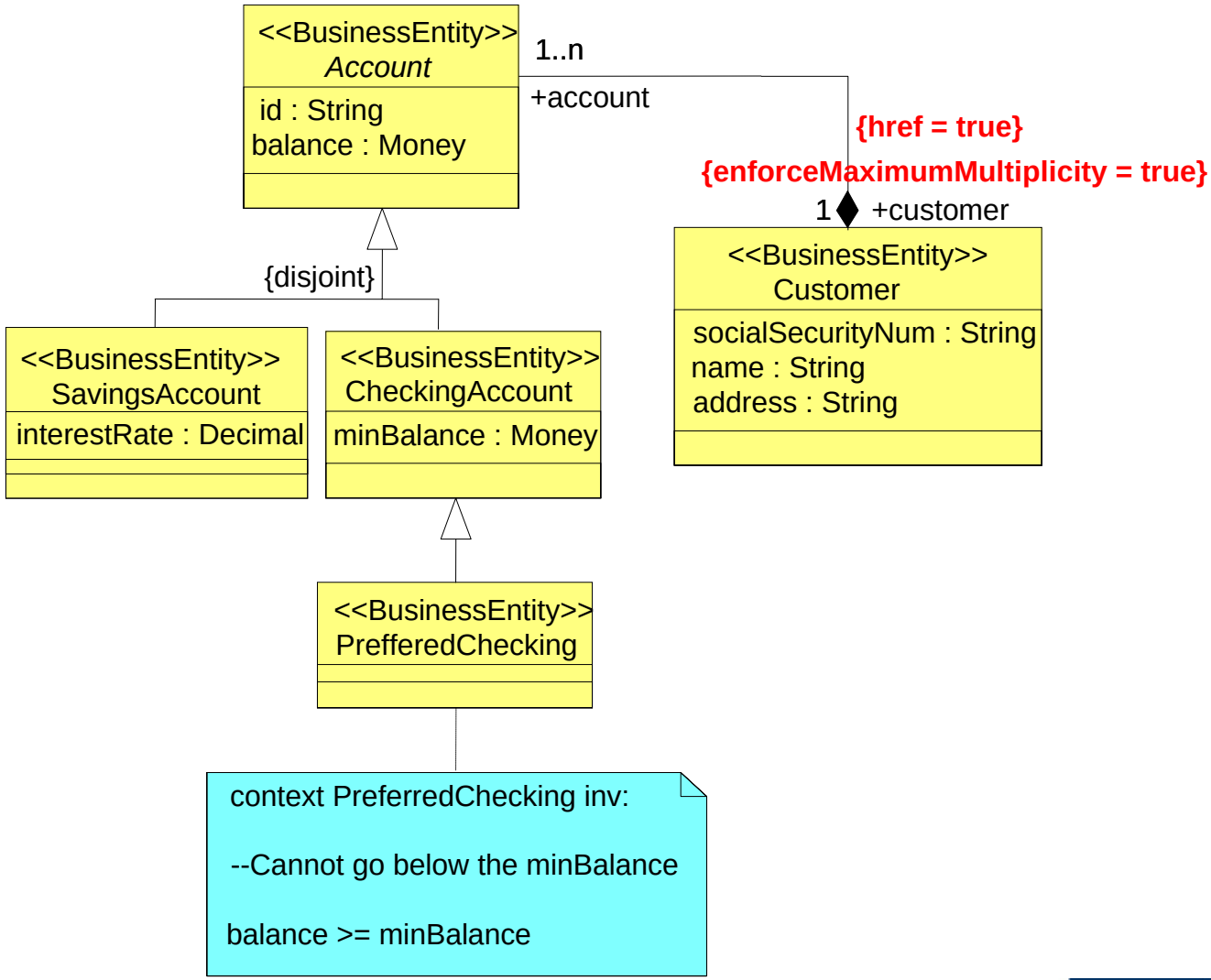



```
...
<!ELEMENT Bank.Customer.id (#PCDATA | XMI.reference)* >
<!ELEMENT Bank.Customer.lastName (#PCDATA | XMI.reference)* >
<!ELEMENT Bank.Customer.firstName (#PCDATA | XMI.reference)* >
<!ELEMENT Bank.Customer.socialSecurityNum (#PCDATA |
XMI.reference)*>
...
```

Mapping the Business Service Model to WSDL



Parameterized Mappings



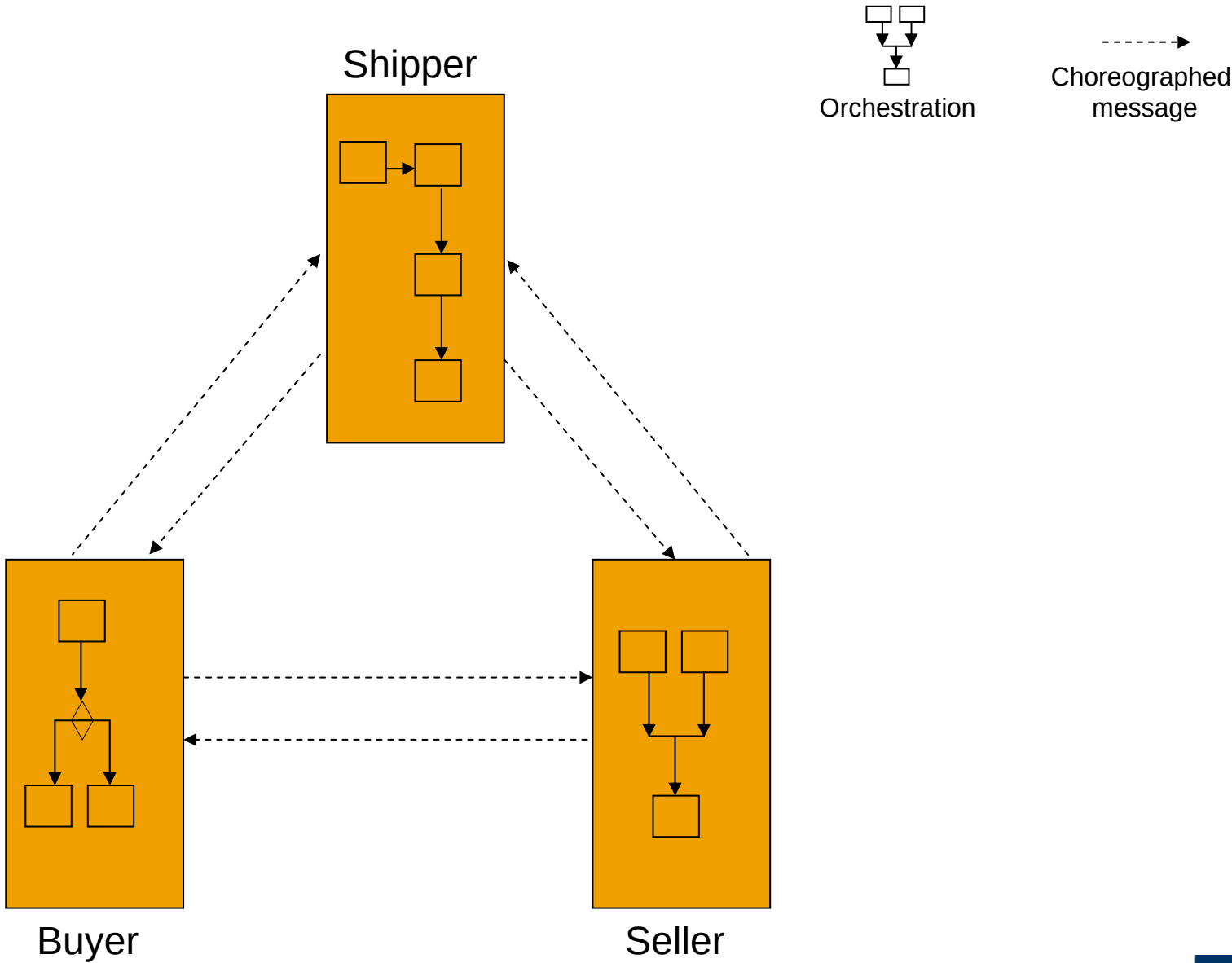
Agenda

- Value Chain Driven Business
- Industrializing Software
- Model-Driven Enterprise Architecture
- Informal vs. Formal Modeling
- Business Process Management
- Metadata Fragmentation
- Metadata Integration via MOF
- XMI and JMI
- UML Profiling
- PIMs and PSMs
- Model Driven Data Transformations
- The Future of MDA: Model-Driven Business Process Platforms

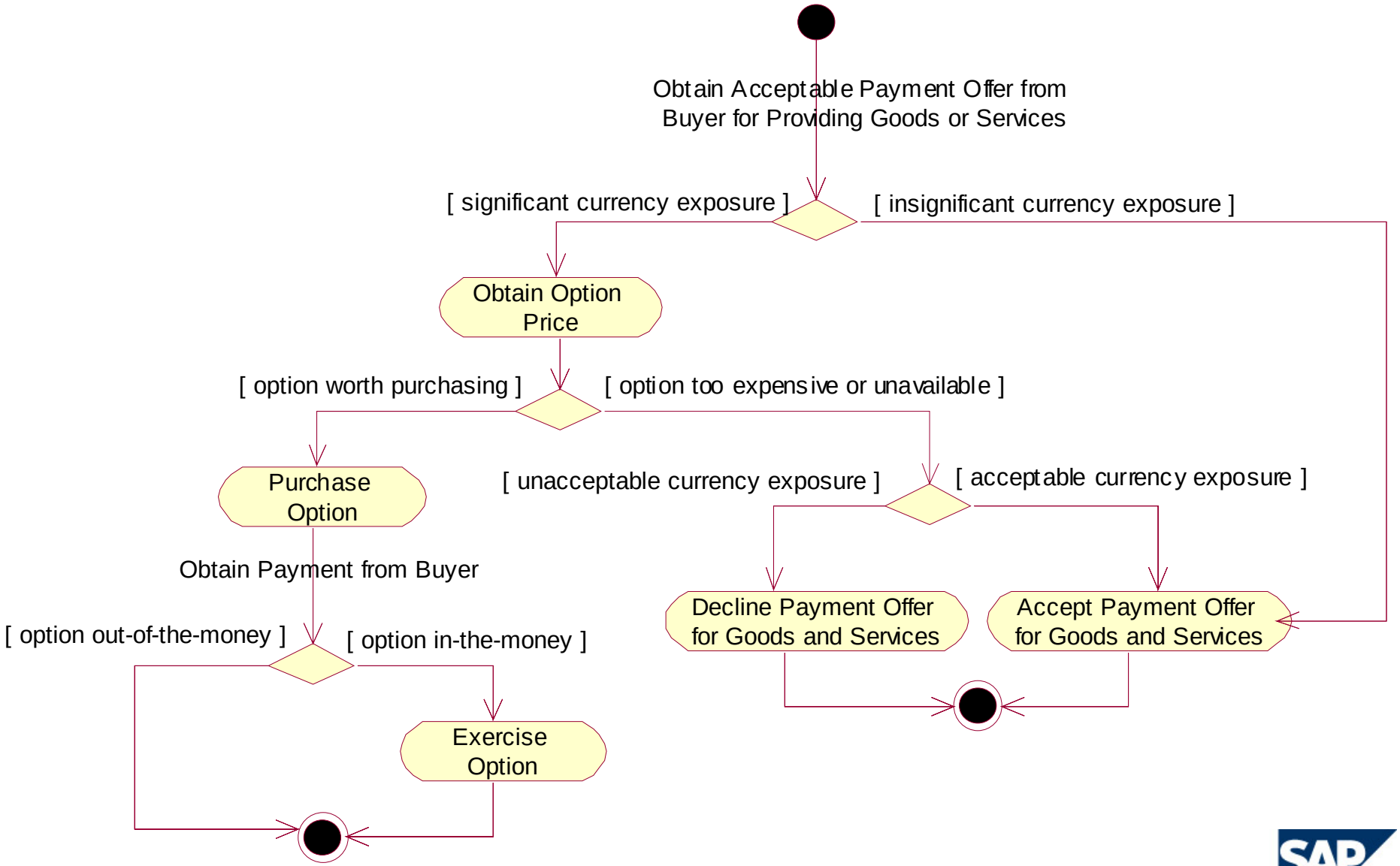
Collaborative Business Processes

- A link in a value chain may be more than just one service interface
- Need to model protocols involving communication back and forth
- Business process models
  - Steps in business processes trigger or are triggered by invocations of services across links

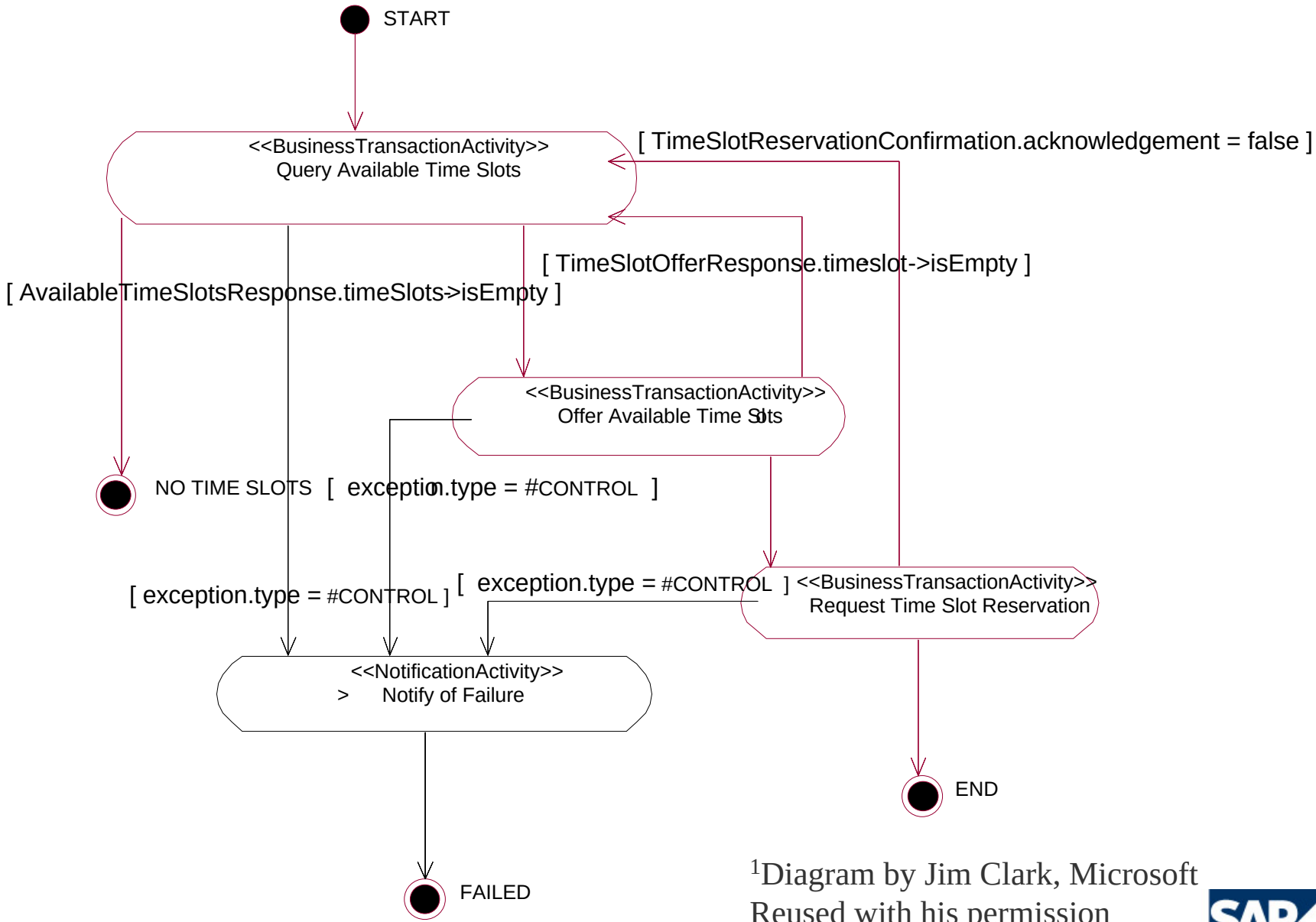
Choreography and Orchestration



Informal Business Process Model

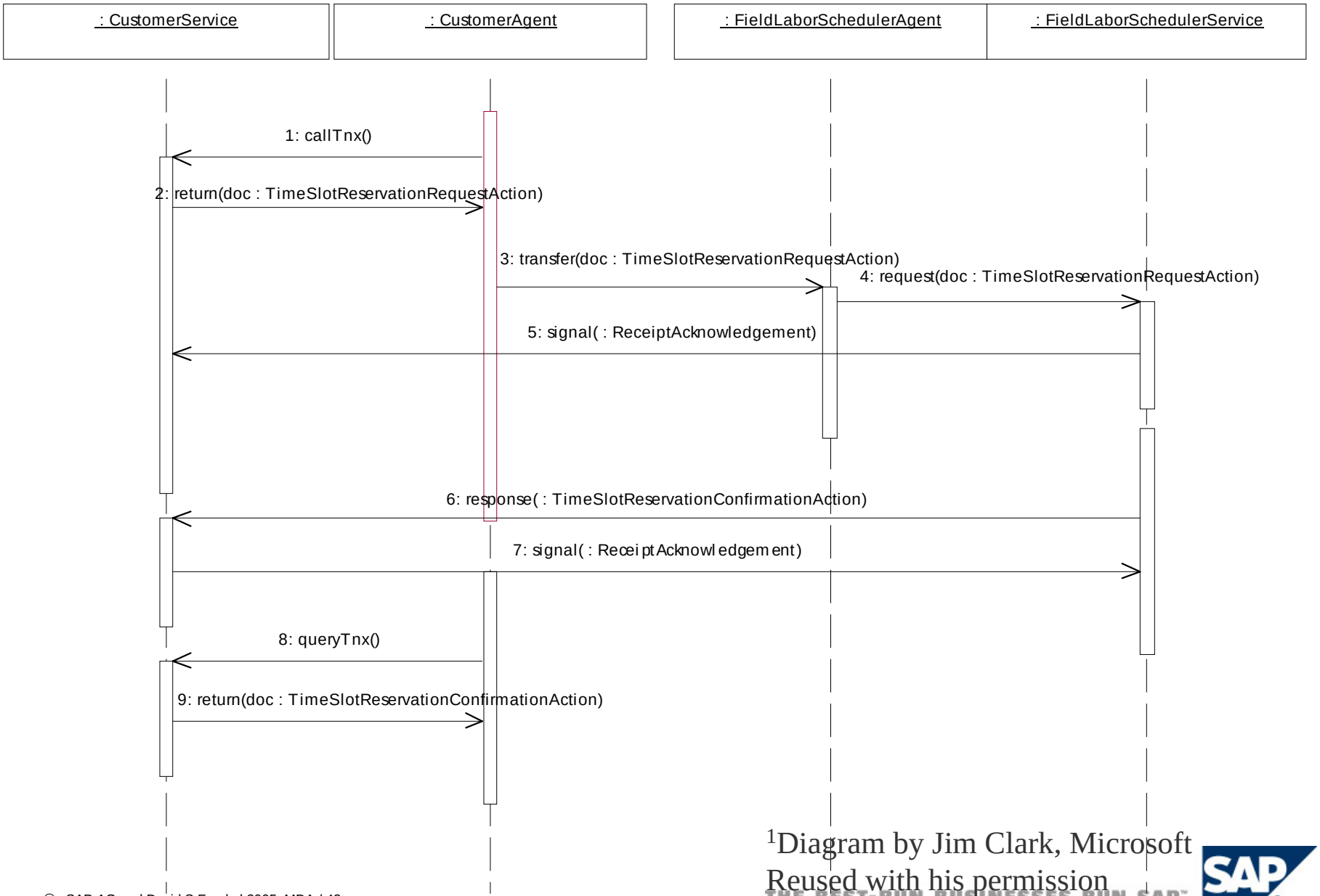


Formal Business Process Model[1]



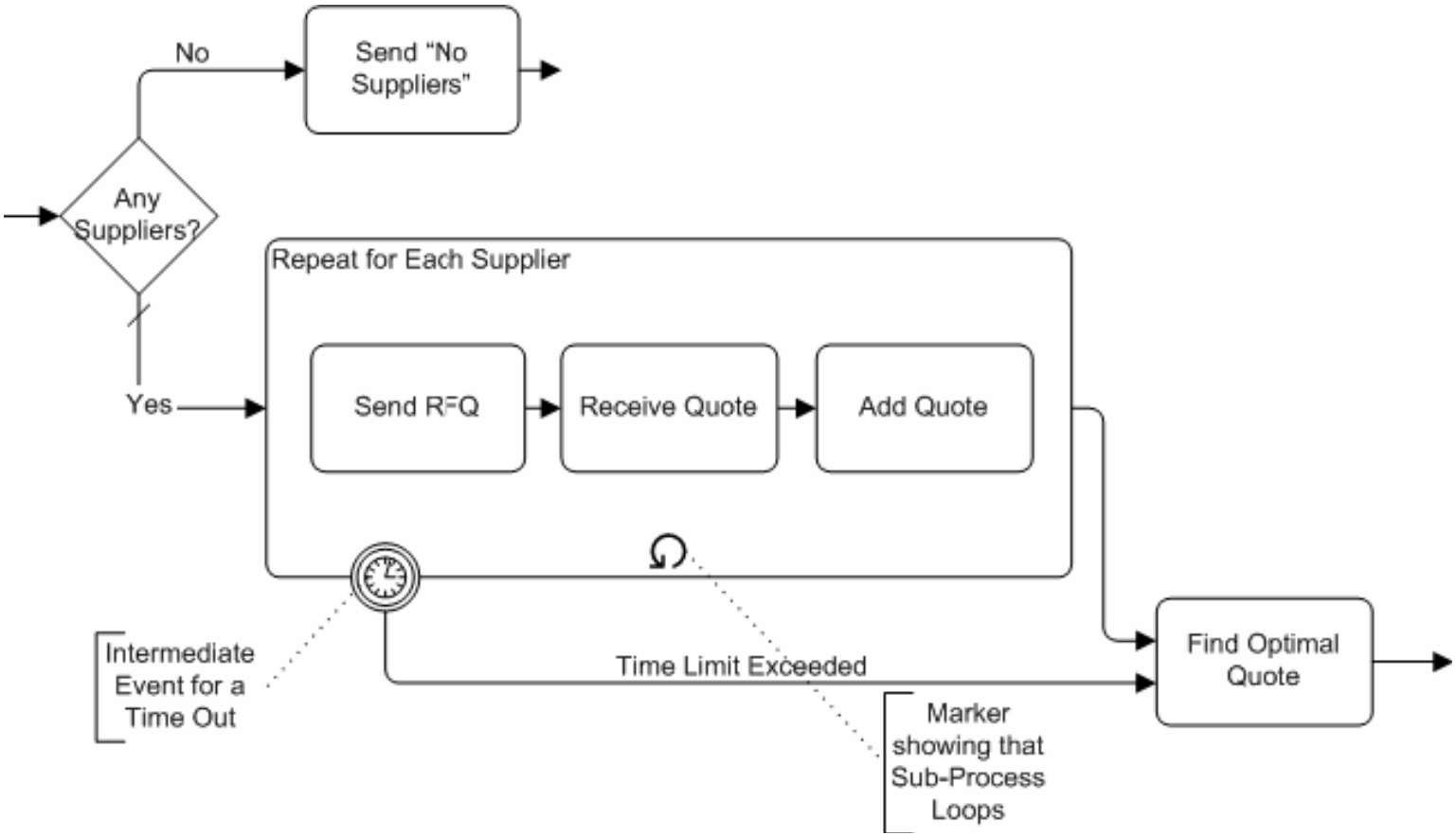
[1]Diagram by Jim Clark, Microsoft
Reused with his permission

Interaction Model[1]



[1]Diagram by Jim Clark, Microsoft
Reused with his permission

Business Process Modeling Notation (BPMN)



Example from "Introduction to BPMN" by Stephen White, IBM
Available at www.bpmi.org

The “M” in CA-BPM

Managing **business processes is the motivation**

Computer assistance is for this purpose

Not **the same as describing some process flows to do EAI**

- **Includes this but not motivated by it**

Industrial precedent

- **CAD models drive production machinery**

Agenda

- Value Chain Driven Business
- Industrializing Software
- Model-Driven Enterprise Architecture
- Informal vs. Formal Modeling
- Business Process Management
- Metadata Fragmentation
- Metadata Integration via MOF
- XMI and JMI
- UML Profiling
- PIMs and PSMs
- Model Driven Data Transformations
- The Future of MDA: Model-Driven Business Process Platforms

What is Metadata?

Originally, metadata meant only “data about data”

- **Database schema are distinct from the data itself**

Metadata now includes

- **UML models**
- **Data transformation rules**
- **APIs expressed in IDL, MIDL, C#, Java, WSDL, etc.**
- **Business process and workflow models**
- **Product configuration descriptors and tuning parameters**
- **Information that drives deployment tools**
- **Information that drives runtime management (ops)**
  - **Including Service Level Agreements**

Metadata Fragmentation

Example: One enterprise component may have several disparate forms of metadata

- **Platform-independent UML**
- **Java interfaces**
- **XML descriptors**
- **CORBA® IDL**
- **Object-relational mapping**

Reflection and Metadata Fragmentation

	Operation that client invokes	Metadata that the operation returns
CORBA	get_interface	InterfaceDef
COM	GetTypeInfo	ITypeInfo
Java	getDeclaredMethods	Method

Volume and Value

Volume is large

- Global 1000 companies have tens of thousands of columns in their data models
- New kinds of models coming on line

Value is increasing

- Metadata drives generators and dynamic execution engines
  - Has been true for some time (e.g. workflow, CORBA, COM) but MDA accelerates trend

Increasing amounts of increasingly valuable metadata

Previous Metadata Integration Attempts

Late 1980s and early 1990s

Diagnosis correct: Metadata disparity

Prescription: Have one grand metamodel

- **One kind of model**

Reason for failure: Different stakeholders have different viewpoints

- **And require different modeling constructs**

Agenda

- Value Chain Driven Business
- Industrializing Software
- Model-Driven Enterprise Architecture
- Informal vs. Formal Modeling
- Business Process Management
- Metadata Fragmentation
- Metadata Integration via MOF
- XMI and JMI
- UML Profiling
- PIMs and PSMs
- Model Driven Data Transformations
- The Future of MDA: Model-Driven Business Process Platforms

MOF Overview

OMG standard

- **Sister to UML**

The most fundamental MDA language definition mechanism

- **Even UML is defined via MOF**
- **UML profiles are the other mechanism**

Supports model-driven metadata management

The Basic Premises

There will be more than one modeling language

- **For different system aspects and levels of abstraction**

Different languages have different modeling constructs

- **For relational data modeling: *table*, *column*, *key*, etc.**
- **For workflow modeling: *activity*, *performer*, *split*, *join*, etc.**
- **For OO class modeling: *class*, *attribute*, *operation*, *association*, etc.**

A modest degree of commonality is achievable by using one language to *define* the different languages

- **For example, use same means to describe that…**
  - **a table owns its columns**
  - **a class owns its attributes and operations**
  - **a state machine its transitions**

Metamodels

A metamodel defines a language

A MOF-compliant metamodel consists of

- **Abstract syntax**
  - **Expressed formally via MOF metamodeling constructs**
- **Semantics**
  - **Defines the meaning of the abstract syntax**
  - **Expressed informally (today) via natural language (i.e. English)**

Platform Independence

MOF metamodels are platform independent, meaning independent of…

- **Information formatting technologies such as XML DTD and XML Schema**
- **3GLs and 4GLs such as Java, C++, C#, and Visual Basic**
- **Distributed component middleware, such as J2EE, CORBA, and.NET**
- **Messaging middleware such as WebSphere MQ Integrator (MQSeries) and MSMQ**

MOF technology mappings

- ***XMI*®: XML Metadata Interchange (MOF-XML mapping)**
- ***JMI* : Java Metadata Interface (MOF-Java mapping)**
- ***CMI* : CORBA Metadata Interface (MOF-CORBA mapping)**

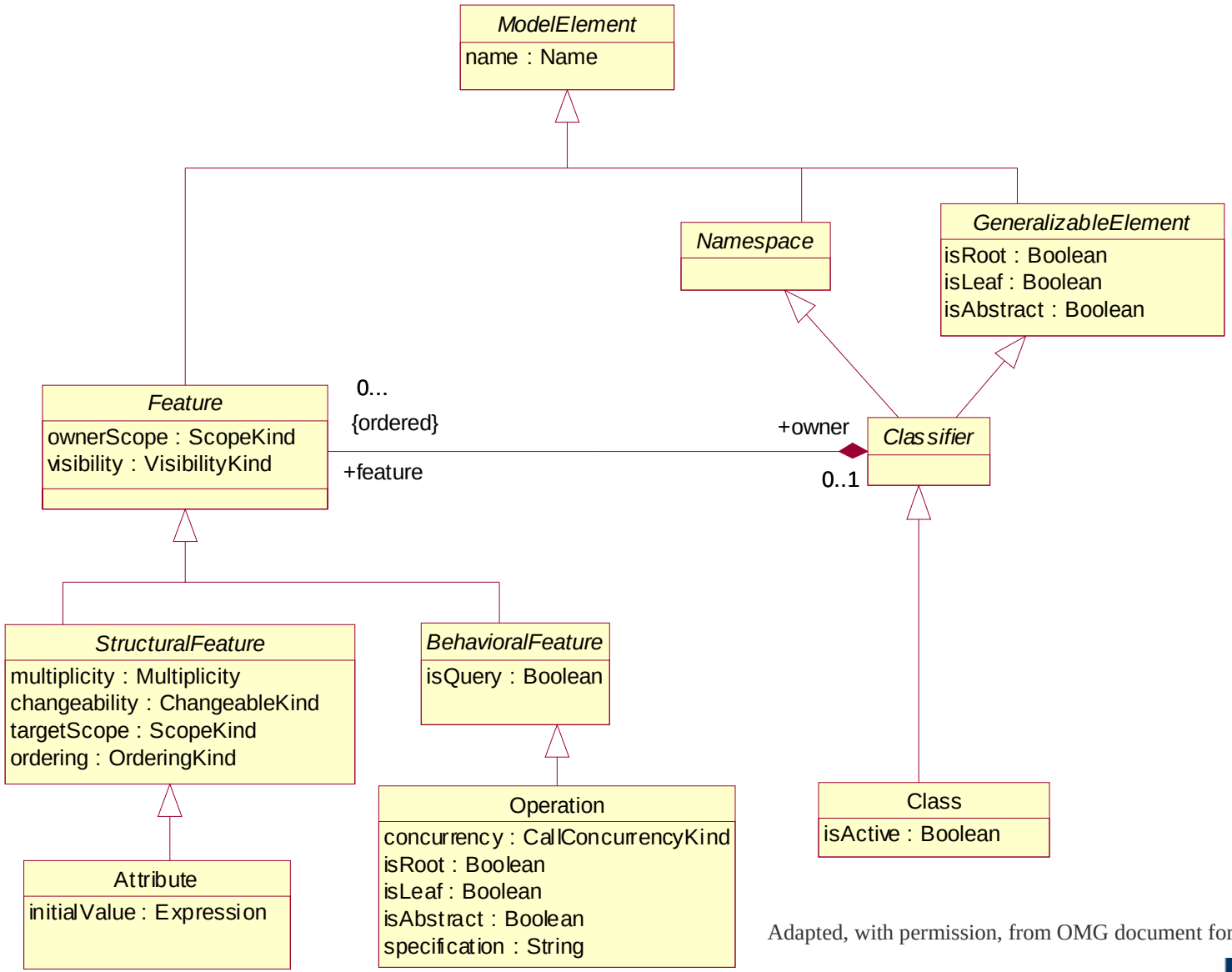
Borrowing from UML

MOF uses UML class modeling constructs

- **Including Object Constraint Language (OCL)**

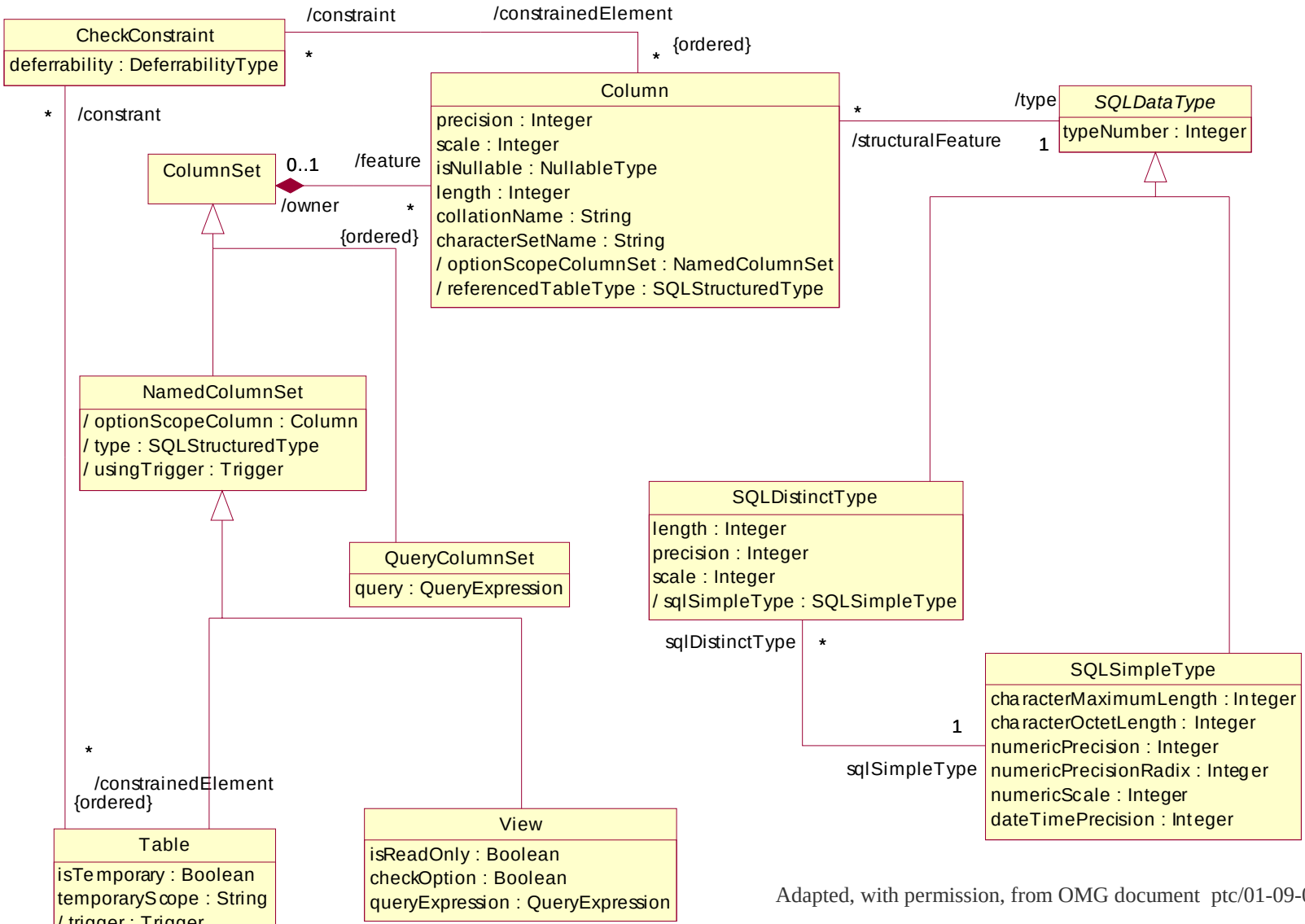
Uses these constructs as the common means for defining abstract syntax

Fragment of UML Metamodel for Class Modeling



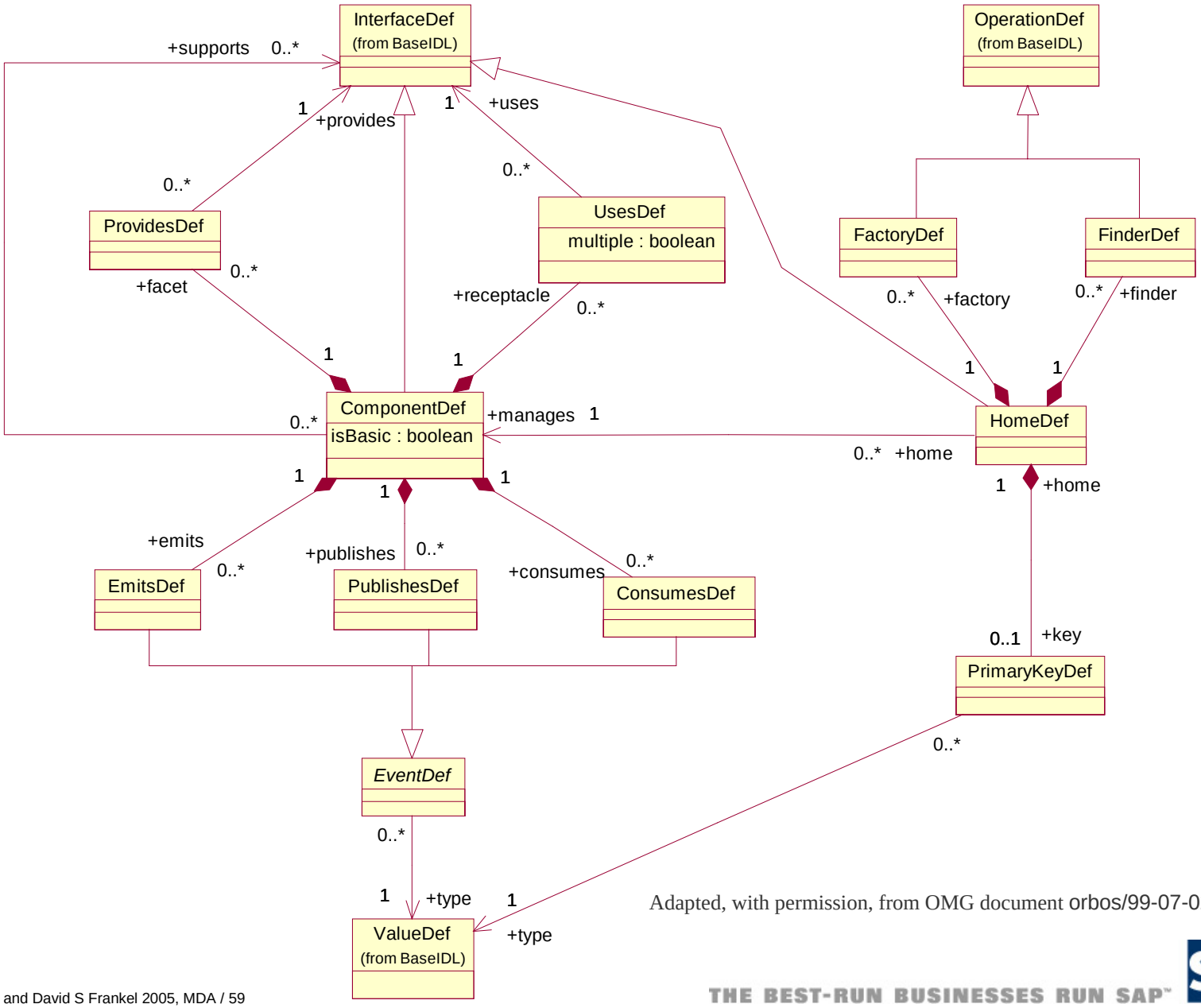
Adapted, with permission, from OMG document formal/01-09-67

Fragment of the CWM Relational Data Metamodel



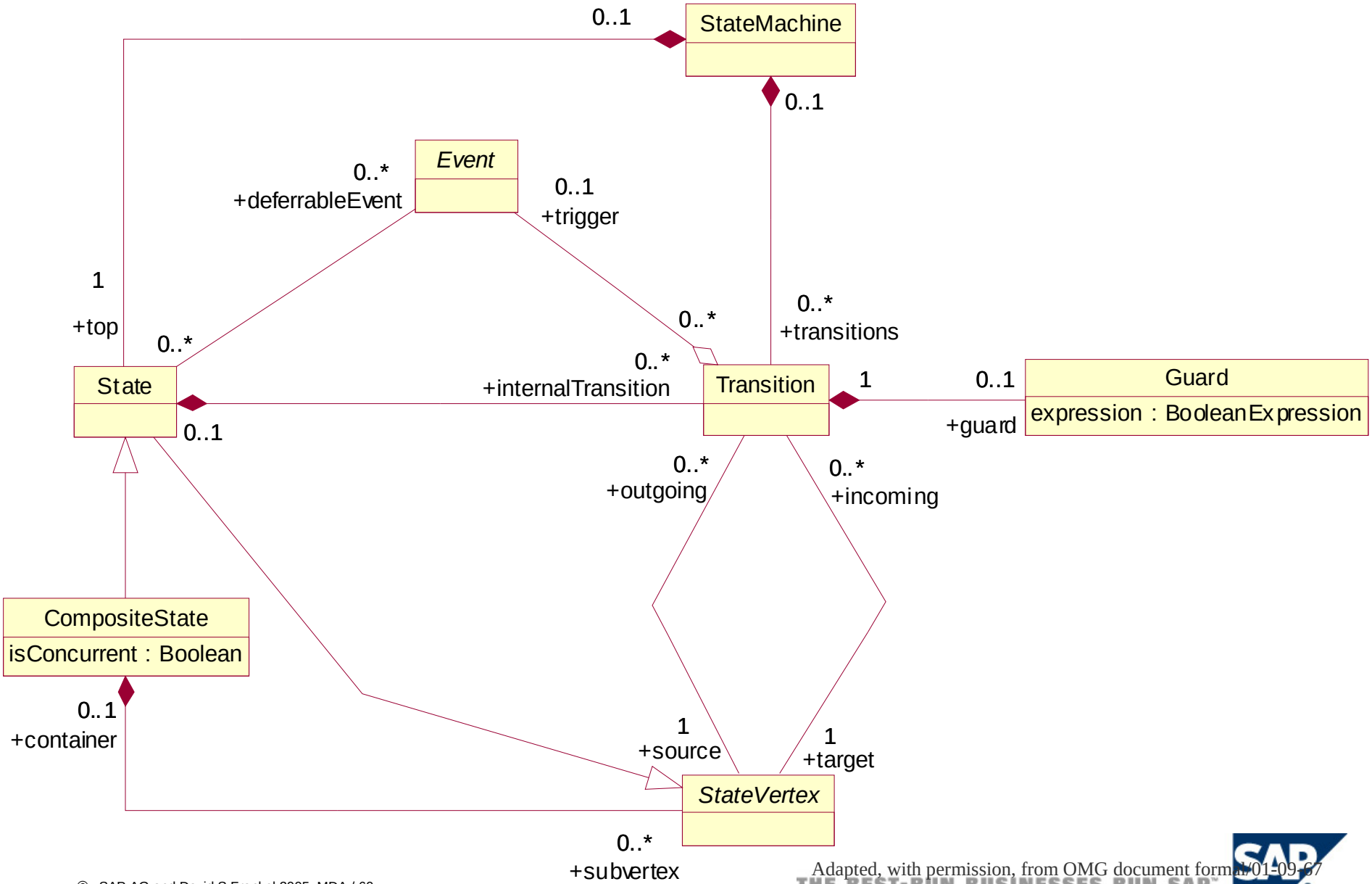
Adapted, with permission, from OMG document ptc/01-09-04

Fragment of the CORBA Component Metamodel



Adapted, with permission, from OMG document orbos/99-07-02

Fragment of the UML Metamodel for State Charts

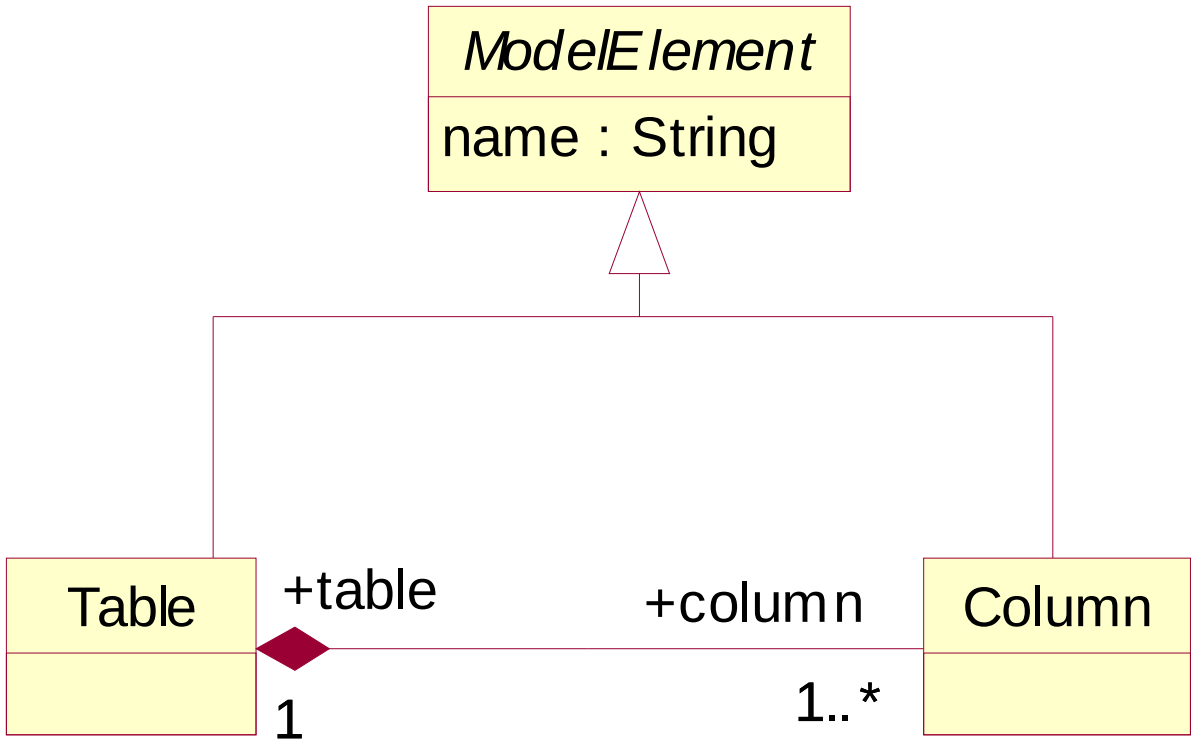


Adapted, with permission, from OMG document formal/01-09-67

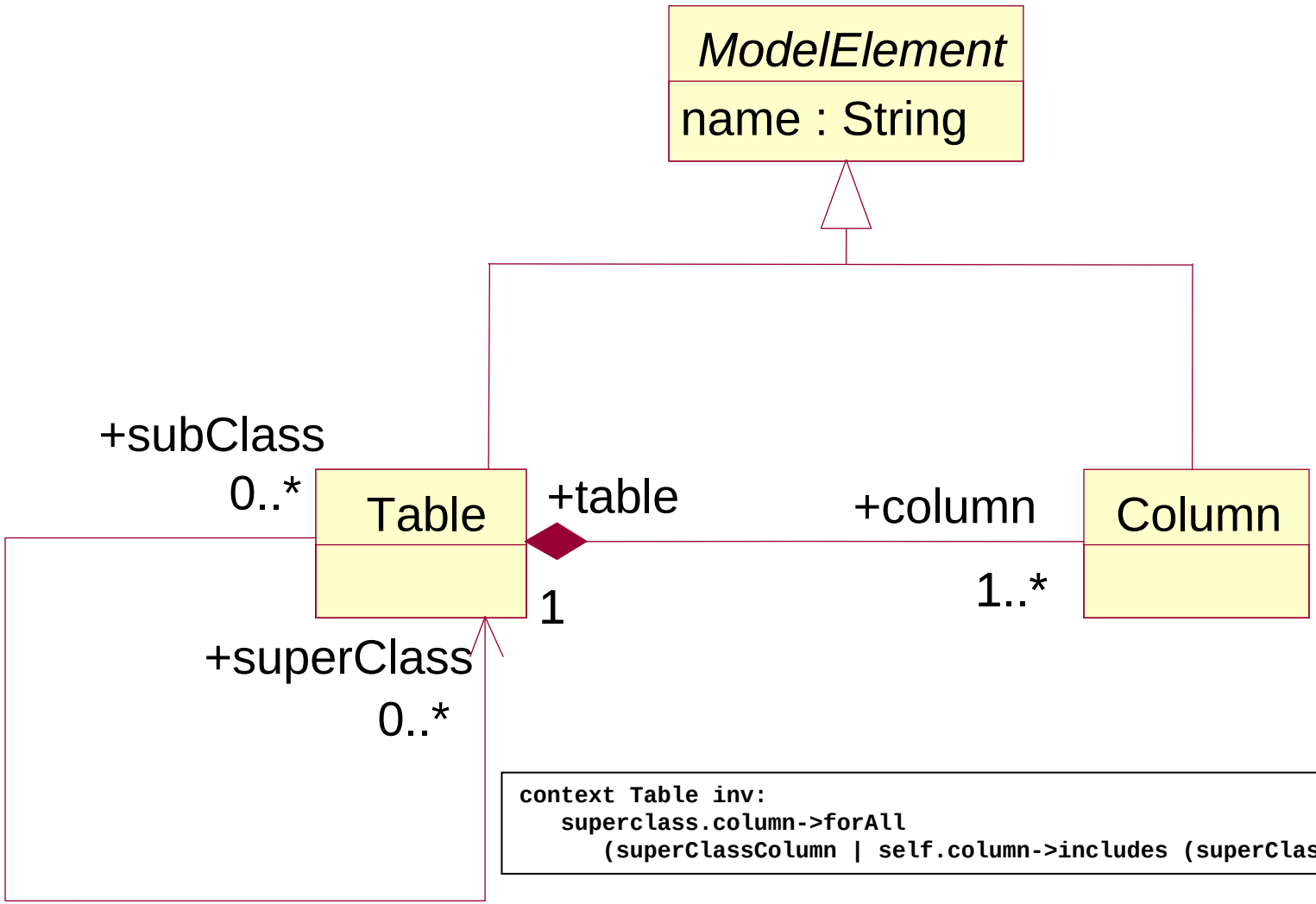
MOF is Not Just for OO Languages

- **MOF uses object-oriented modeling to define modeling constructs**
- **But the modeling constructs it defines need not be object-oriented**

Using MOF Subclassing to Define a Metamodel

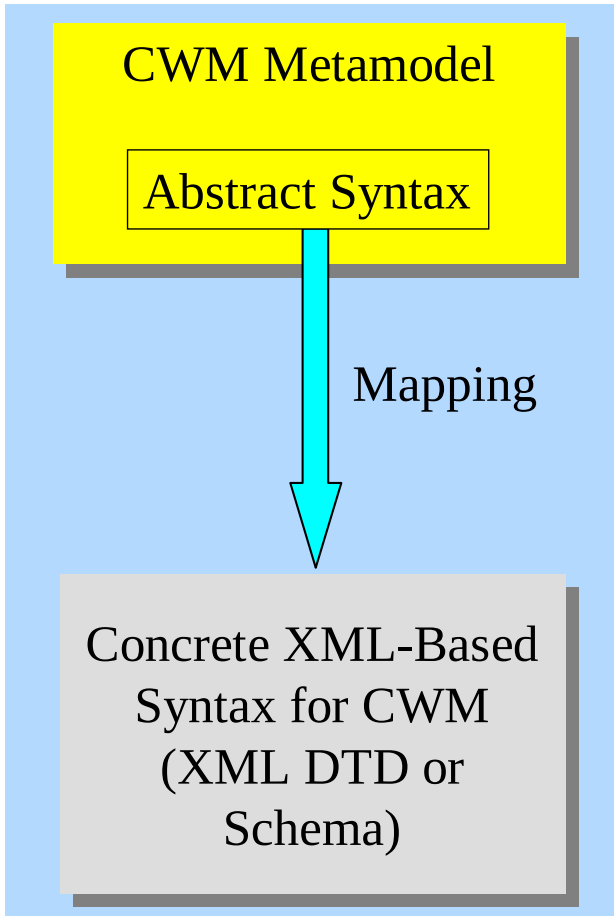


Using MOF to Define Subclassing in a Metamodel

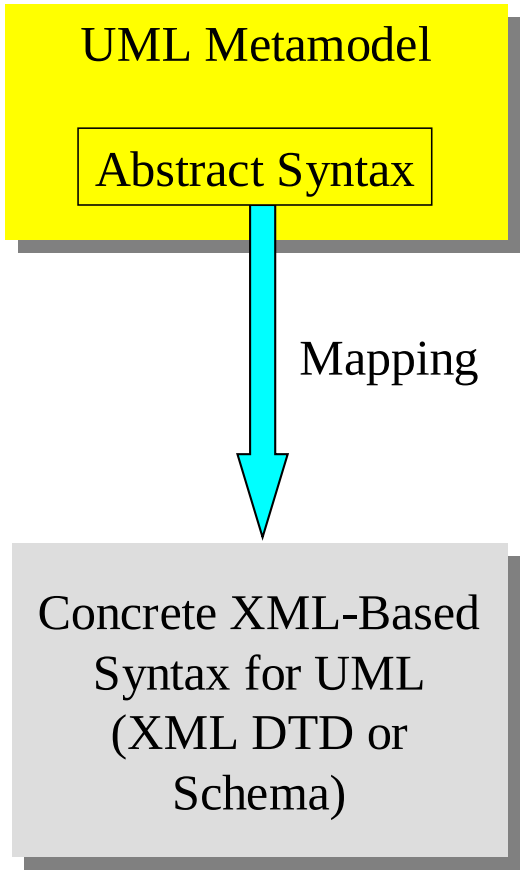



```
context Table inv:
   superclass.column->forAll
      (superClassColumn | self.column->includes (superClassColumn) )
```

Applying the MOF-XML Technology Mapping to CWM



Applying the MOF-XML Technology Mapping to UML



Enforcing Semantics

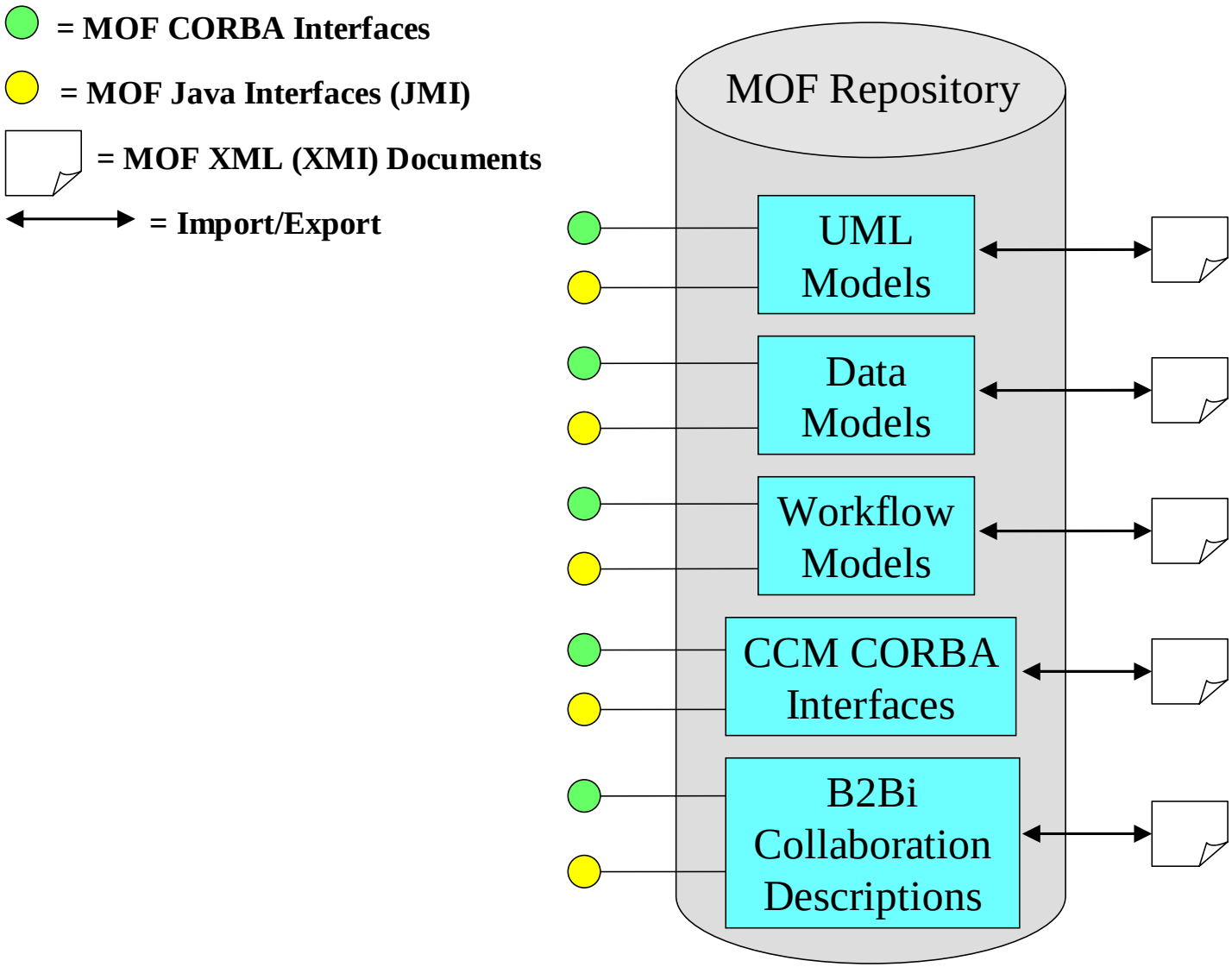
Different metamodels defined using the same constructs

- **Such as composite aggregation, invariants, etc.**

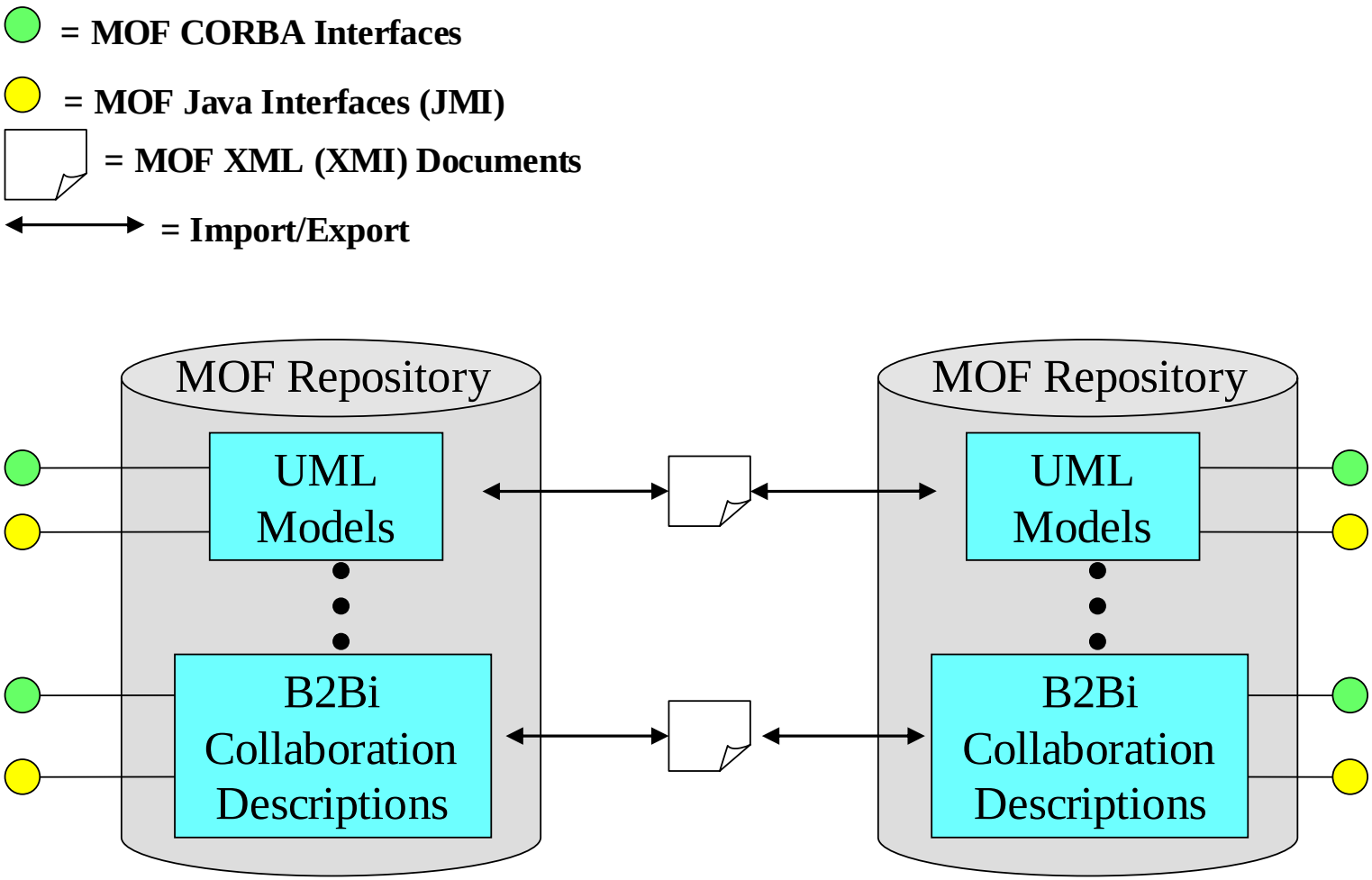
Model-driven metadata management tools understand these constructs' semantics

- **And can enforce them**

Metadata Management Scenario
1—Integrated MOF Repository



Metadata Management Scenario
2—Federated MOF Repositories



Agenda

- Value Chain Driven Business
- Industrializing Software
- Model-Driven Enterprise Architecture
- Informal vs. Formal Modeling
- Business Process Management
- Metadata Fragmentation
- Metadata Integration via MOF
- XMI and JMI
- UML Profiling
- PIMs and PSMs
- Model Driven Data Transformations
- The Future of MDA: Model-Driven Business Process Platforms

A Closer Look at XMI

Return on investment

- MOF was invented before XML was popular
- Platform-independence paid off

XML and XML Schema

- XMI 2.0 maps MOF to XML schema

A common misconception about XMI

- A MOF-XML mapping, not a single DTD for UML models

XMI Complexity vs. UML complexity

- Complexity of UML XMI DTD is due to complexity of the UML metamodel

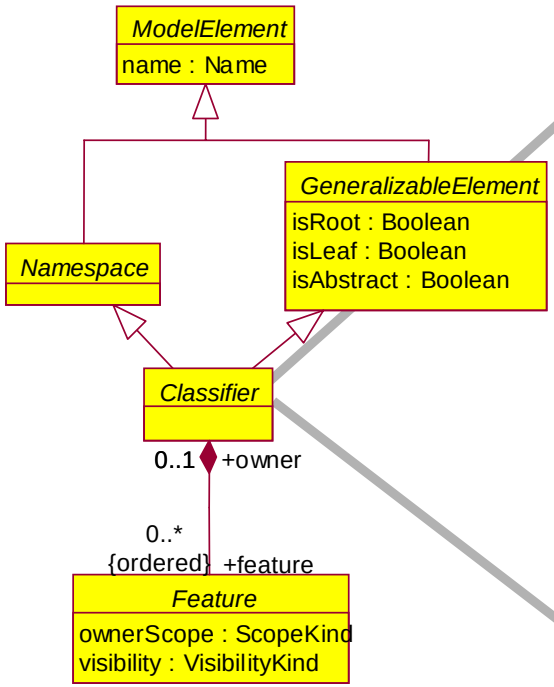
Applying XMI's MOF-XML Mapping Rules to the UML Metamodel

UML Metamodel

XMI's MOF-XML Mapping Rules

Produce

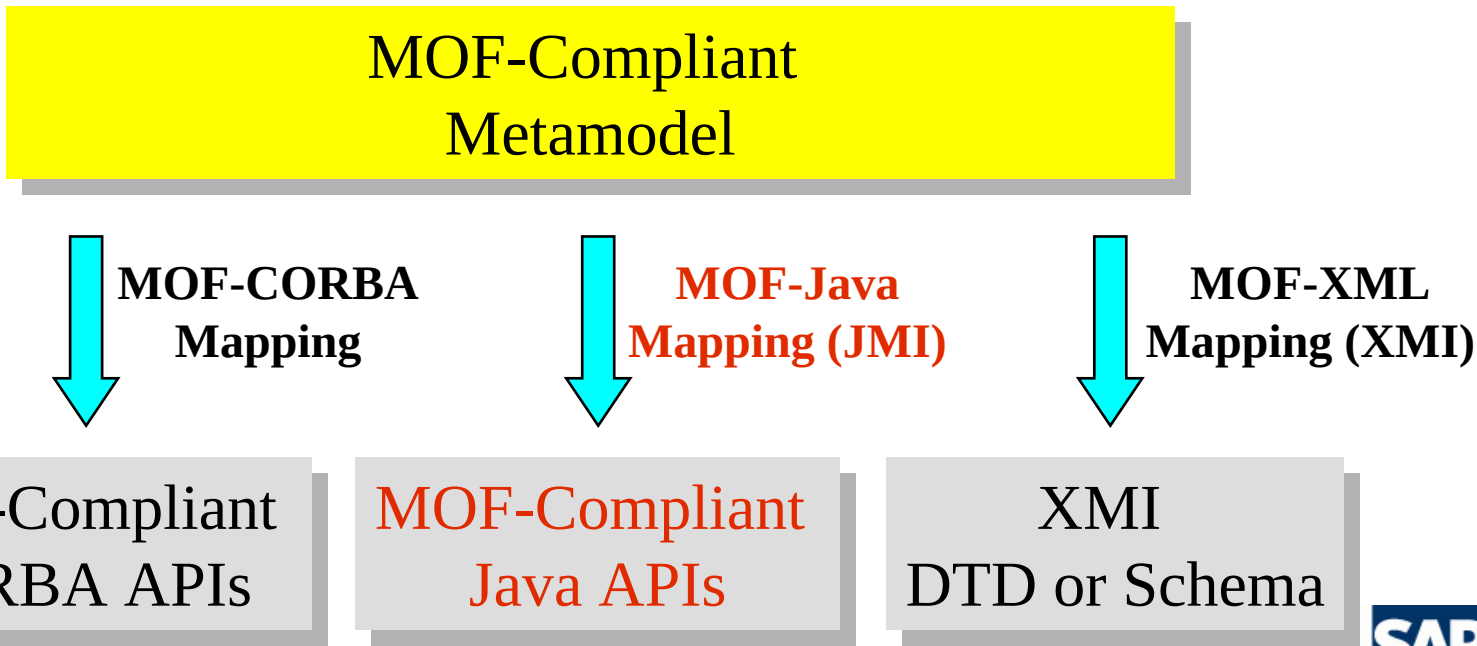
XML DTD




```
...
<!ELEMENT UML:Classifier.feature (
   UML:Feature|
   UML:StructuralFeature|
   UML:Attribute|
   UML:BehavioralFeature|
   UML:Operation|
   UML:Method|
   UML:Reception)*>
...
<!ENTITY % UML:ClassifierFeatures
   '%UML:GeneralizableElementFeatures; |
    UML:Namespace.ownedElement |
    UML:Classifier.feature |
    ...'>
...
<!ELEMENT UML:Classifier (%UML:ClassifierFeatures;)*>
...
```

A Closer Look at JMI

- **For representing MOF metadata as Java objects**
- **Specified via Sun JSR #40**
- **Specifies syntax and semantics of generated interfaces**



MOF in the Computer Industry

Deeply wired into Eclipse

Other implementations: Sun, Adaptive, Metamatrix

New MOF-based initiatives

- **Business Process Definition Metamodel (OMG)**
  - **BPMI.org involved**
- **Business Rules Metamodel (OMG)**
  - **Key people from business rules community involved**
- **Ontology Definition Metamodel (OMG)**
  - **Key people from Semantic Web community involved**
- **Distributed Management Task Force (DMTF)**
  - **Moving toward MOF-based metadata**
- **Model-Driven data transformations a huge opportunity (CWM)—a killer app for MDA**

Microsoft committed to model-driven approach

- **But not to MOF**

The Eclipse Modeling Framework (EMF)

Heritage: Visual Age 3.0

- **Model-Driven, XMI-based metadata management**
- **Invisible to IDE user**

Open sourced the mature engine as part of Eclipse

EMF penetration in IBM tooling

- ***WebSphere DB2 Information Integrator*—formerly named DB2 Information Integrator**
- ***Websphere Business Integration Modeler*—formerly Holosophix**
- ***Rational Software Modeler*—IBM's new UML modeling tool**
- ***Rational Application Developer for WebSphere Software*—formerly named WebSphere Studio Application Developer**
- ***Rational Software Architect*—An extension of the Rational Software Modeler that supports generating code to the WebSphere runtime**

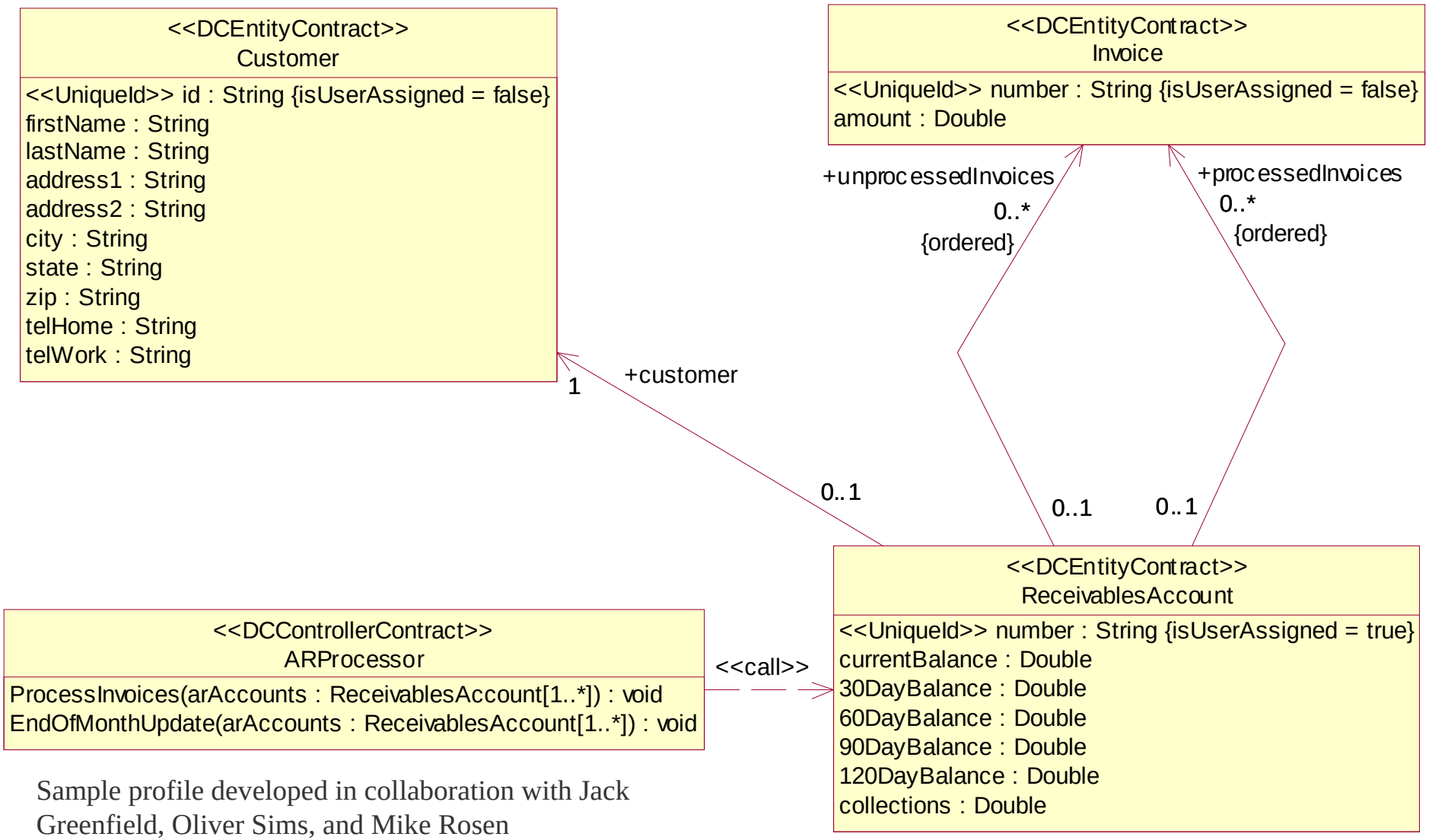
Agenda

- Value Chain Driven Business
- Industrializing Software
- Model-Driven Enterprise Architecture
- Informal vs. Formal Modeling
- Business Process Management
- Metadata Fragmentation
- Metadata Integration via MOF
- XMI and JMI
- UML Profiling
- PIMs and PSMs
- Model Driven Data Transformations
- The Future of MDA: Model-Driven Business Process Platforms

Extending UML via Profiles

- **UML extension mechanisms allow UML to be only a base language for many languages in the family**
- **"A family of languages" (credit to Steve Cook of Microsoft)**
- **To model a particular domain, e.g. business information, business services, business collaborations, real time systems, telecomm, etc.**
- **To model at specific levels of abstraction**
  - **Platform-independent**
  - **Platform-specific**
- **To parameterize mappings to specific technologies**

Stereotypes and Tagged Values



Sample profile developed in collaboration with Jack Greenfield, Oliver Sims, and Mike Rosen

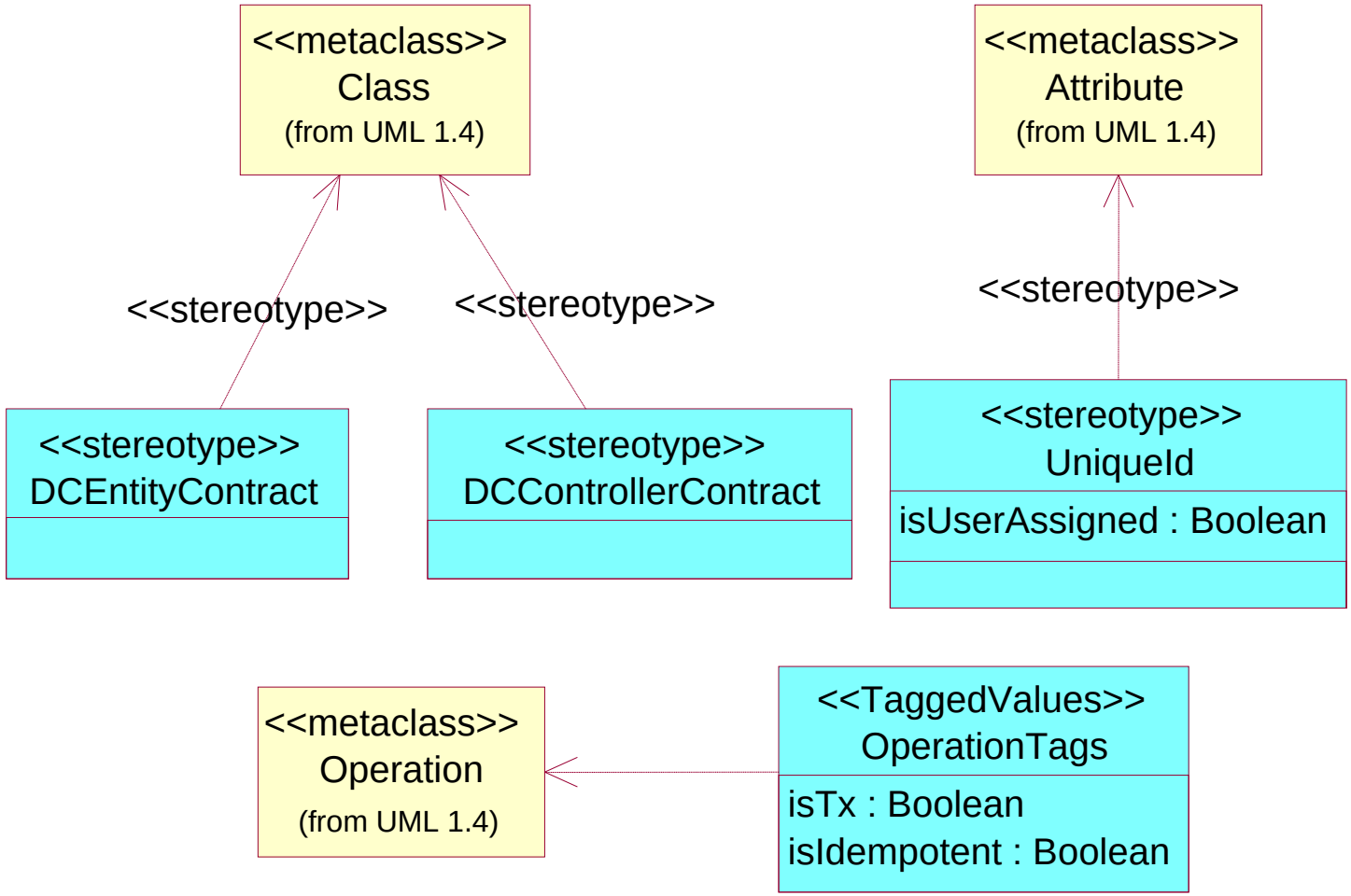
To Define a UML Profile...

Select a subset of UML

Use UML's built in extension mechanisms to extend the subset

- **Stereotypes, e.g. <<DCEntityContract>>**
- **Tagged Values, e.g. {isTx = true}**

Formal Class Model of the Sample Profile




```
context DCEntityContract inv:
   self.feature->exists (isStereotyped ('UniqueId'))
```

Profiles vs. Heavyweight Extensions

Profiles

- Pro: Modelers can use a profile with generic UML tools
  - Sample models for this book that use profiles were created with generic
- Con: Extension capabilities limited

Heavyweight extensions (i.e. defined via MOF)

- Con: Modelers cannot use with generic UML tools
- Pro: Can use full power of object-oriented class modeling to define extensions

Standardized MDA Modeling Languages

- UML
- MOF
- SysML
- UML Profile for EJB™
- UML Profile for CORBA
- UML Profile for EAI
- UML Profile for EDOC
- UML Profile for Schedulability and Time (Realtime)
- UML Profile for Testing
- UN/CEFACT Modeling Methodology (UMM)
- Common Warehouse Metamodel (CWM™)

Agenda

- Value Chain Driven Business
- Industrializing Software
- Model-Driven Enterprise Architecture
- Informal vs. Formal Modeling
- Business Process Management
- Metadata Fragmentation
- Metadata Integration via MOF
- XMI and JMI
- UML Profiling
- PIMs and PSMs
- Model Driven Data Transformations
- The Future of MDA: Model-Driven Business Process Platforms

PIMs and PSMs

PIM = Platform-Independent Model

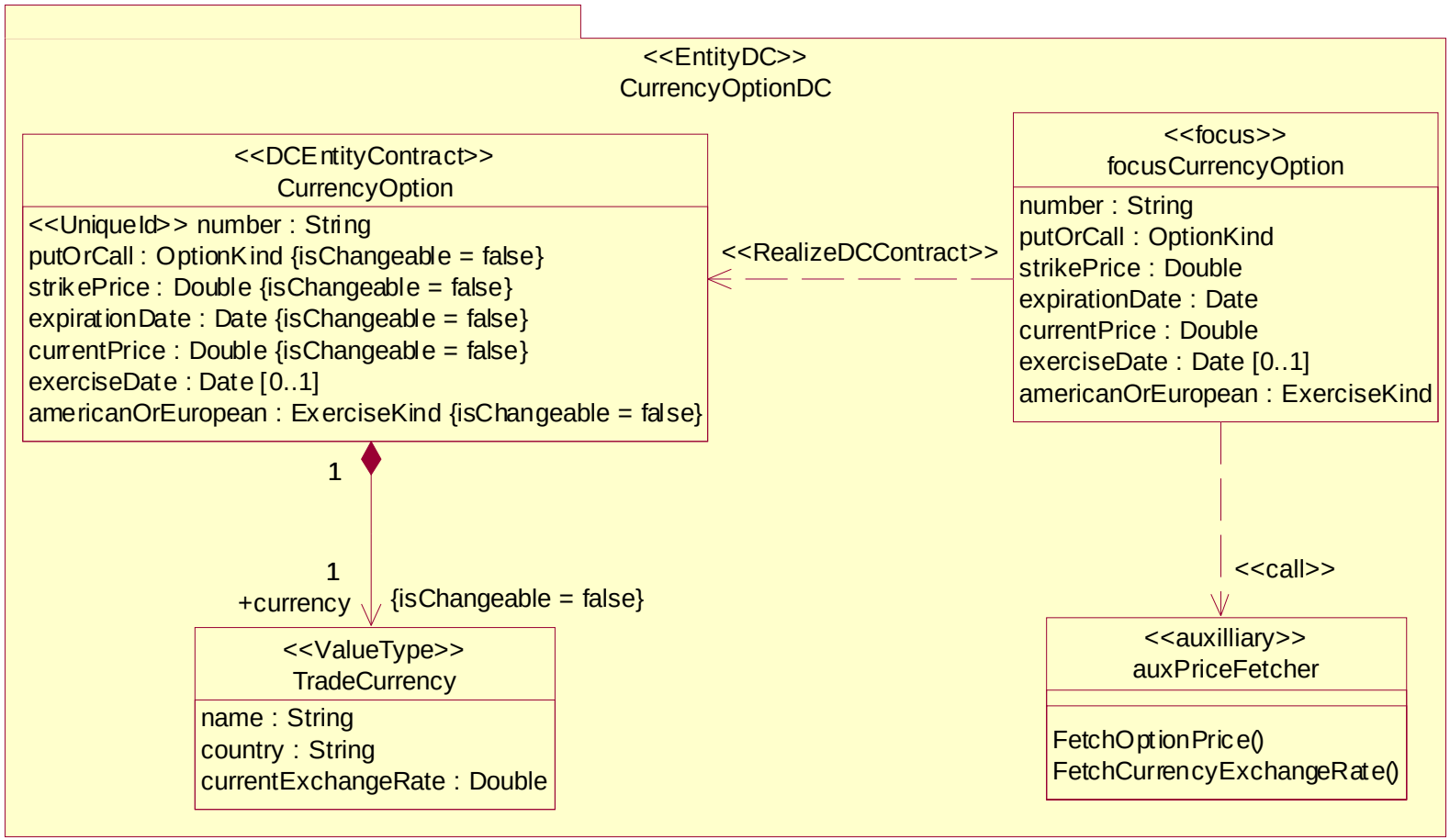
PSM = Platform-Specific Model

***Platform* is a relative concept**

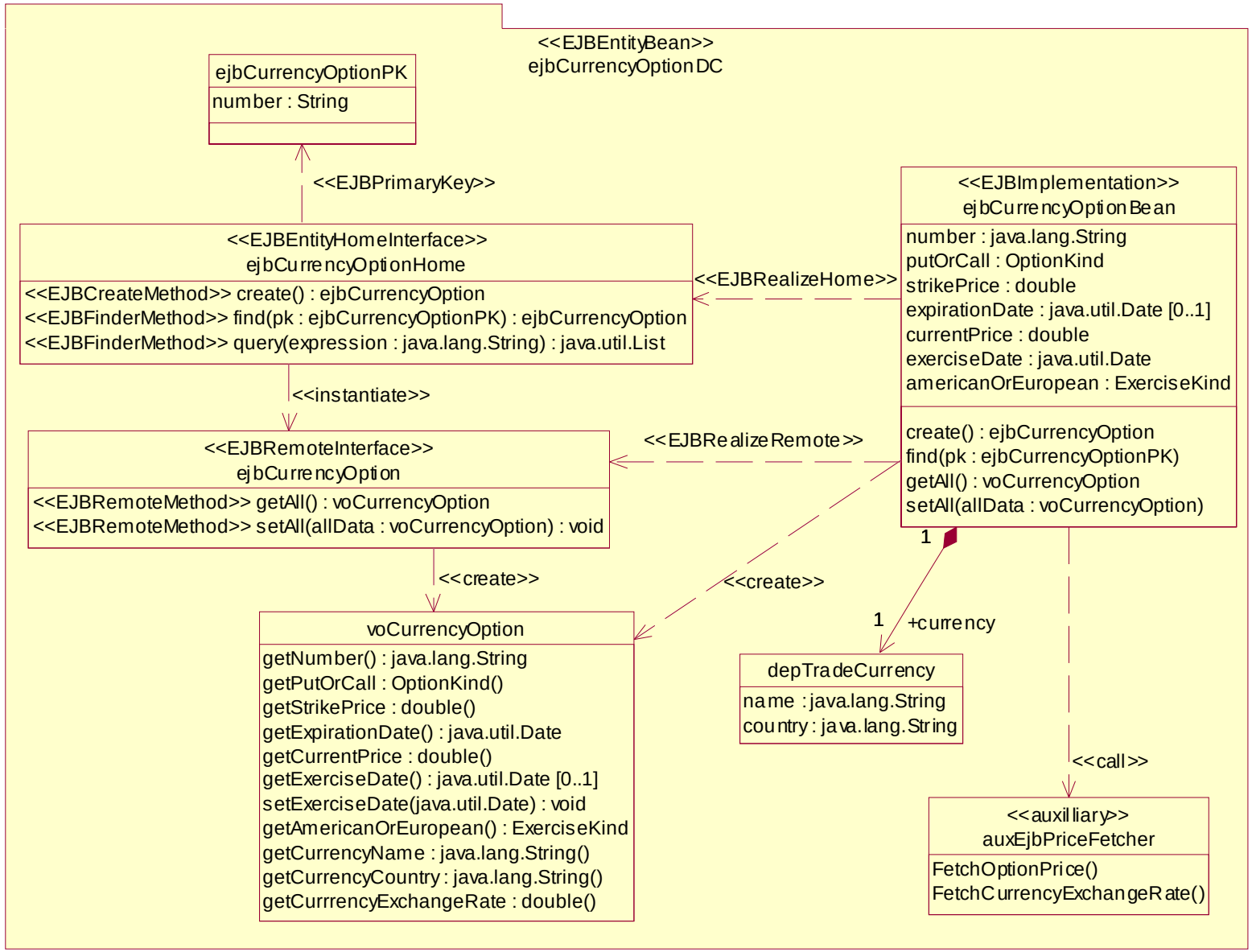
Original conception of PSM: A model of the generated artifacts

- **At the same abstraction level as the code and other artifacts generated from the PIM**
- **Allows engineers to view the generated code via a semantically rich model**
- **Can be a read-only artifact**

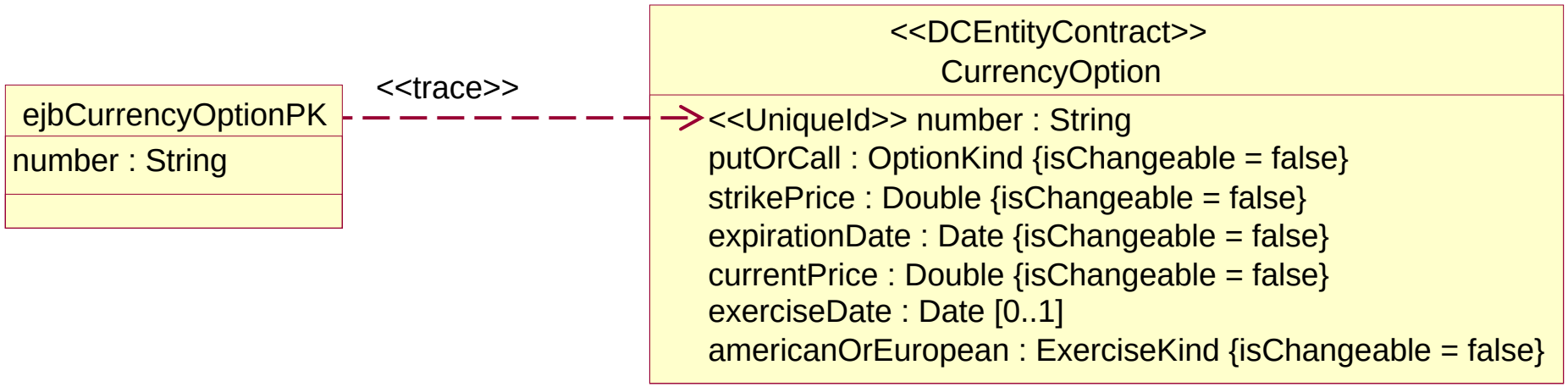
PIM for a Distributed Entity Component



EJB-Specific Component Model (PSM)



Formal Relationship Between Elements at Different Abstraction Levels



Read-Only PSMs

When code generation proceeds directly from PIM

Helps humans visually grasp the platform-specific architecture

Helps during debugging, when debugging tools don't know about the PIM abstractions

No fixed notions of PSM, PIM, CIM...

MDA is machinery for authoring, transforming, and managing specifications

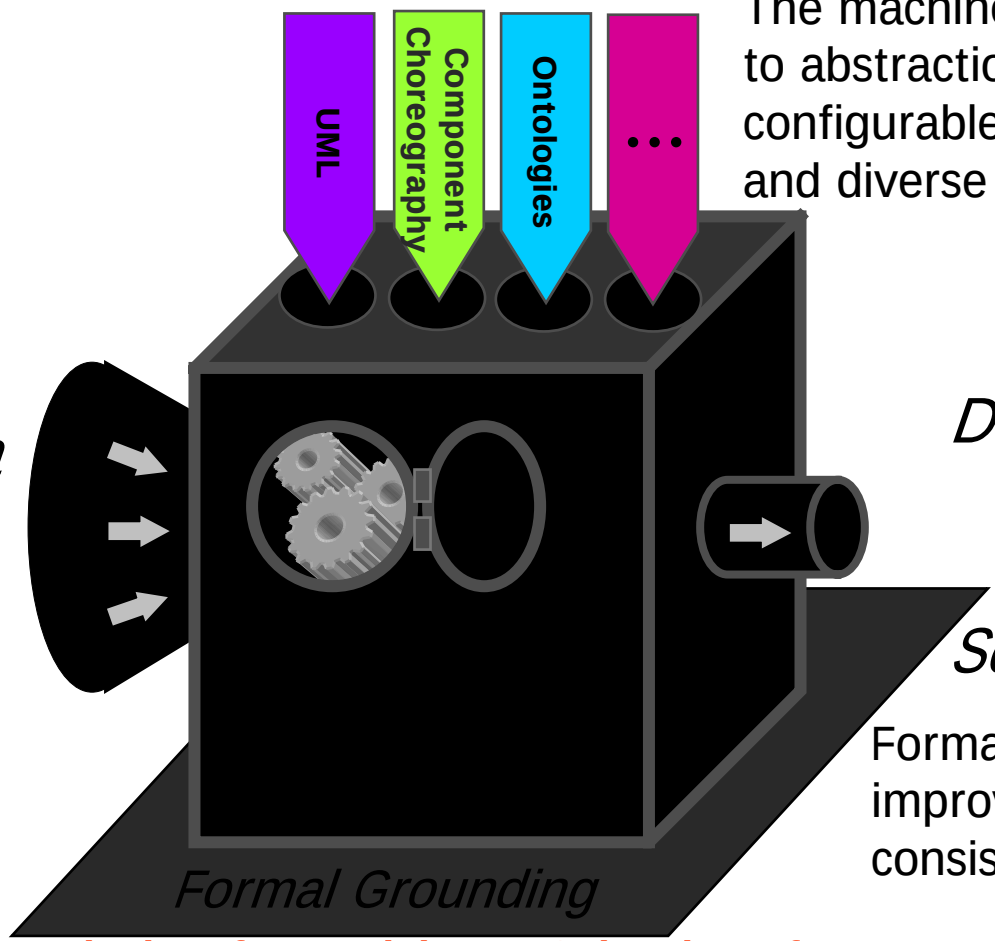
Specifications may take the form of:

Models

Data

Code

Schemas



The machinery is neutral as to abstraction level, configurable for numerous and diverse uses

Code

Data

Models

Schemas

Formal grounding would improve our ability to verify consistency and correctness

Adapted, with permission, from Erick Von Schweber of Synsyta LLC

Architectural Patterns and Frameworks to Which MDA Can Be Applied

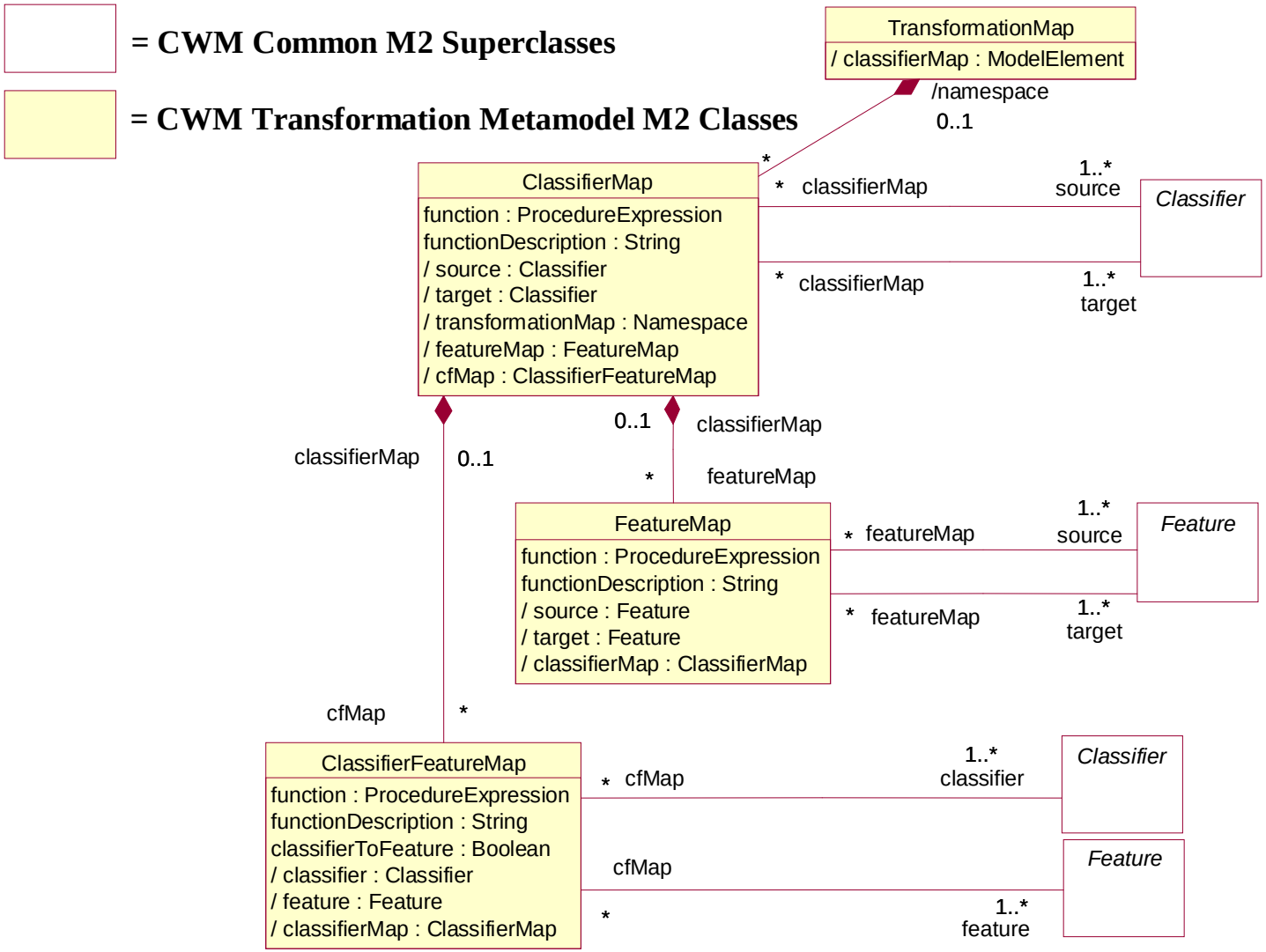
Application Development via Refinement
Architecture-Driven Modernization (ADM)
Service-Oriented Architecture
Multiparty Collaboration
Business Process Management
Software/Hardware Codesign
Ontology Modeling & Alignment
Embedded & Real-Time Systems
Model Integrated Computing
Software Factories
Platform Hierarchies
Orchestration
System of Systems
Reflective Systems & Metaprogramming
Agent Architectures
Multi-Dimensional Separation of Concerns
Representation Optimization
Policy & Business Rules
Federal Enterprise Architecture (FEA)
DoD Architecture Framework (DoDAF)
The Open Group Architecture Framework (TOGAF)
RM-ODP

Adapted, with permission, from Erick Von Schweber of Synsyta LLC

Agenda

- Value Chain Driven Business
- Industrializing Software
- Model-Driven Enterprise Architecture
- Informal vs. Formal Modeling
- Business Process Management
- Metadata Fragmentation
- Metadata Integration via MOF
- XMI and JMI
- UML Profiling
- PIMs and PSMs
- Model Driven Data Transformations
- The Future of MDA: Model-Driven Business Process Platforms

Core of the CWM Transformation Metamodel



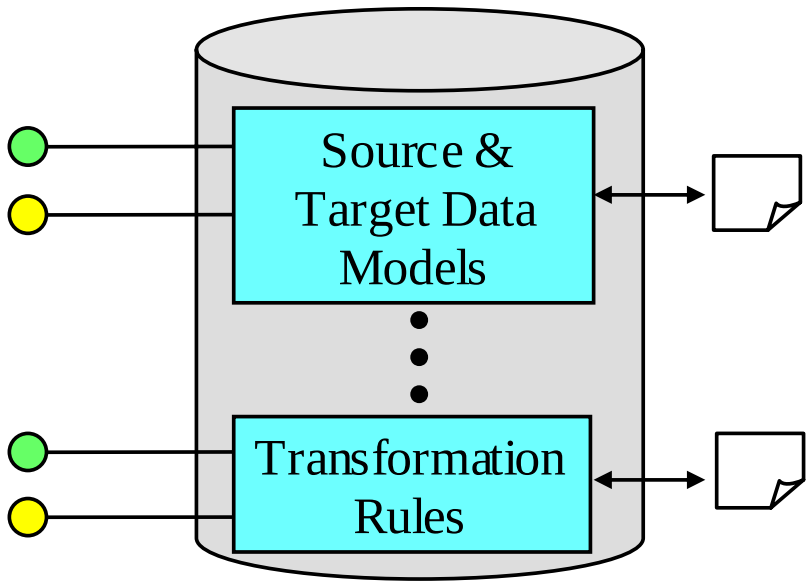
Managing CWM Transformation Rules as MOF Metadata

= MOF CORBA Interfaces

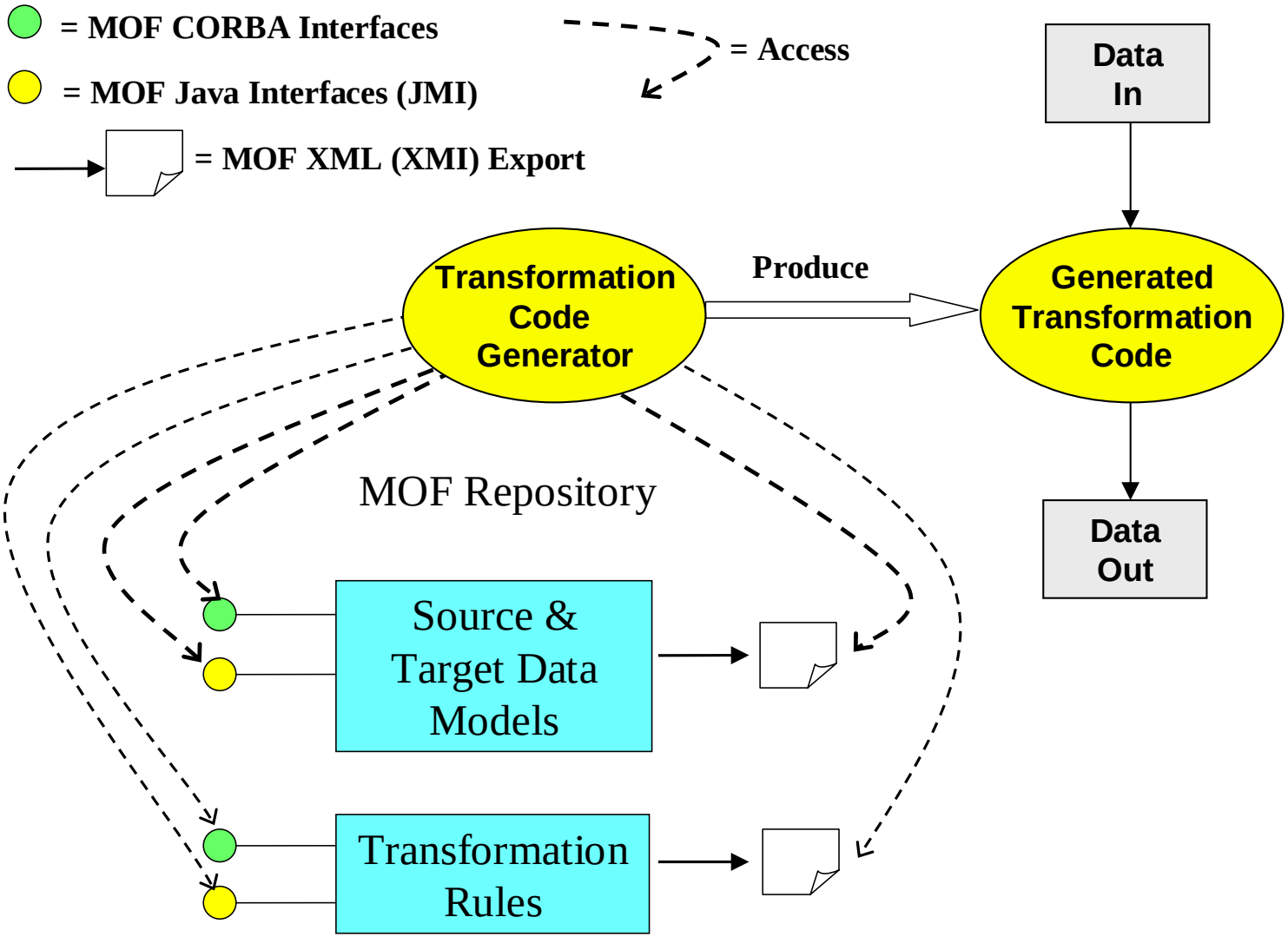
= MOF Java Interfaces (JMI)

= MOF XML (XMI) Documents

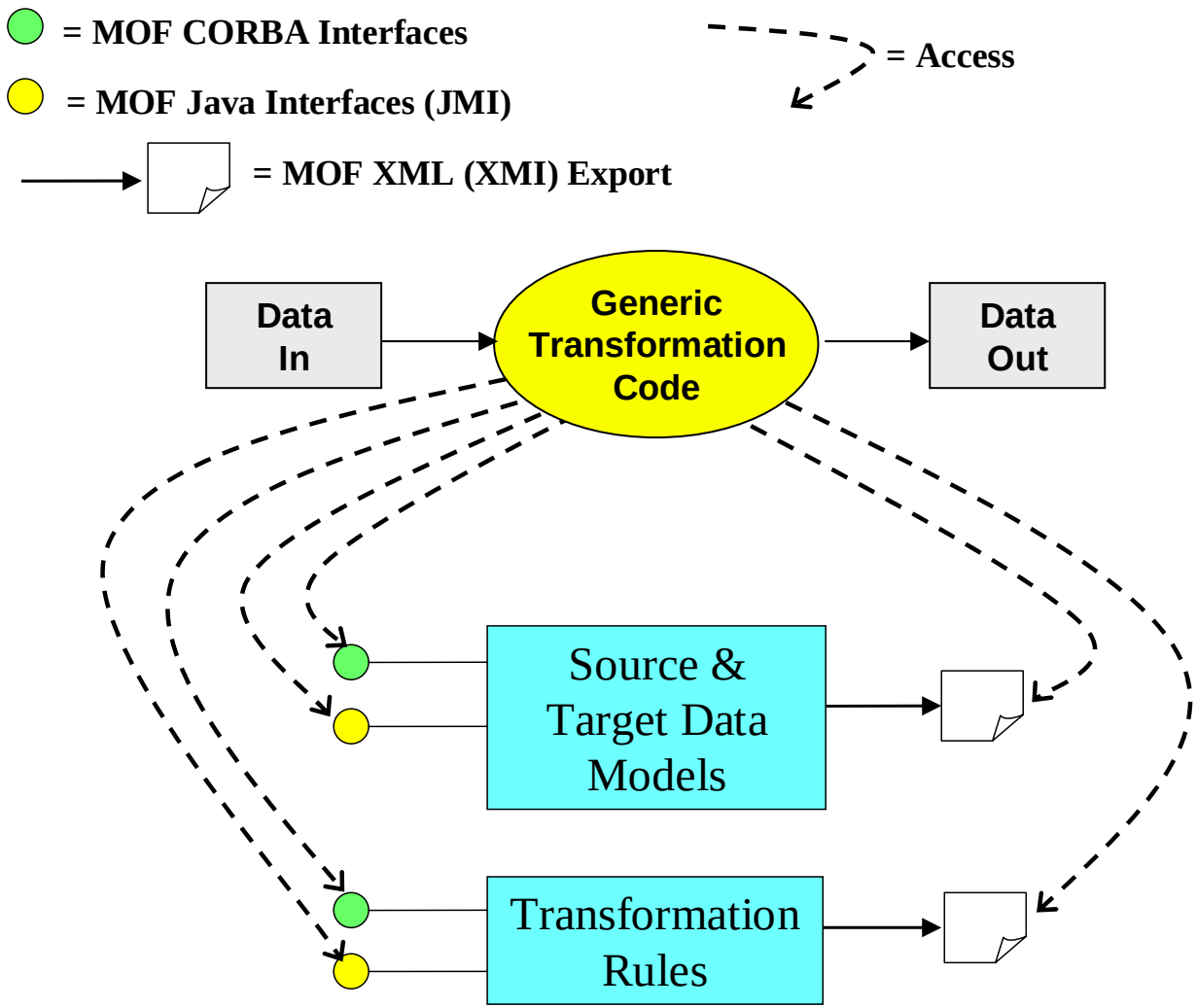
= Import/Export



Generating Transformation Code



Dynamic Transformation



Agenda

- Value Chain Driven Business
- Industrializing Software
- Model-Driven Enterprise Architecture
- Informal vs. Formal Modeling
- Business Process Management
- Metadata Fragmentation
- Metadata Integration via MOF
- XMI and JMI
- UML Profiling
- PIMs and PSMs
- Model Driven Data Transformations
- The Future of MDA: Model-Driven Business Process Platforms

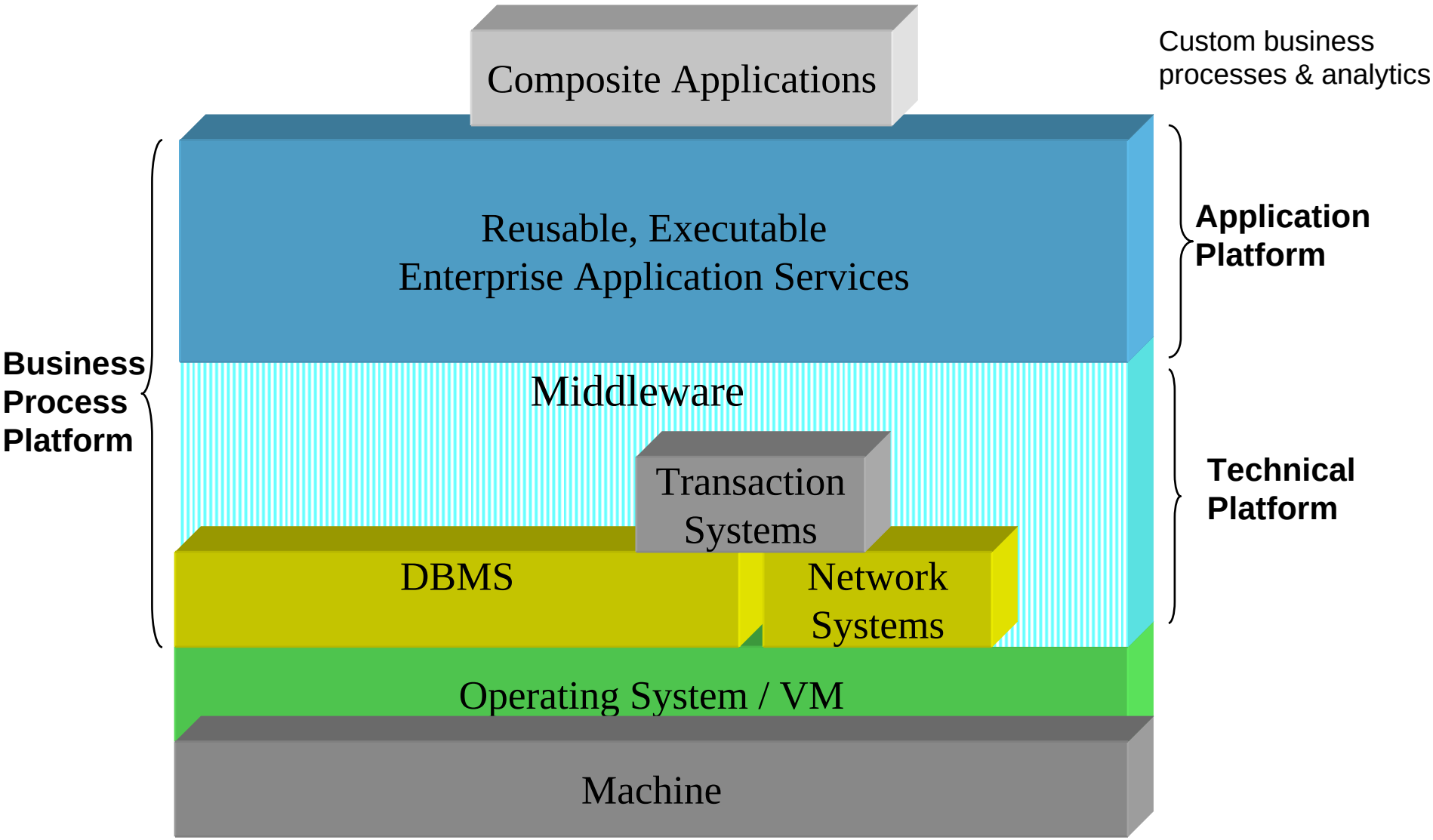
Raising the Abstraction Level of the Platform

Concurrent with MDA's focus on raising the abstraction Level of development languages

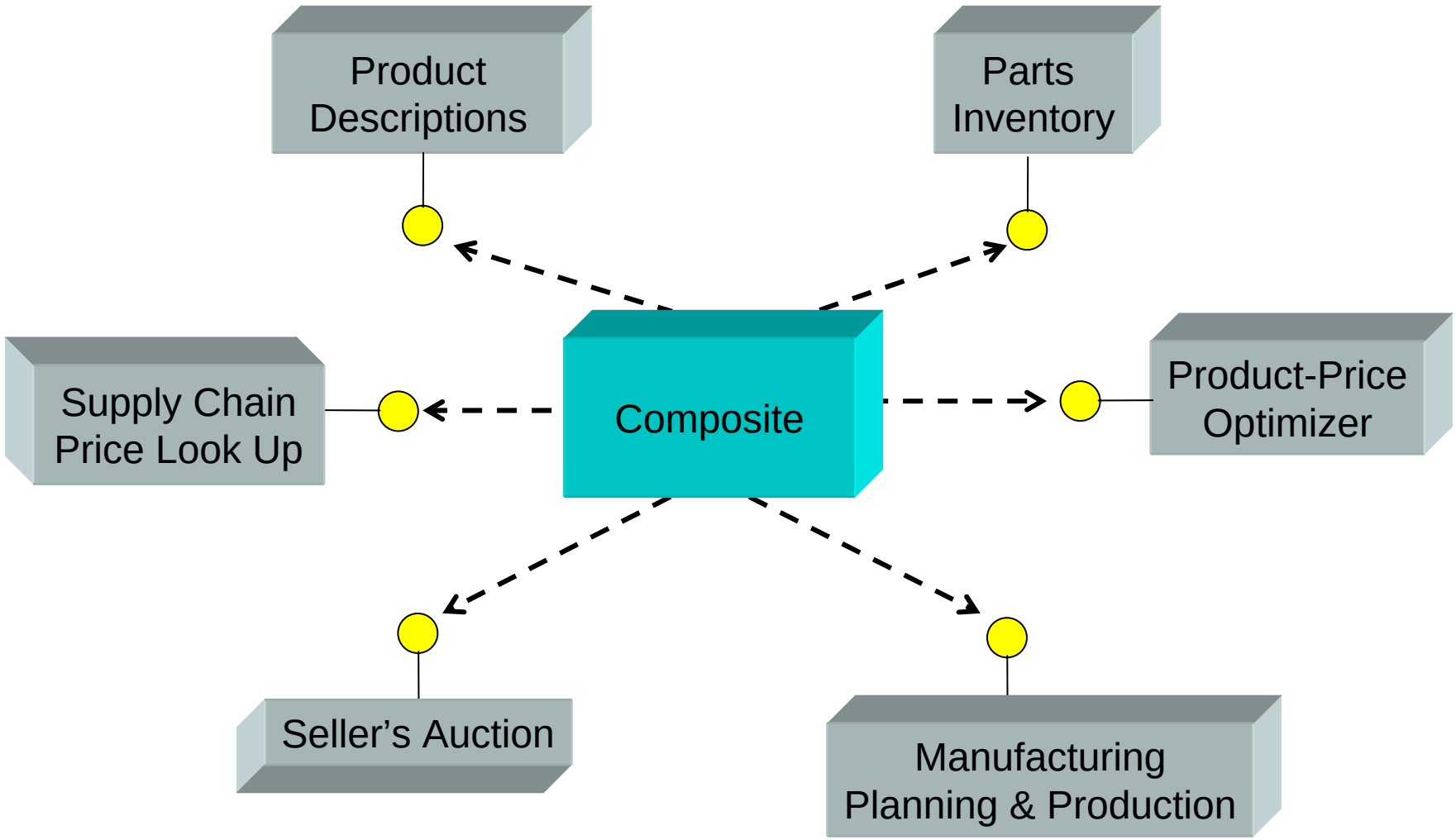
Powerful transition…but gradual

- **Complications to overcome**

Business Process Platform
= Technical Platform + Application Platform

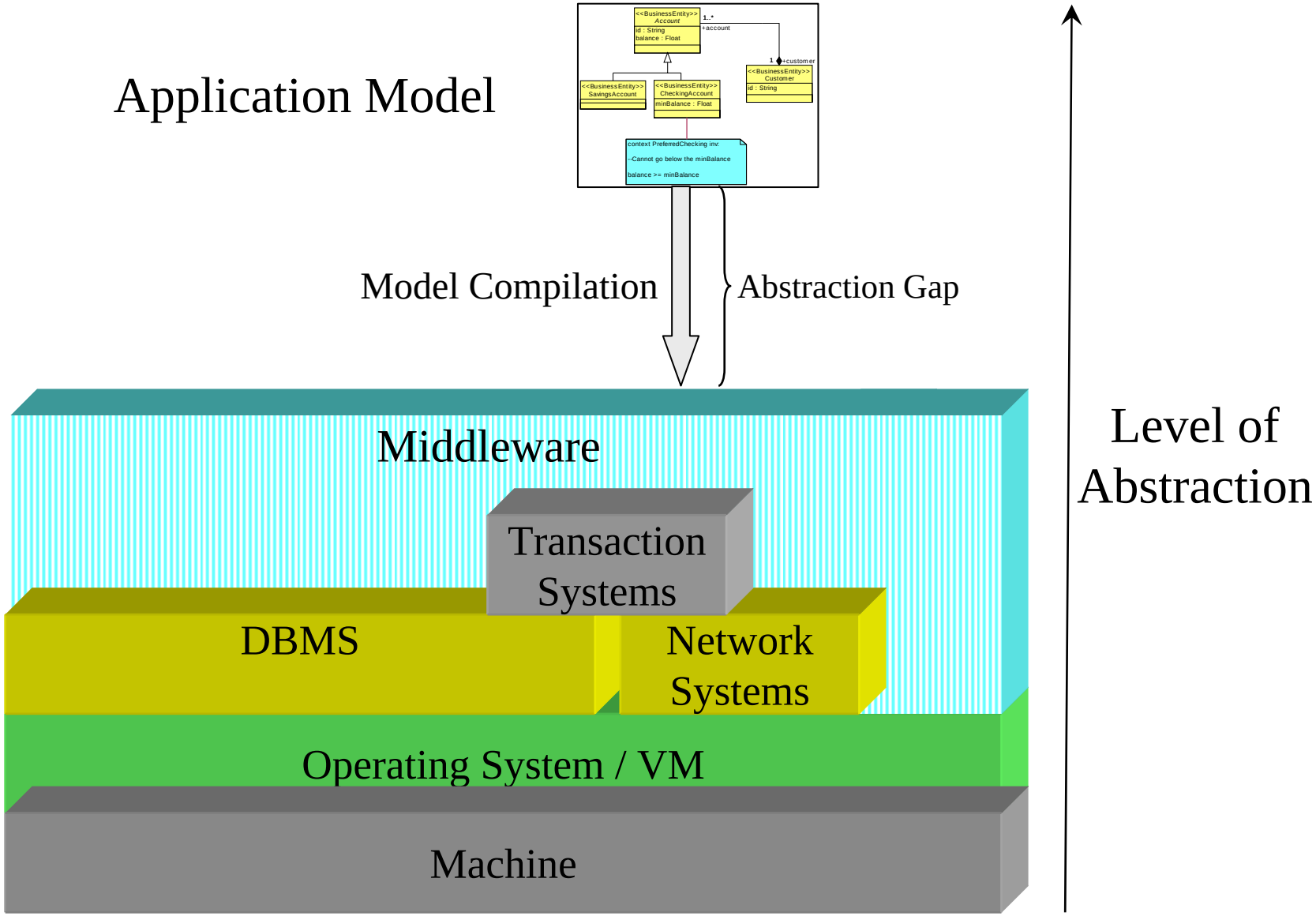


Composite App: Procure-to-Pay, Order-to-Cash, Manufacture-to-Inventory

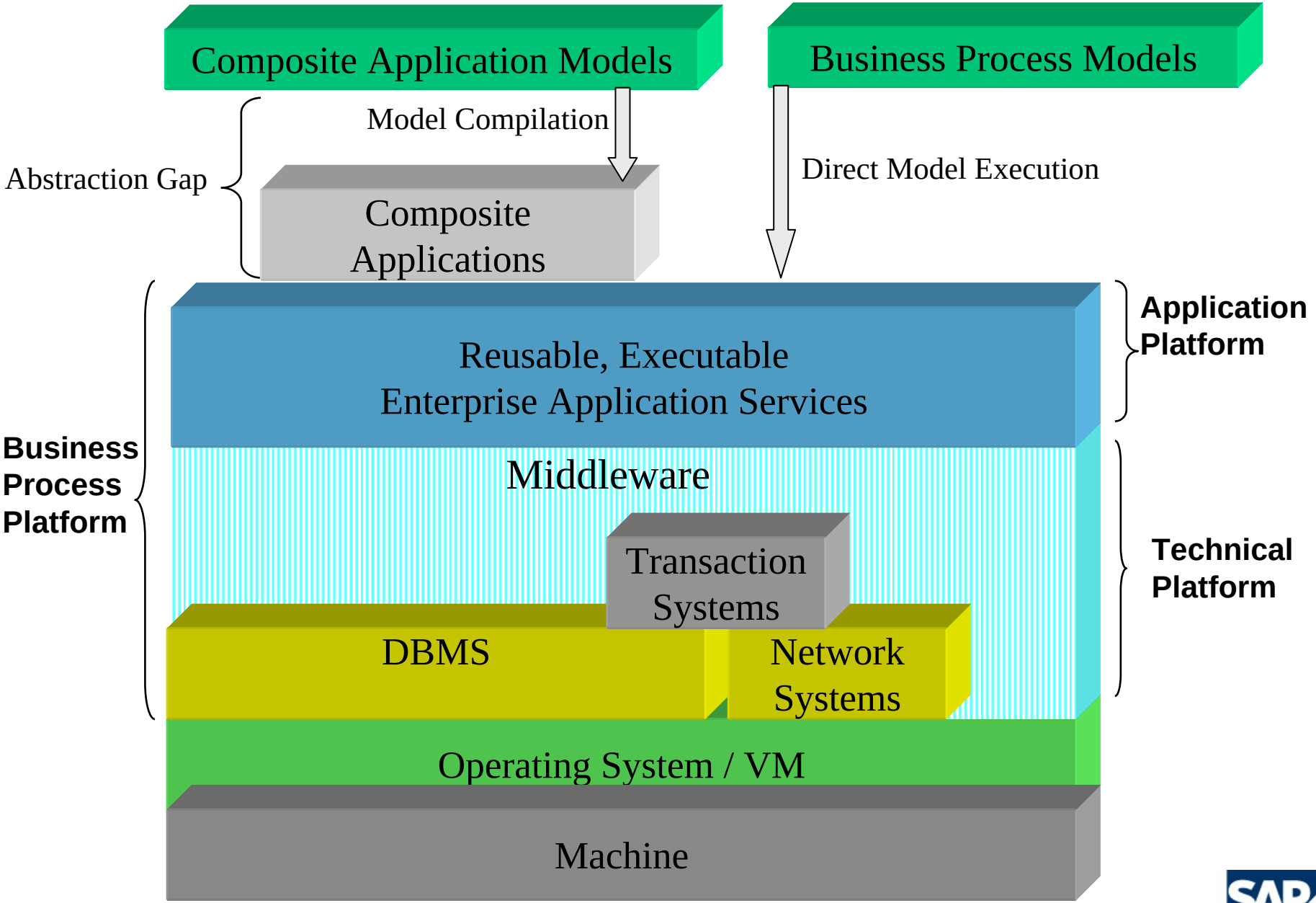


David Burdett, SAP Labs

Model Compilers and the Abstraction Level



Business Process Platform



Convergence
At the Intersection of the Business and IT[1]

Business

CA-BPM
Bringing computer assistance to business process management

IT

MDA
Raising the level of abstraction for software development

[1]Howard Smith and Peter Fingar coined this phrase

This Will Not be an Easy Transition

This jump in the platform abstraction level is more difficult than the last jump (middleware)

- **Just as raising the abstraction level for development languages above 3GLs is more difficult than the last jump to 3GLs**

Crawl, Walk, Run

- **Provide business value at every step**

Some of the Complications

Configuration/version management problems do not go away

- **They can even get worse**

Semantically thin specifications won't scale

Product line practices needed to scope compositions

Multiple Levels of Granularity and Abstraction

- **Multiple architectural patterns and methodologies**
- **Fixed notions of PIM, PSM, etc. won't scale**

The Limits of Semantically Thin Specifications

As service-oriented systems scale up…

- **How do you build semantic interoperability on top of syntactic interoperability?**
  - **Do collaborating parties have a common understanding the contract of a service?**
  - **You can't rely on that conversation by the water cooler…**
  - **The parties might have different human languages as native tongues**
- **How do you find suitable services to compose?**
- **How do you find suitable implementations of suitable services?**
- **Same for reusable business processes**

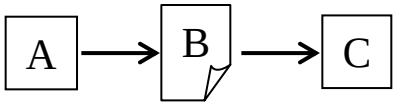
Inferences don't have to be certain to be helpful

Semantically Rich Specifications

Need a metadata-rich environment to *assist* humans using the business process platform

- **Design by Contract™**
  - **Service message types specified as precisely as possible**
    - **Invariants**
  - **Service operations functional contract specified as precisely as possible**
    - **Preconditions and postconditions (more numerous than invariants)**
  - **Using machine-readable, declarative constraint languages**
  - **We've know how to do this for decades**
  - **Also improves quality**
  - **Also need to learn to specify QoS requirements and capabilities as precisely as possible**
- **Formal grounding of languages**
- **Inferences identify candidates or flag potential problem combinations**
  - **Let the human decide what to do**
  - **Record what the human decides**
  - **Show the next human what the others decided**
  - **Learn**

Bridging the Semantic Web and MDA Worlds



Means tool A outputs B, and B serves as input to tool C.

Generic UML Modeling Tool

UML Model Using UML Profile

Generic UML Model

Native Semantic Web Ontology Development Tool

UML-OWL Bridge

MOF/XMI Based Tooling (e.g. Semantic Web Eclipse Plug-In)

Native OWL Document

XMI OWL Document

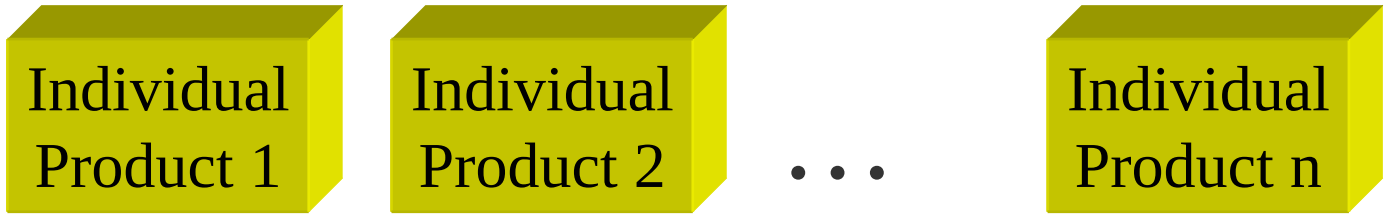
ER Tool

OWL-XMI Bridge

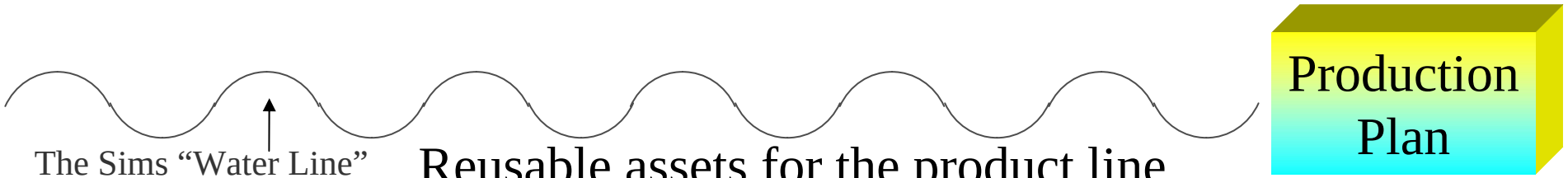
ER-OWL Bridge

ER Model

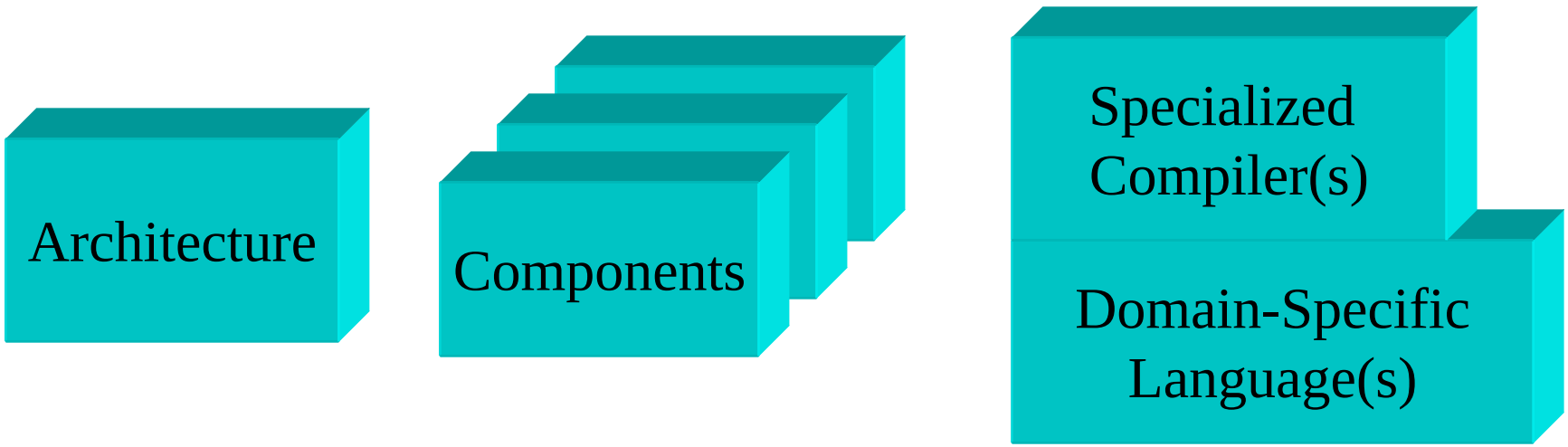
Product Line Practices



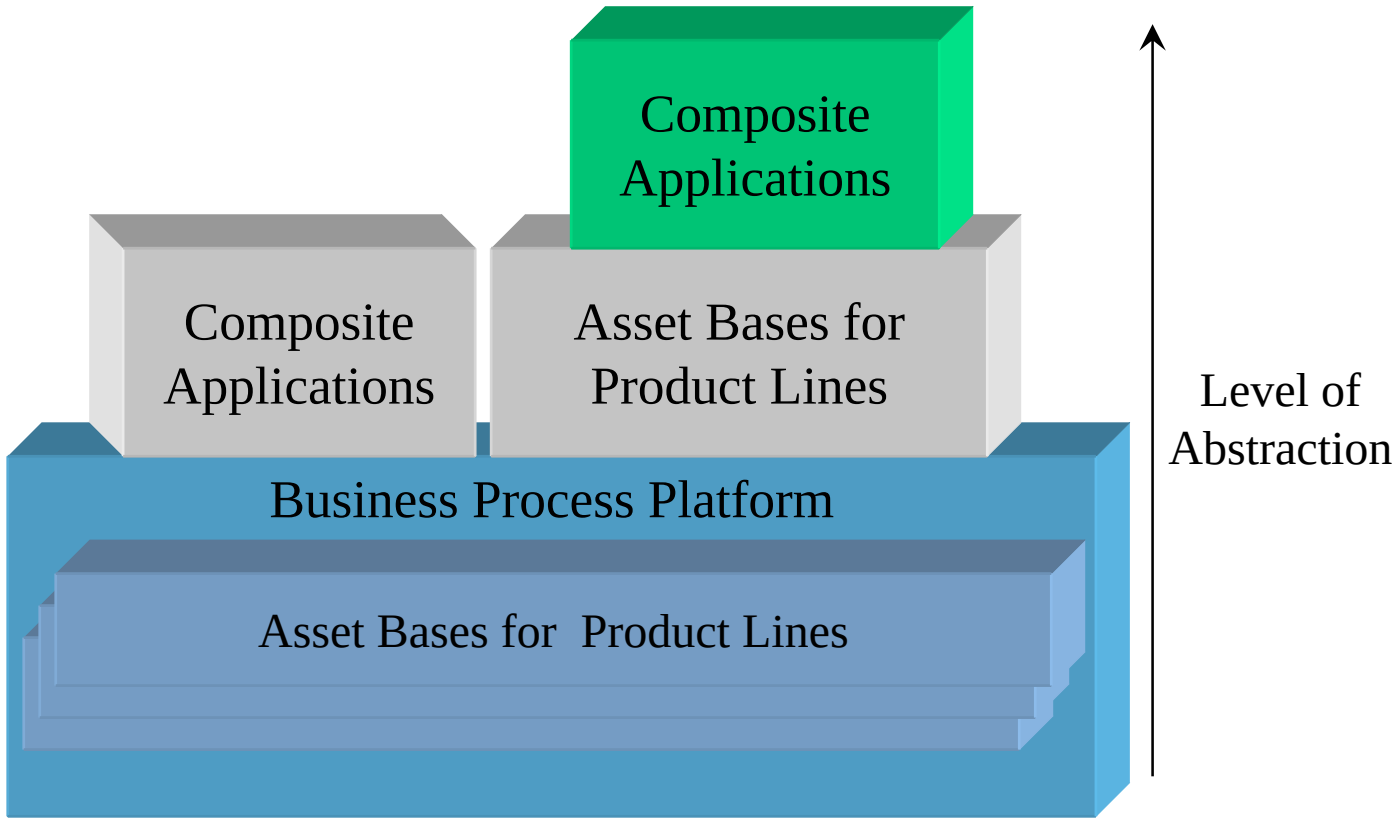
Individual systems produced via *product development*



Reusable assets for the product line
Created via *core asset development*



Applying Product Line Practices



Agenda Review

- Value Chain Driven Business
- Industrializing Software
- Model-Driven Enterprise Architecture
- Informal vs. Formal Modeling
- Business Process Management
- Metadata Fragmentation
- Metadata Integration via MOF
- XMI and JMI
- MOF in the computer industry
- UML Profiling
- Model Driven Data Transformations
- The Future of MDA: Model-Driven Business Process Platforms

Copyright 2005 SAP AG and David S. Frankel. All Rights Reserved

Copyright 2005 SAP AG und David S. Frankel. Alle Rechte vorbehalten