

**Presentation at "MDA Information Day"  
during the OMG Technical Meeting in April 2002**

# **MDA and System Design**

Makoto Oya  
Hitachi, Ltd.  
April 23, 2002

## Activities in OMG

- 1989: OMG Established.
- Standardization of Distributed Object Middleware
  - 1995: CORBA2; 2001: CORBA2.5
- Domain (industry specific & cross-industry) Standardization
  - 1995-: Standards in various domains
- Modeling Standardization
  - 1997: UML(Unified Modeling Language)
  - 1997: MOF; 1999: XMI; 2000:CWM
  - 2001: Application-specific UML Profiles (EDOC, EAI)
- Architecture (Reference Model)
  - 1990: OMA (Object Management Architecture)
  - 2001: **MDA (Model Driven Architecture)**
- 2001-: starting Standardization based on MDA
- 2002(planned): UML V2 --- expected to include MDA base functionality

# Agenda

- Background and Vision
- MDA's Approach
- Toward Realization of MDA
- Summary

# Background and Vision

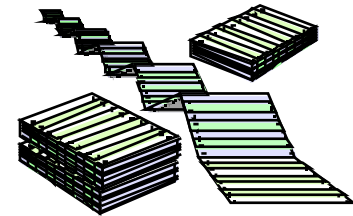
# Integration of Business Processes



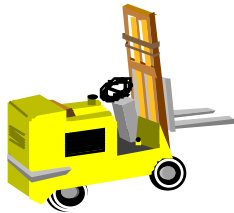
**Sales**



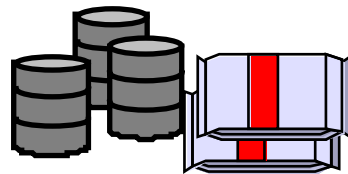
**Engineering**



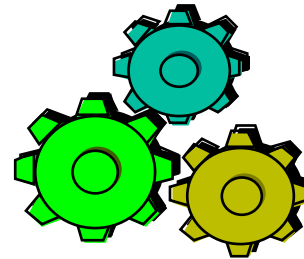
**Accounting**



**Shipping/  
Receiving**



**Inventory**



**Manufacturing**



**Payables/  
Receivables**

- Improved steadily for more than 20 years.
- However, still existing big challenges.

# Root of Problems = Varieties of Platforms

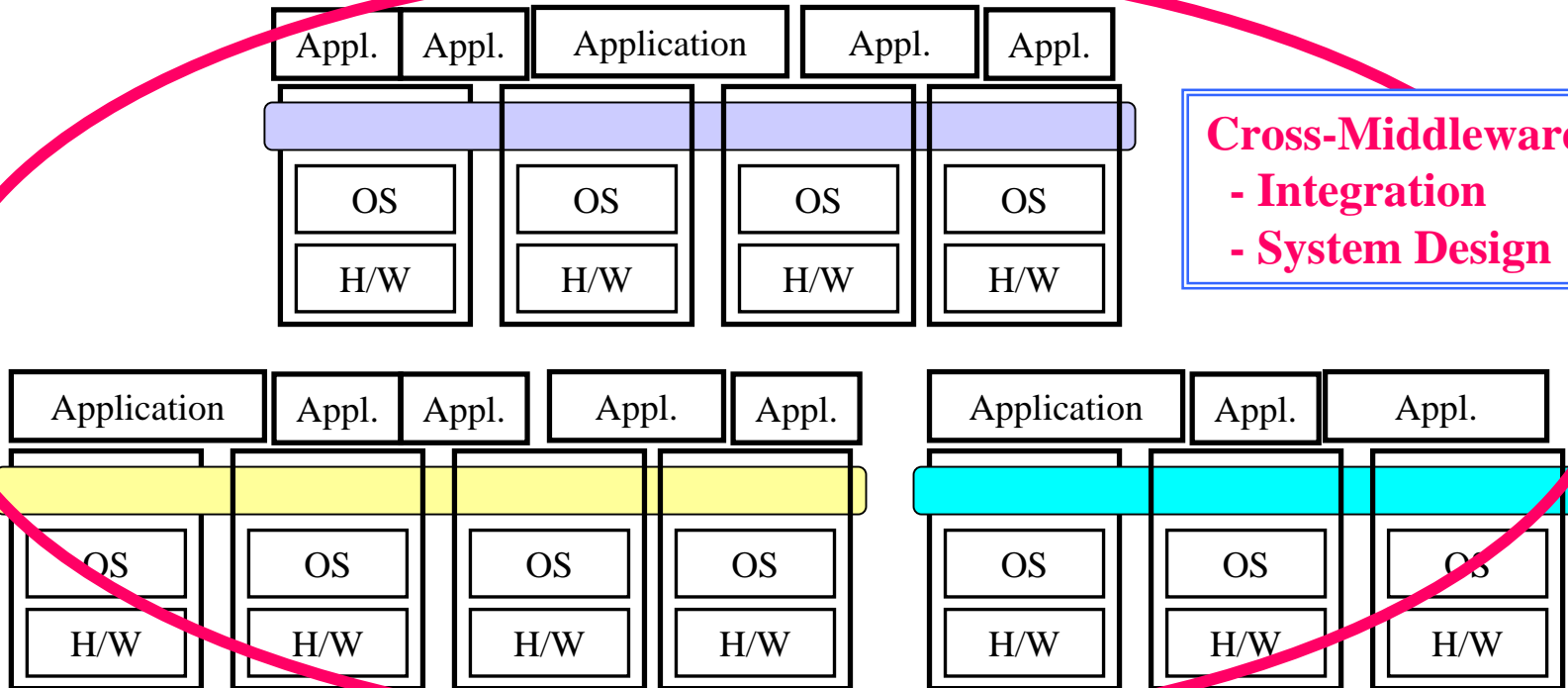
- Variety of Hardware Architectures
  - Pentium, PowerPC, PA-RISC, Sparc, 370, ...
- Variety of Networks
  - Ethernet, ATM, IP, SS7, Appletalk, USB, Firewire, ...
- Variety of Programming Languages
  - C/C++, Java, Visual Basic, C#, Perl, JavaScript, VBScript, COBOL, PL/I, Fortran, ...
- Variety of Operating Systems
  - Unix, Windows, NT/XP, Mainframe OS, MacOS, Windows CE, Mobile phone, Set-top box, Game machine, ...
- **Then, Variety of Middlewares**
  - JAVA/CORBA, COM+/.NET, Web Services(SOAP, ebXML, ...)

# Success, Evolution and Proliferation of Middleware

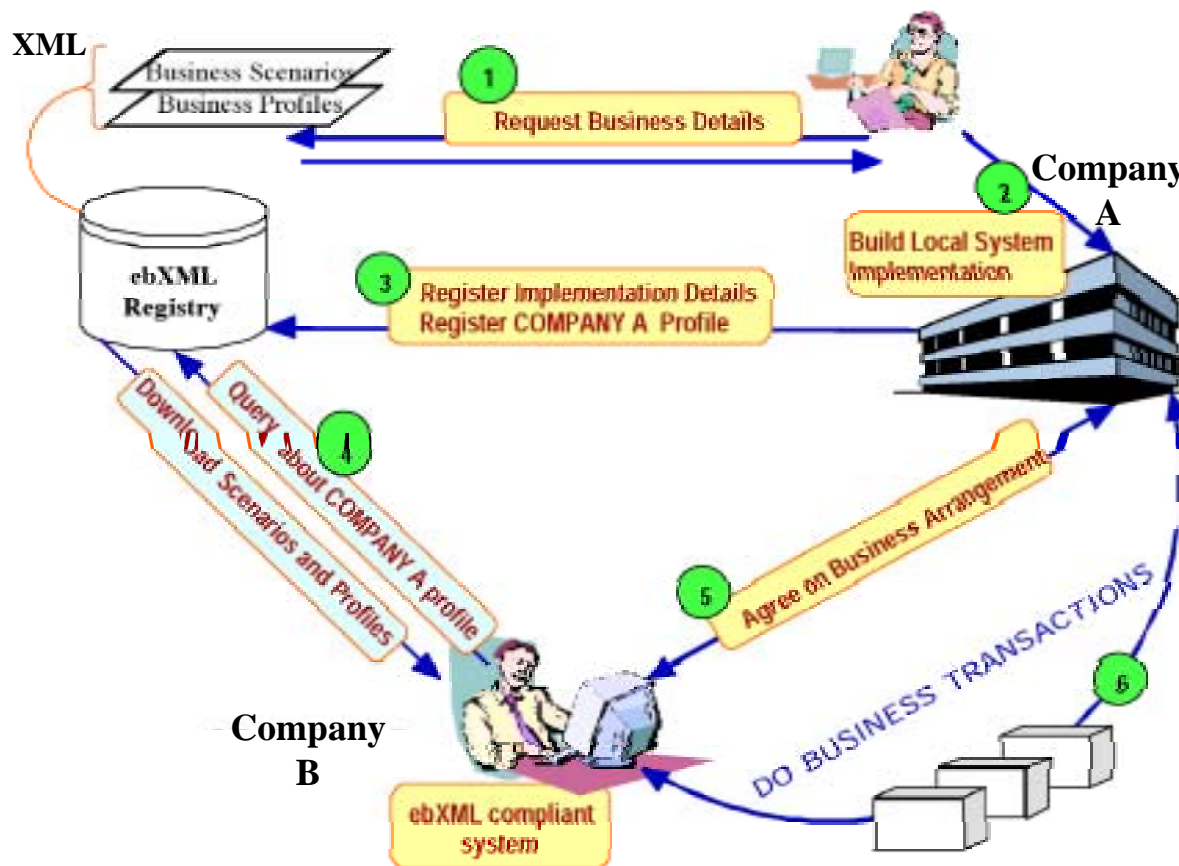
- Standardization and popularization of middleware have solved integration problems across different hardware architectures and operating systems.
- Newly arising challenges: Mixture of Middlewares  
CORBA, Java, COM+, various Web Services, .NET, ...

**Total Integration**

**Cross-Middleware**  
- Integration  
- System Design



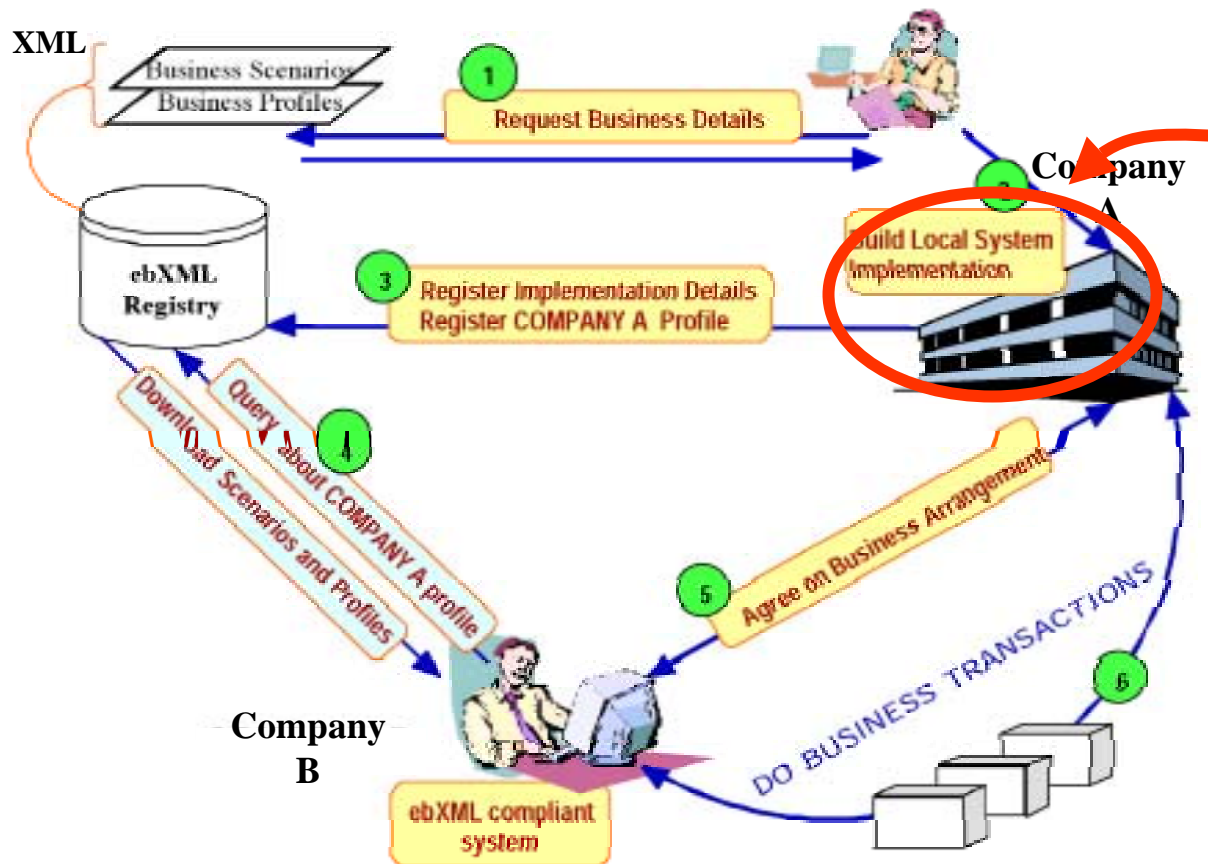
# Challenge: Integration across Middlewares



Let us consider about ebXML Web Services as an example of middleware.

(Note) from ebXML "Technical Architecture Specification" (v1.0.4)

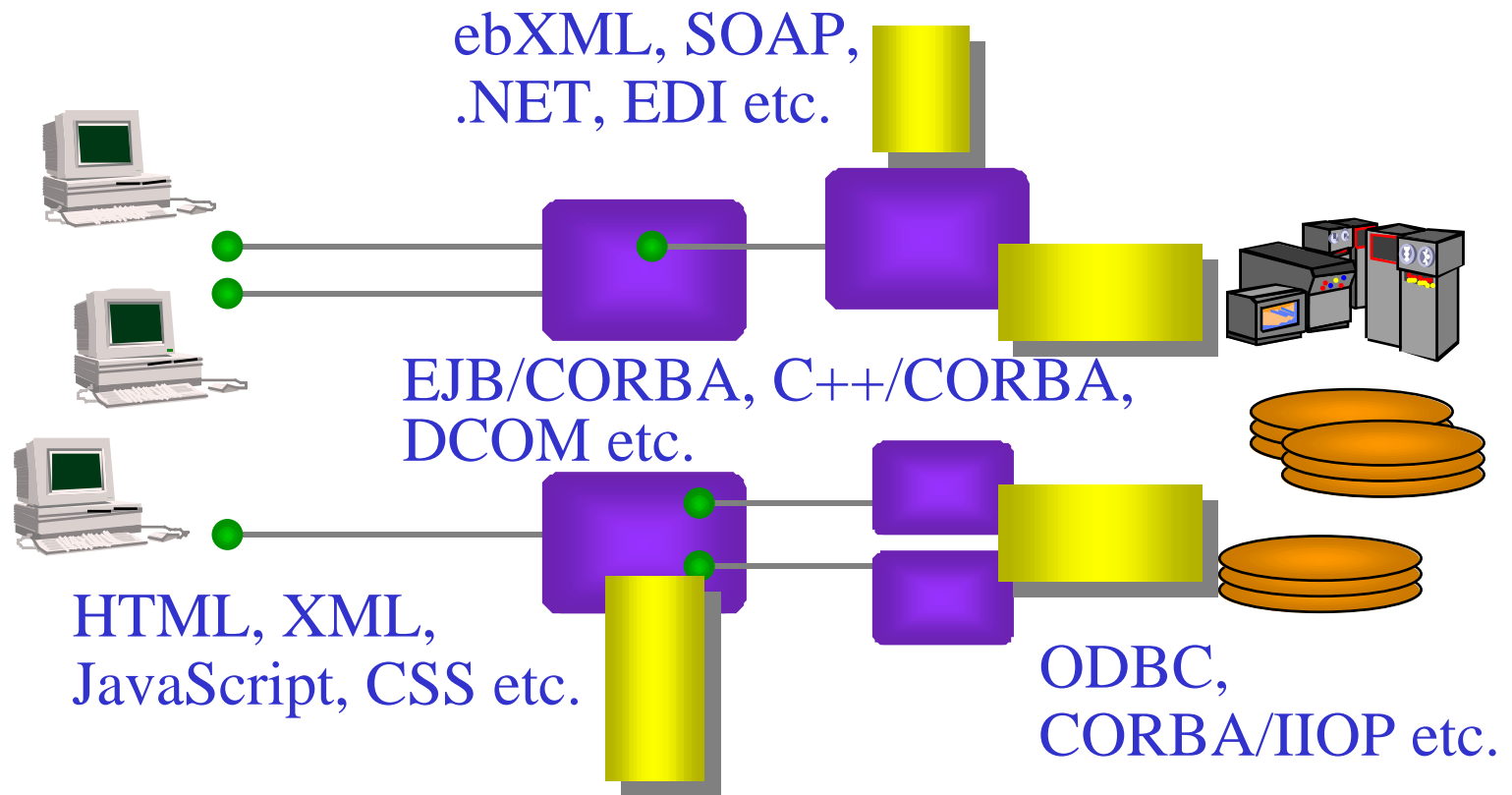
# Integration across Middlewares



**Integration Point between Web Services and other IT world**

- New development ?
- No. Integration to connect Web Service and existing enterprise system.

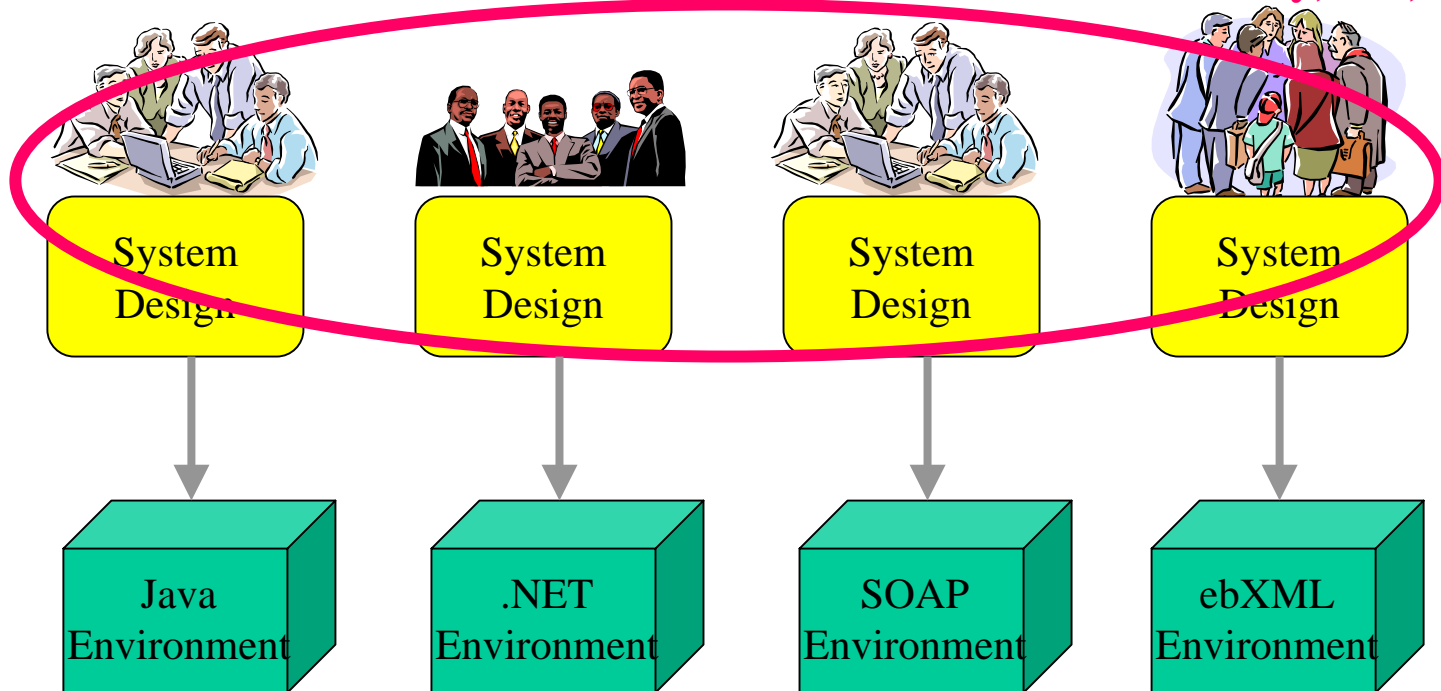
This means integration to connect Web service and in-house middleware environment is needed.



# The Other Challenge: System Design across Middlewares

Essentially the same system, but ...

System design, at least, should be done commonly, but, ...



# MDA Vision

- Cannot avoid co-existence of plural (middleware) platforms
- MDA
  - Model Driven Architecture
- Platform independent system design
  - described using UML (Unified Modeling Language) in general
  - called as **PIM (Platform Independent Model)**
- From PIM, system design for each platform is driven
  - called as **PSM (Platform Specific Model)**
- From PSM, actual skeleton of codes is driven

Note: What is a "Model"?

# Notes: What is a "Model" ?

## ● "Model" mentioned here means?

- Something showing concepts
- Scale-downed description or presentation
- Existing thing presenting some characteristics to design a new thing
- **System Design** - - - - precisely speaking:  
Design documents/information to create actual systems

## ● UML(Unified Modeling Language)

- Standardized notation to describe system design
  - Logical module structure => Class Diagram
  - Status transition => Activity Diagram, Collaboration Diagram
  - etc.

# MDA Vision

● Cannot avoid co-existence of plural (middleware) platforms

● MDA

= Model Driven Architecture

Model =  
System Design

● Platform independent system design

➤ described using UML (Unified Modeling Language) in general

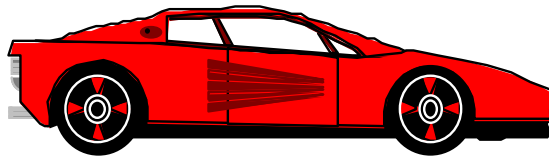
➤ called as **PIM (Platform Independent Model)**

● From PIM, system design for each platform is driven

➤ called as **PSM (Platform Specific Model)**

● From PSM, actual skeleton of codes is driven

# Simple Example



**<Car>**  
**<doors> 2</ doors>**  
**<colour> red</ colour>**  
**</Car>**



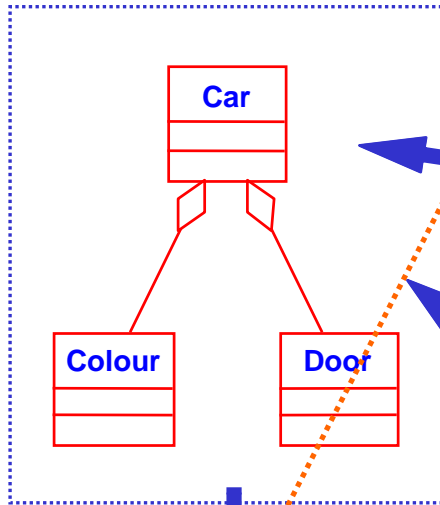
**<auto doors="2" colour="red"/>**

**public class Car {**  
**public colour colour;**  
**public int door#; }**

# Example in XMI

**Precise mapping rule is defined in OMG XMI Standard.**

**PIM (denoted by UML)**



```

<Class>
  <Name> Car</Name>
</Class>
Model in XMI
    
```

```

<element name="Car"/>
<!ELEMENT Car(Colour*, Door*)>
XMI Schema & DTD
    
```

```

Class Car
{
  Colour colour
  Door door
}
Java, IDL
    
```

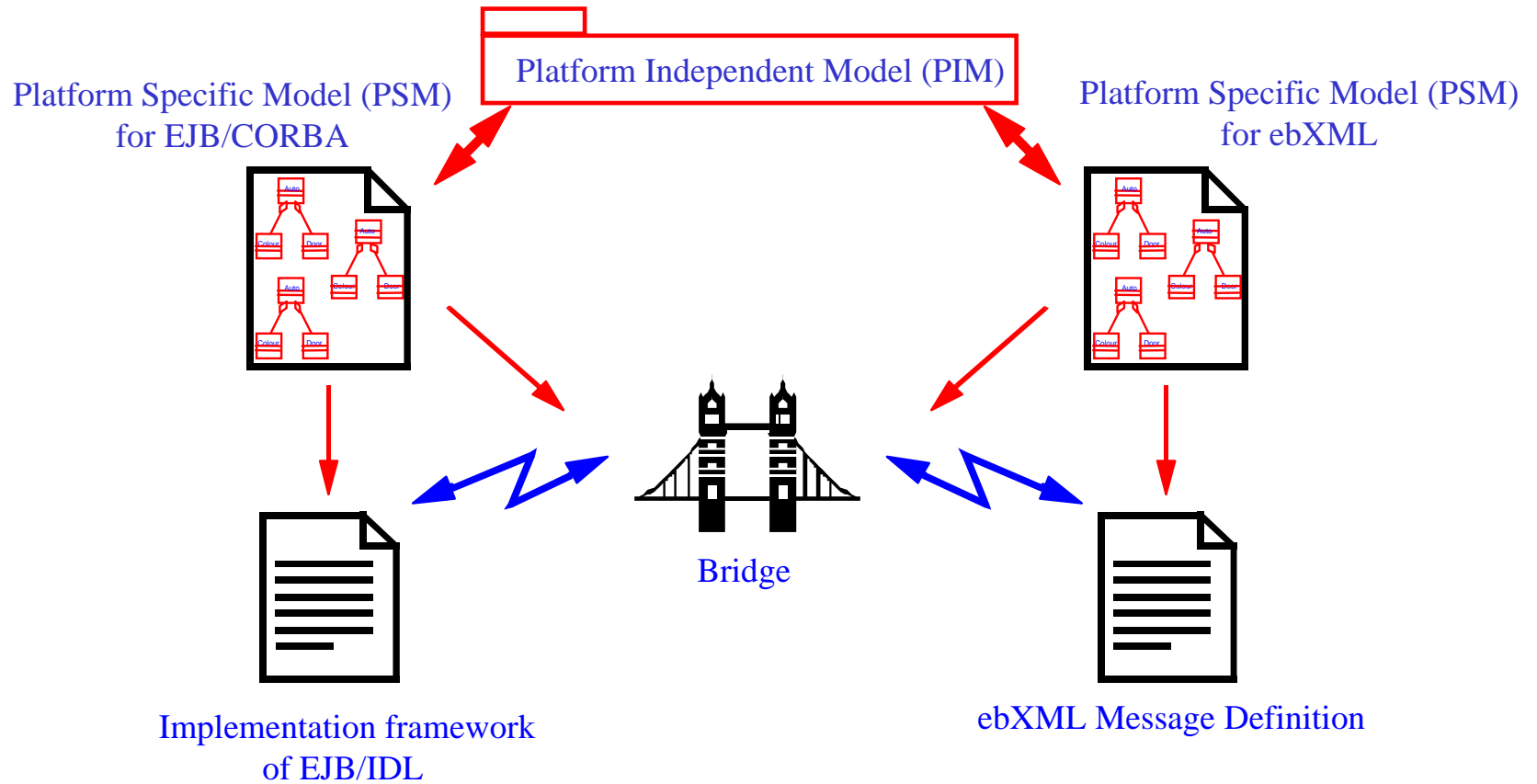
```

<Car>
  <Colour>Red</Colour>
  <Door>2</Door>
</Car>
XMI doc.
    
```

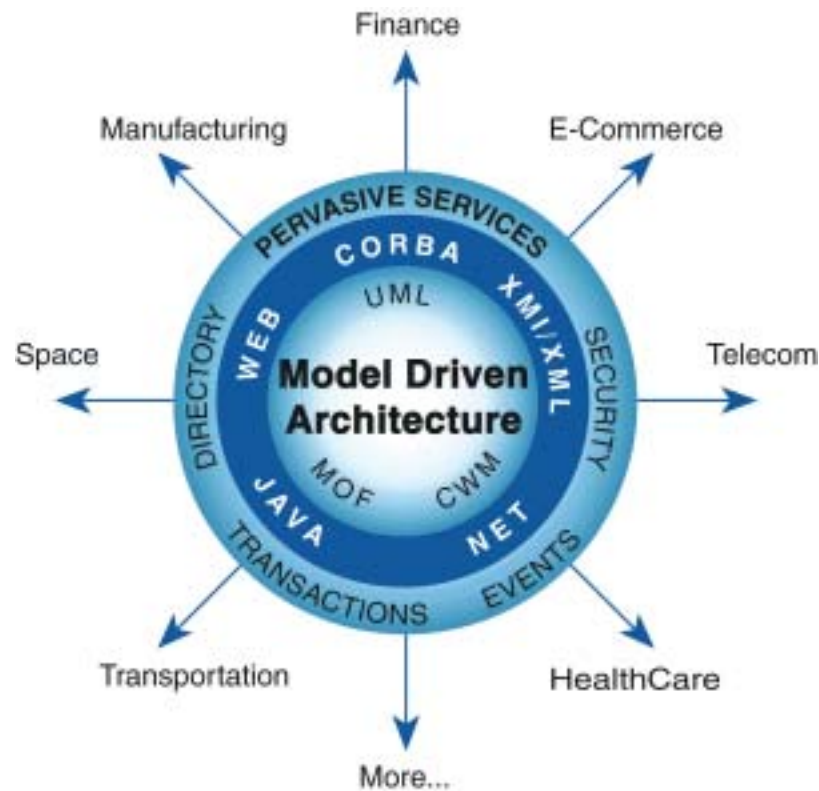
Exchange of Model

Exchange of XML Data

# Driven by Model




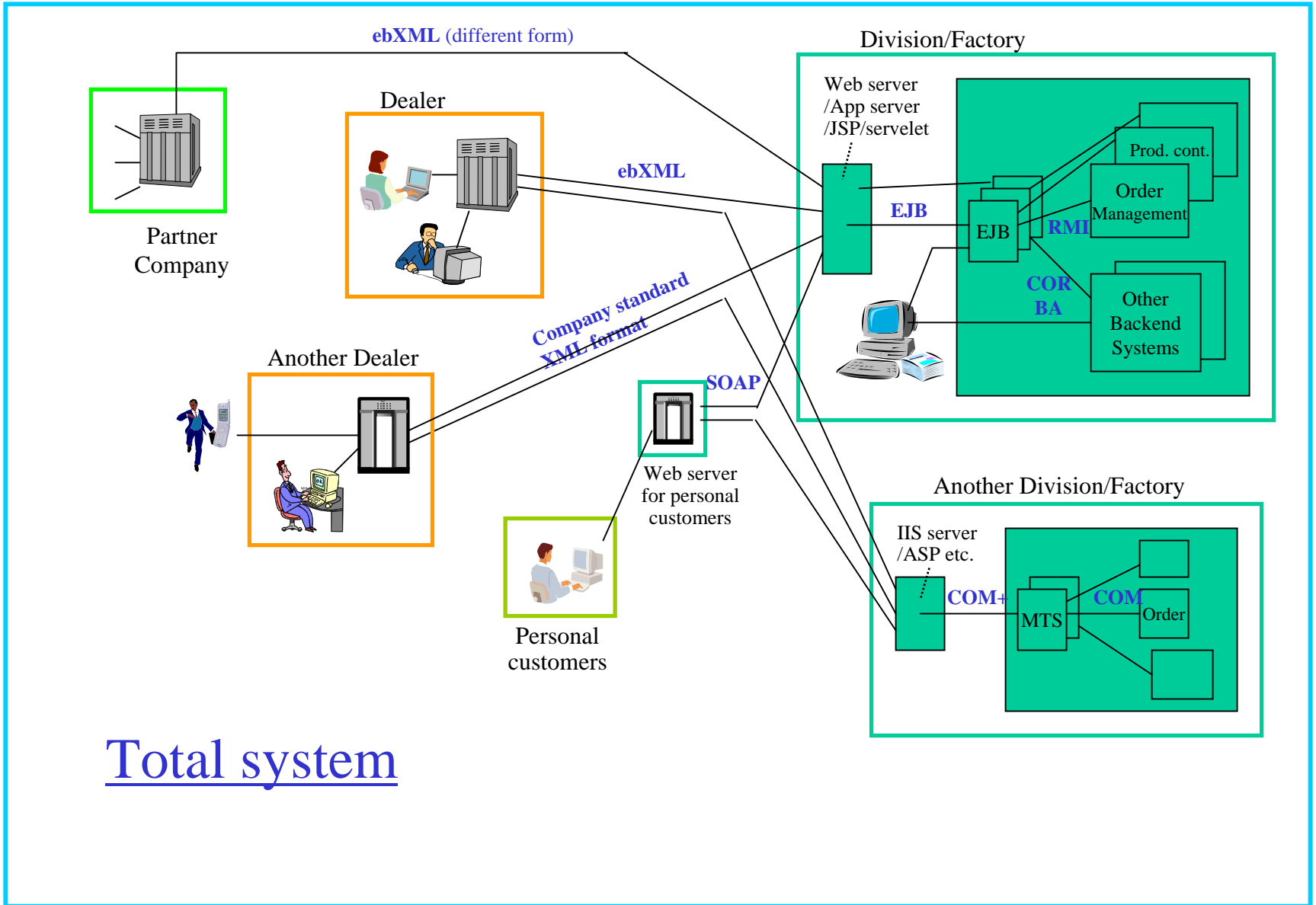
# MDA (Model Driven Architecture):

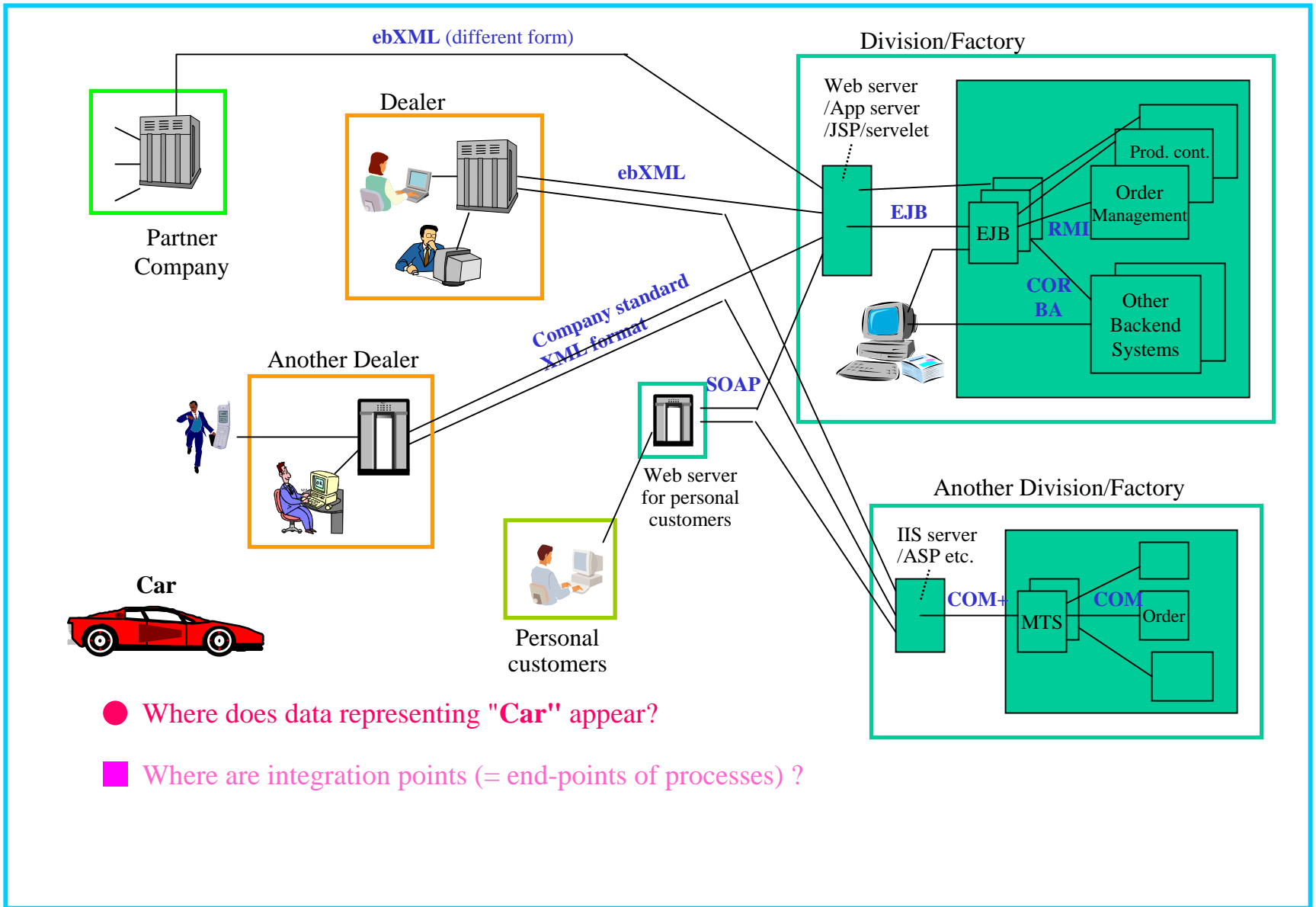


# MDA's Approach

# An Example to show the MDA's Approach

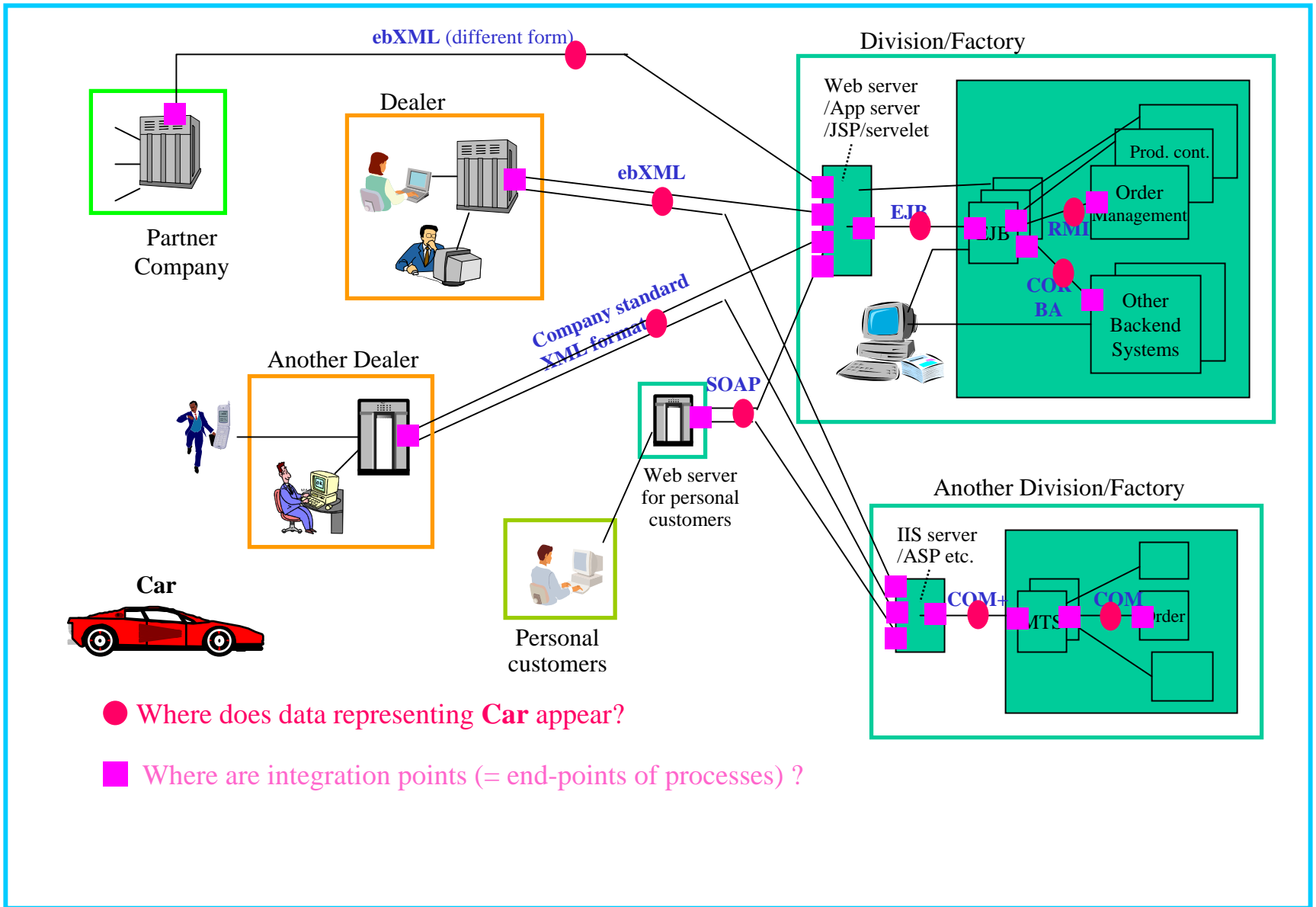
- Order/Sales system of "Car" (  )
  - Order option: Color and kind of door --- Toy car! (^\_^)
- Points to see:
  - Logically same data representing Car appears at various places.
  - "Integrate point" performing logically same process also appears at various points.
- MDA's approach
  - PIM (Platform Independent Model)
  - PSM (Platform Specific Model)

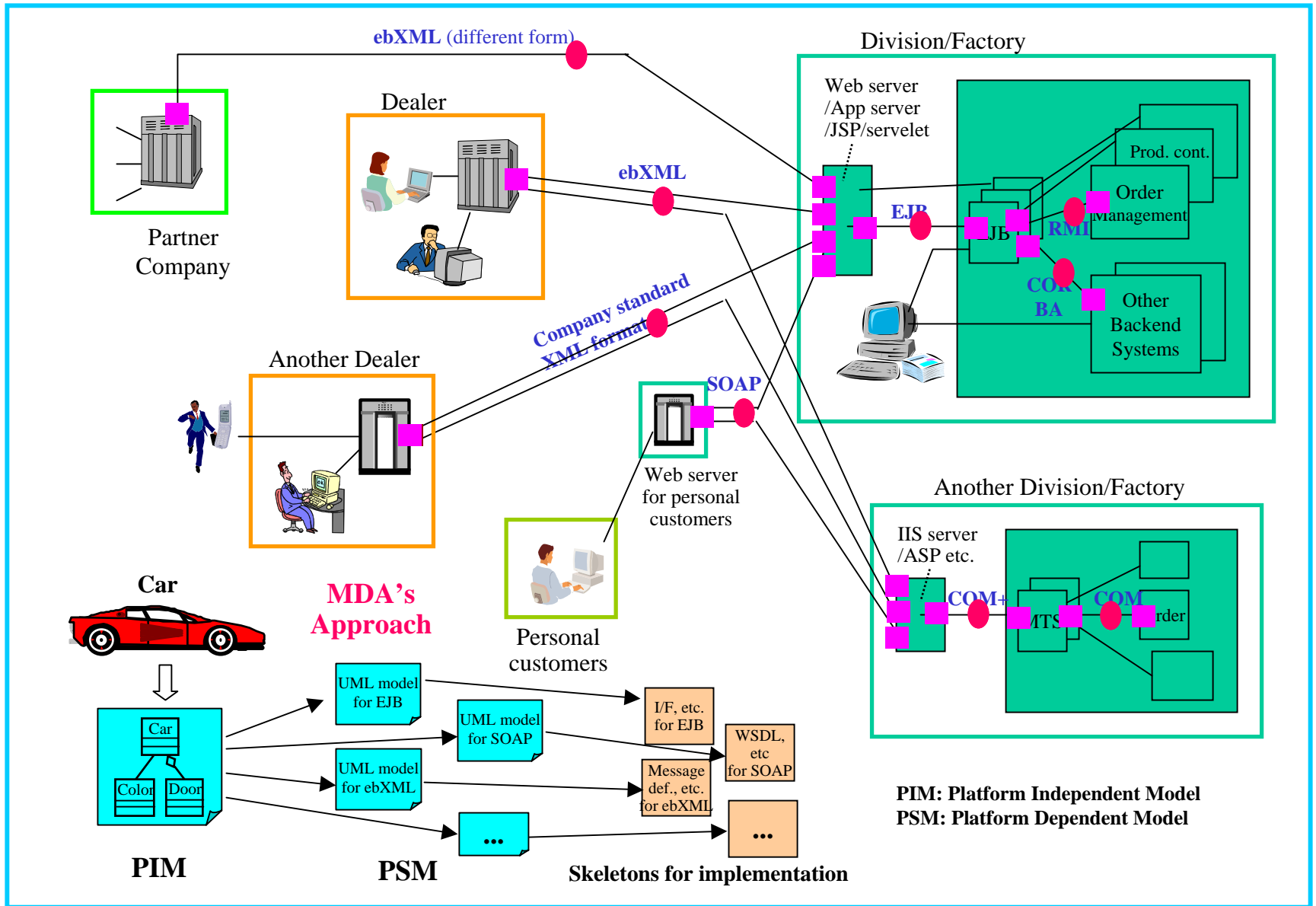


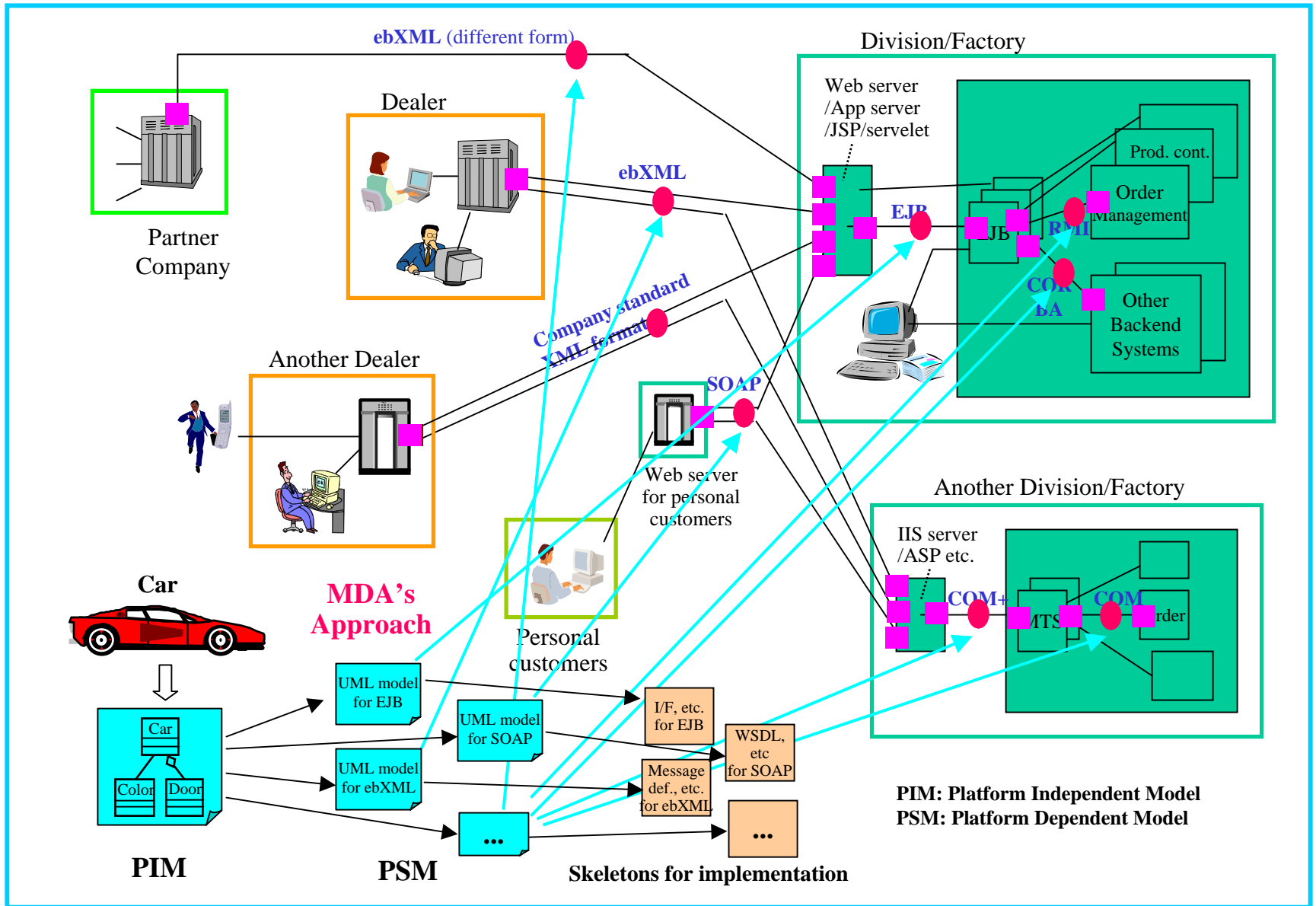


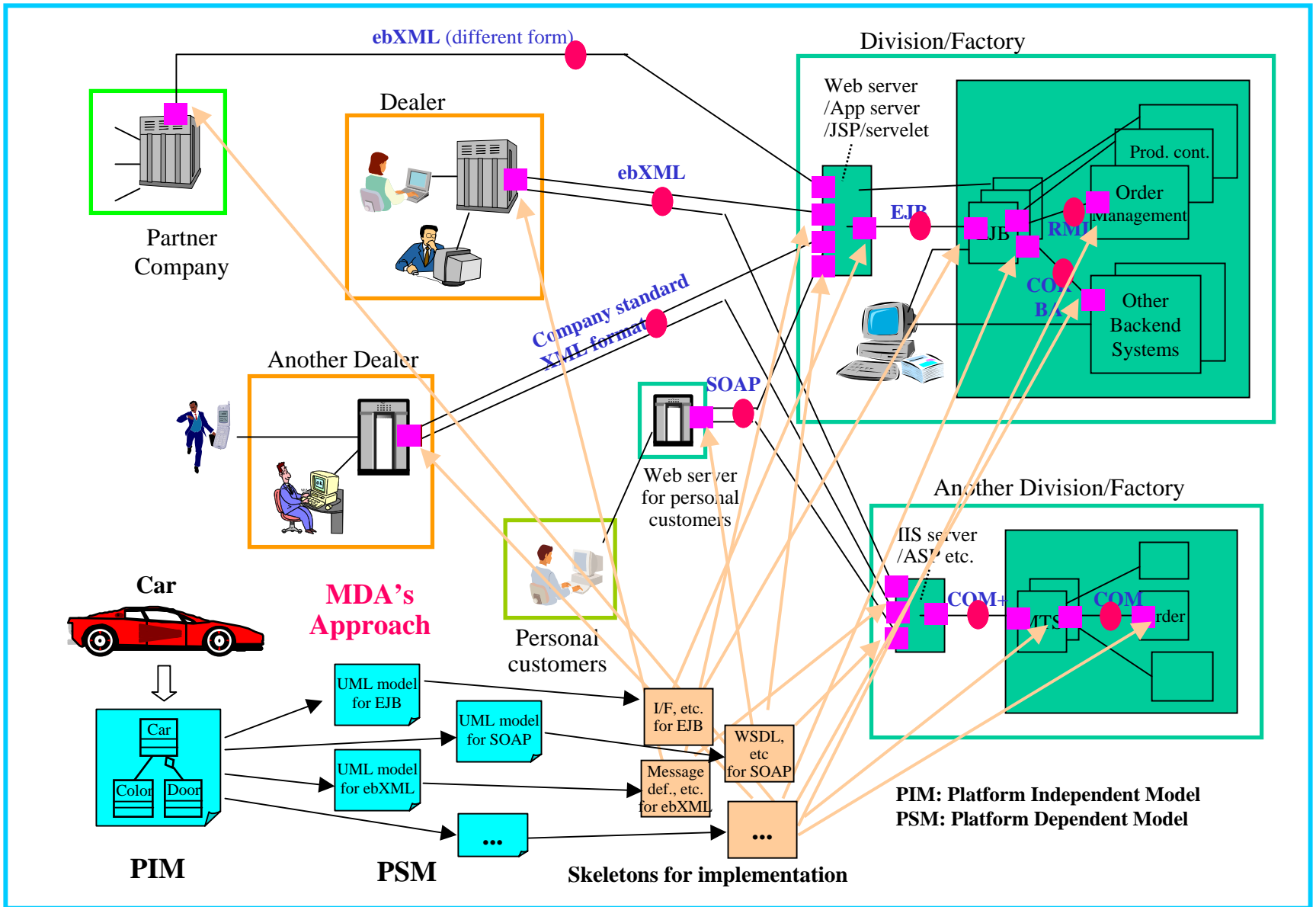
● Where does data representing "Car" appear?

■ Where are integration points (= end-points of processes) ?

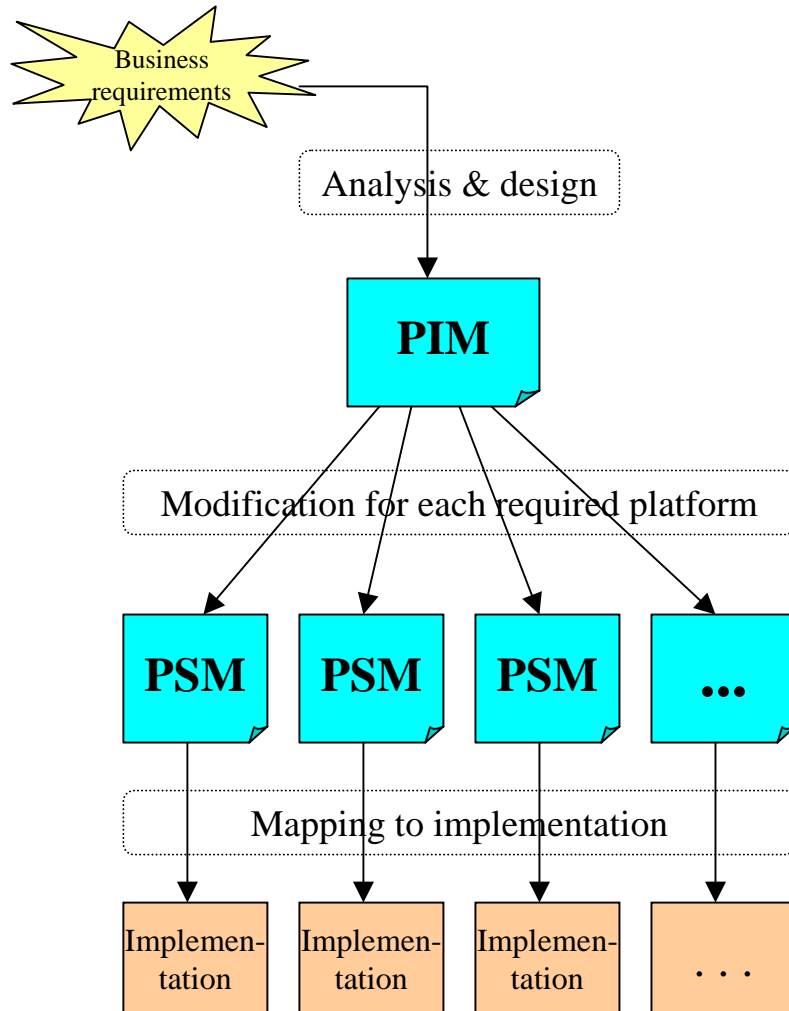








# MDA's Approach



- Model driven
- PIM and PSM
- PIM represents system design independently of platforms
- PSM represents implementation level design based on a particular platform specific characteristics
- Mappings:
  - PIM => PSM
  - PSM => Implementation
- Flexible development process and life cycle:
  - PSM => PIM
  - PIM => PIM, PSM => PSM
  - (Implementation => PSM)

# Example of PIM and PSM

## A simple order/response system

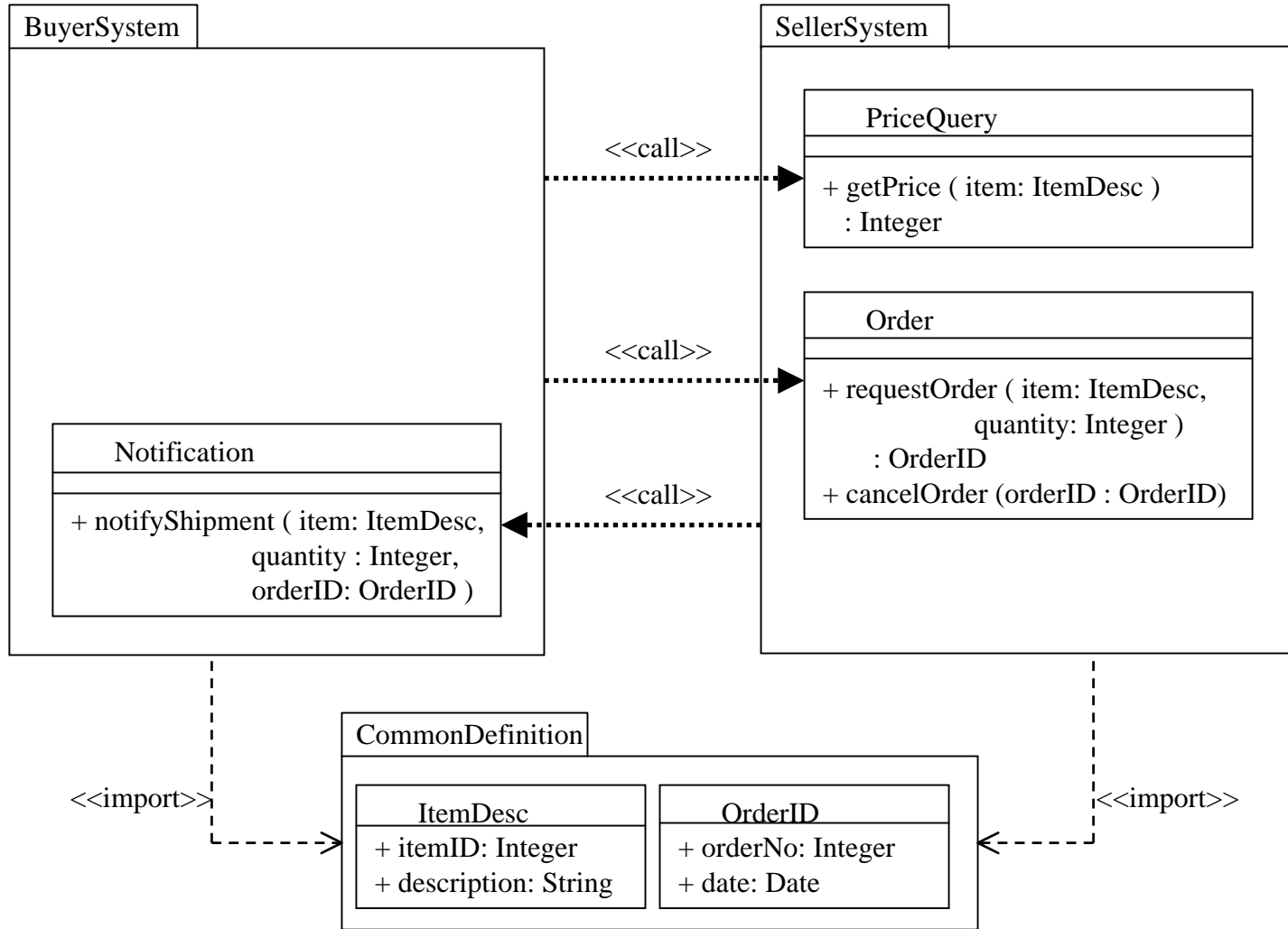
- Query of price (PriceQuery)
- Ordering (Order)
- Shipment notification (Notification)

## PSM

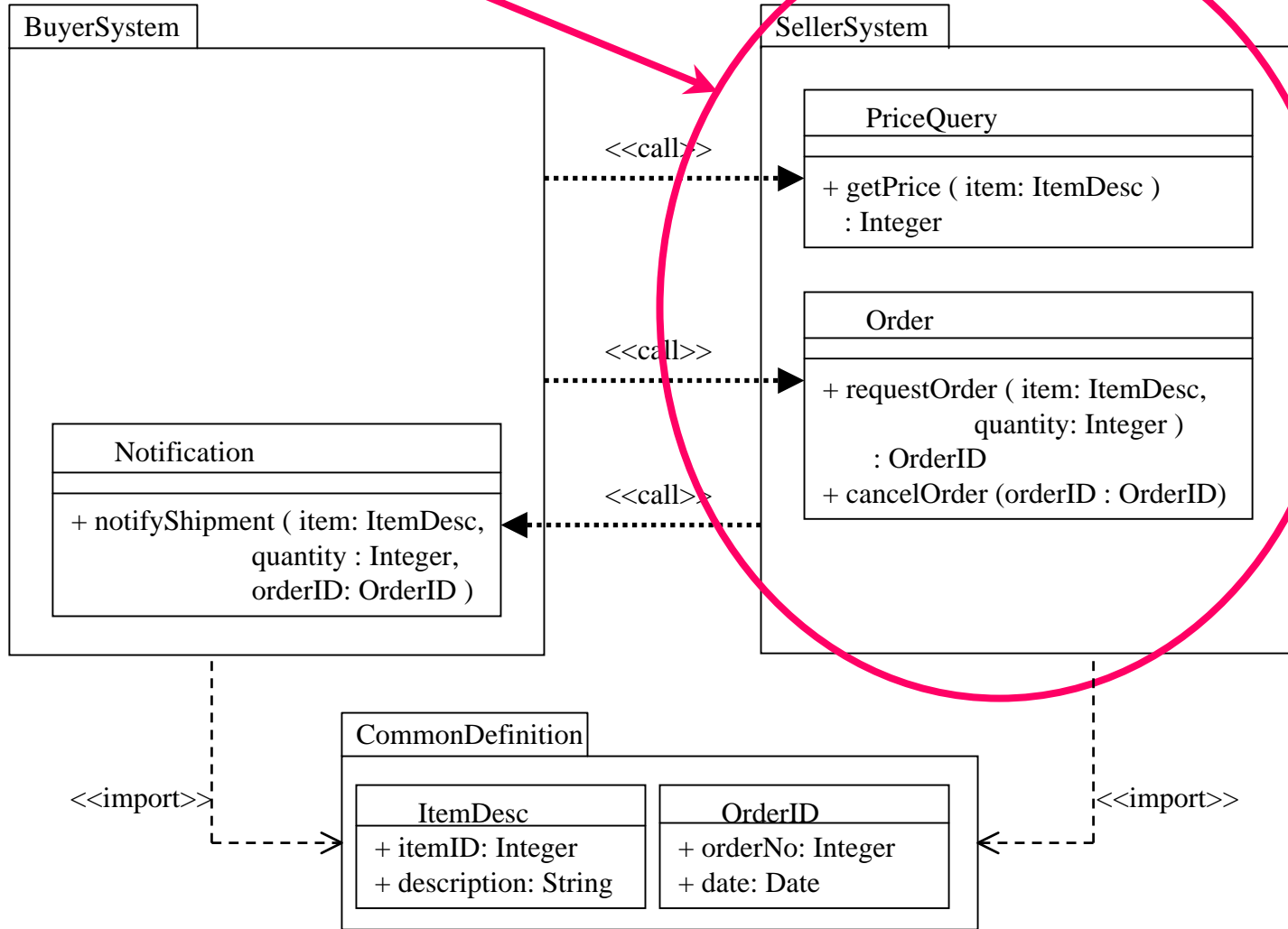
- EJB mapping example
  - assuming intra-trade in an enterprise.
- SOAP mapping example
  - Web Service; assuming inter-enterprise trade.

Note: PIM: Platform Independent Model, PSM: Platform Specific Model

# Example of PIM

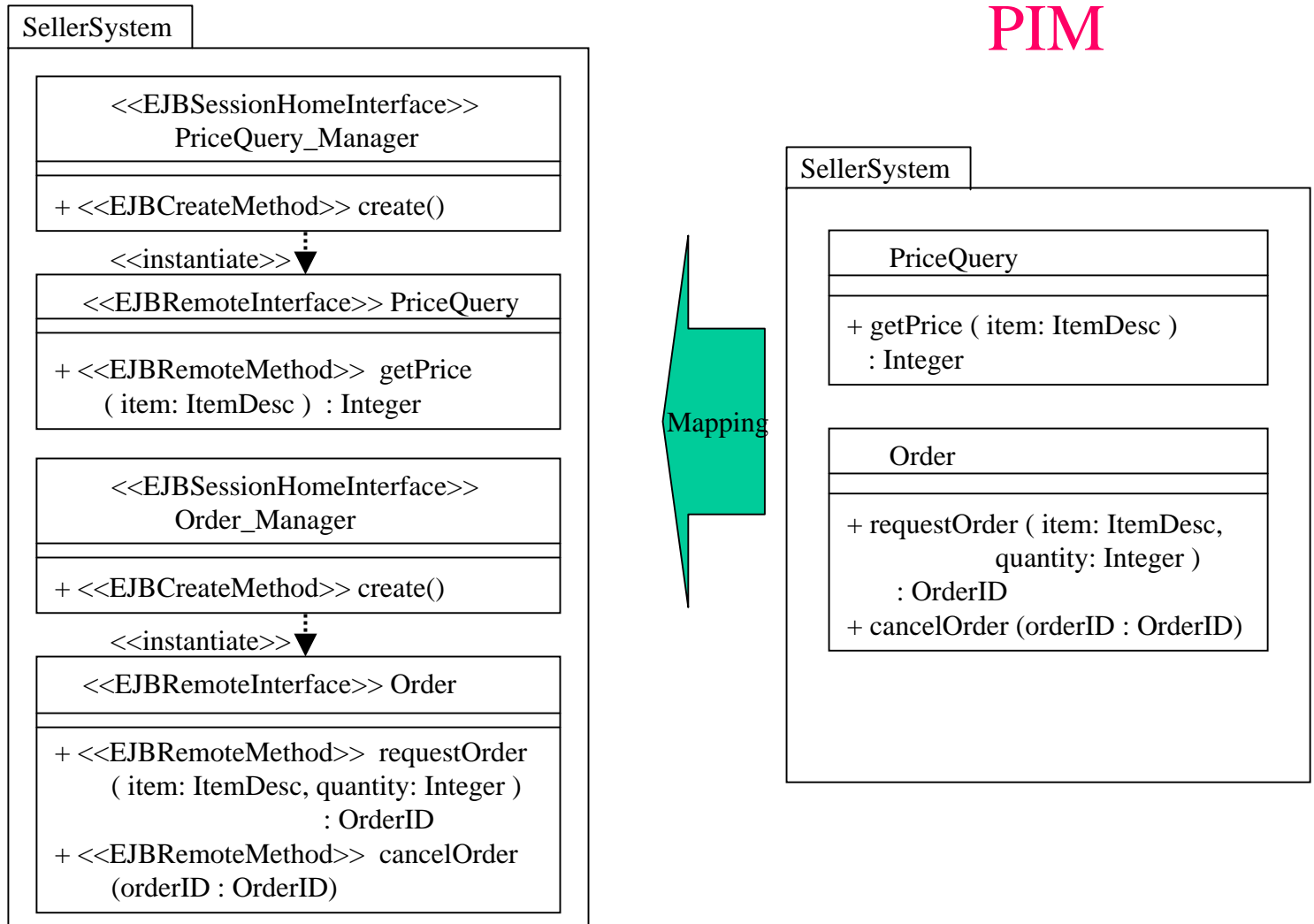


# Example of PSM (showing this portion)



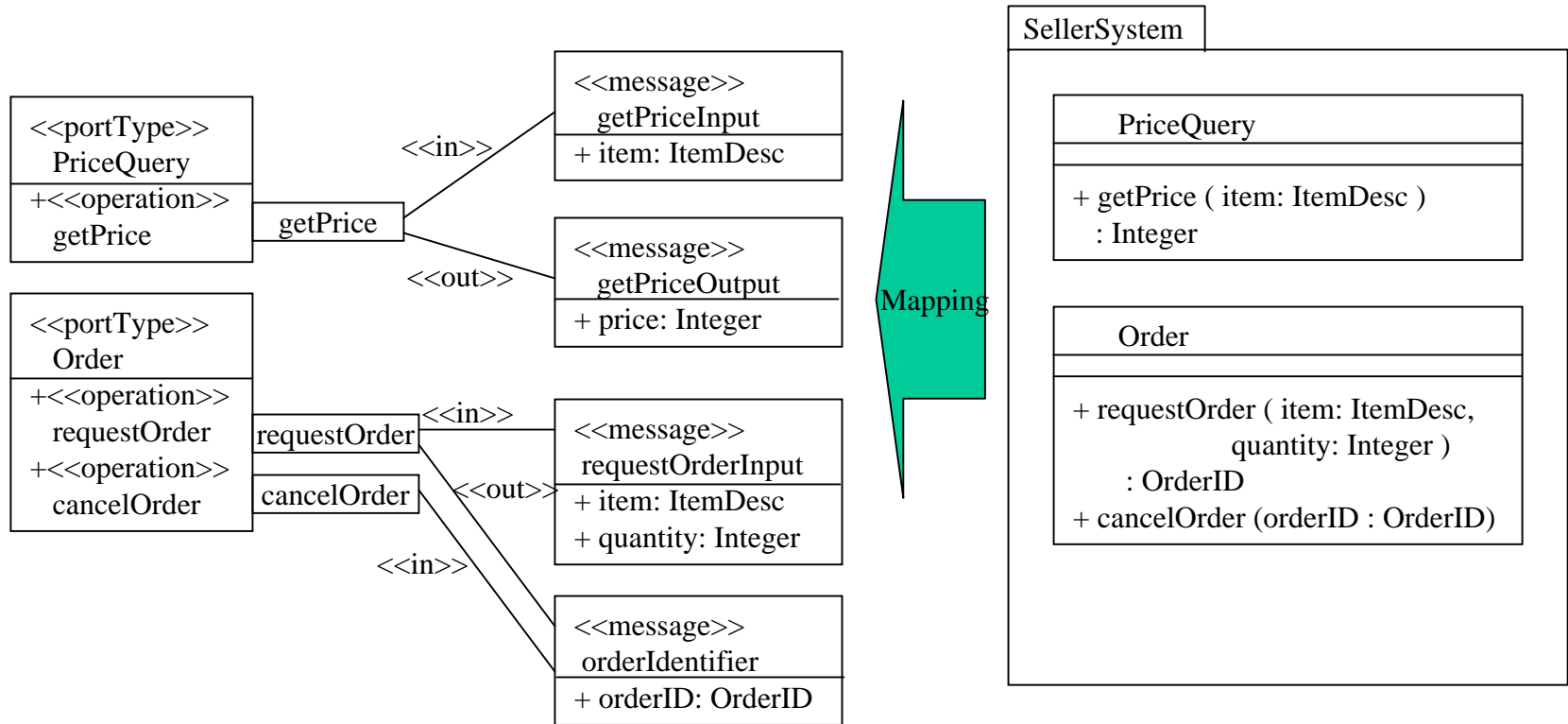
# PSM (for EJB)

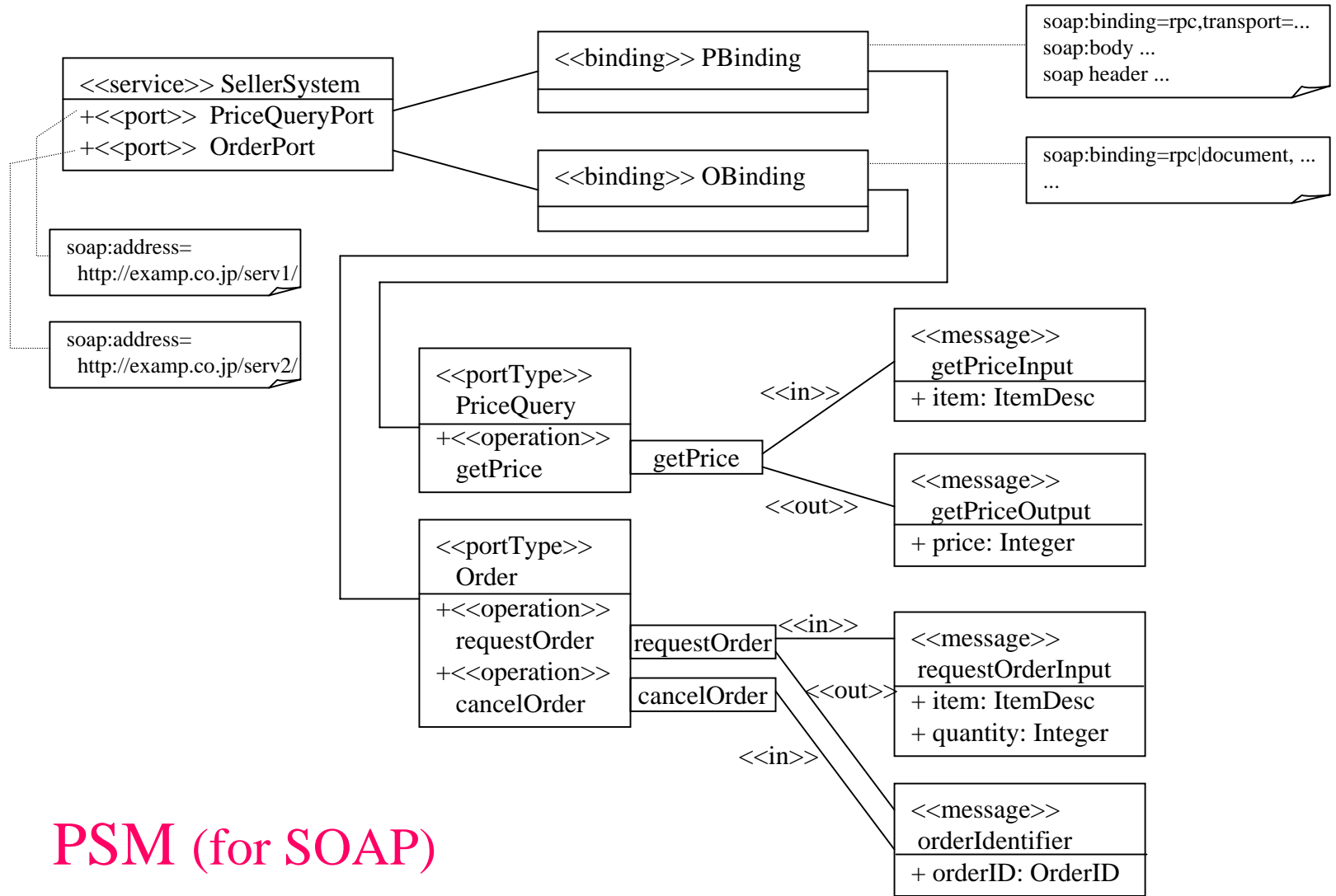
# PIM



# PSM (for SOAP)

# PIM





# PSM (for SOAP)

## WSDL

```

<definitions name="uri-BuySellSystem" ...
  xmlns:cd="uri-CommonDefinition" ...>
  <import namespace="uri-CommonDefinition"/>

  <message name="getPriceInput">
    <part name="item" element="cd:ItemDesc"/>
  </message>
  <message name="getPriceOutput">
    <part name="price" element="int"/>
  </message>
  <message name="requestOrderInput">
    <part name="item" element="cd:ItemDesc"/>
    <part name="quantity" element="int"/>
  </message>
  <message name="orderIdentifier">
    <part name="orderID" element="cd:OrderID"/>
  </message>

  <portType name="PriceQuery">
    <operation name="getPrice">
      <input message="getPriceInput"/>
      <output message="getPriceOutput"/>
    </operation>
  </portType>
  <portType name="Order">
    <operation name="requestOrder">
      <input message="requestOrderInput"/>
      <output message="orderIdentifier"/>
    </operation>
    <operation name="cancelOrder">
      <input message="orderIdentifier"/>
    </operation>
  </portType>

```

```

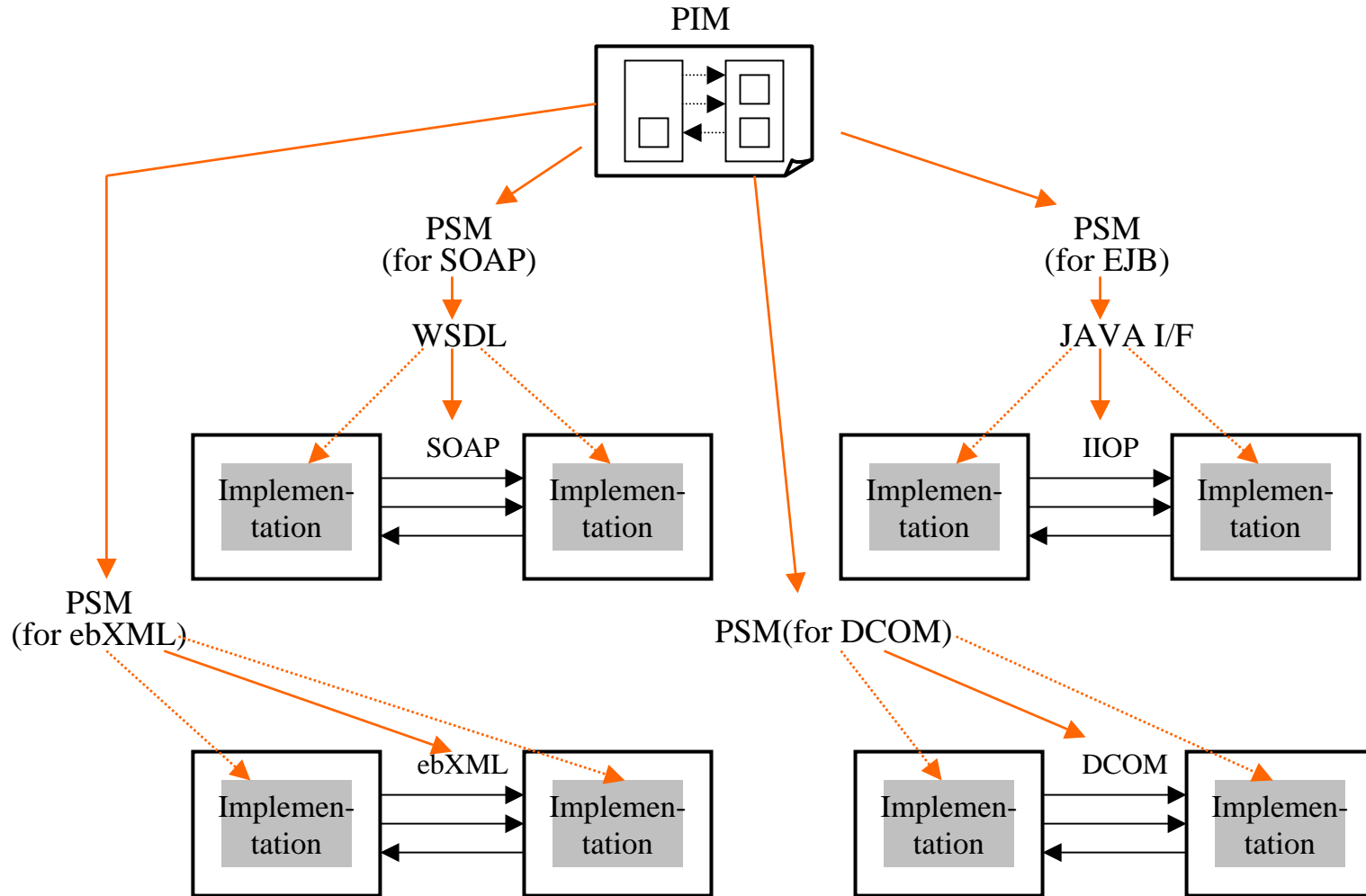
<binding name="PBinding" type="PriceQuery">
  <soap:binding style="rpc"
    transport="schemas.xmlsoap/org/soap/http"/>
  <operation name="getPrice">
    <input>
      <soap:body use="encoded" namespace= ... />
      <soap:header ... />
    </input>
    <output>
      ...
    </output>
  </operation>
</binding>
<binding name="OBinding" type="Order">
  <soap:binding style="rpc|document" transport=... />
  <operation name="requestOrder">
    ...
  </operation>
  <operation name="cancelOrder">
    ...
  </operation>
</binding>

<service name="SellerSystem">
  <port name="PriceQueryPort" binding="PBinding">
    <soap:address location="http://examp.co.jp/serv1"/>
  </port>
  <port name="OrderPort" binding="OBinding">
    <soap:address location="http://examp.co.jp/serv2"/>
  </port>
</service>

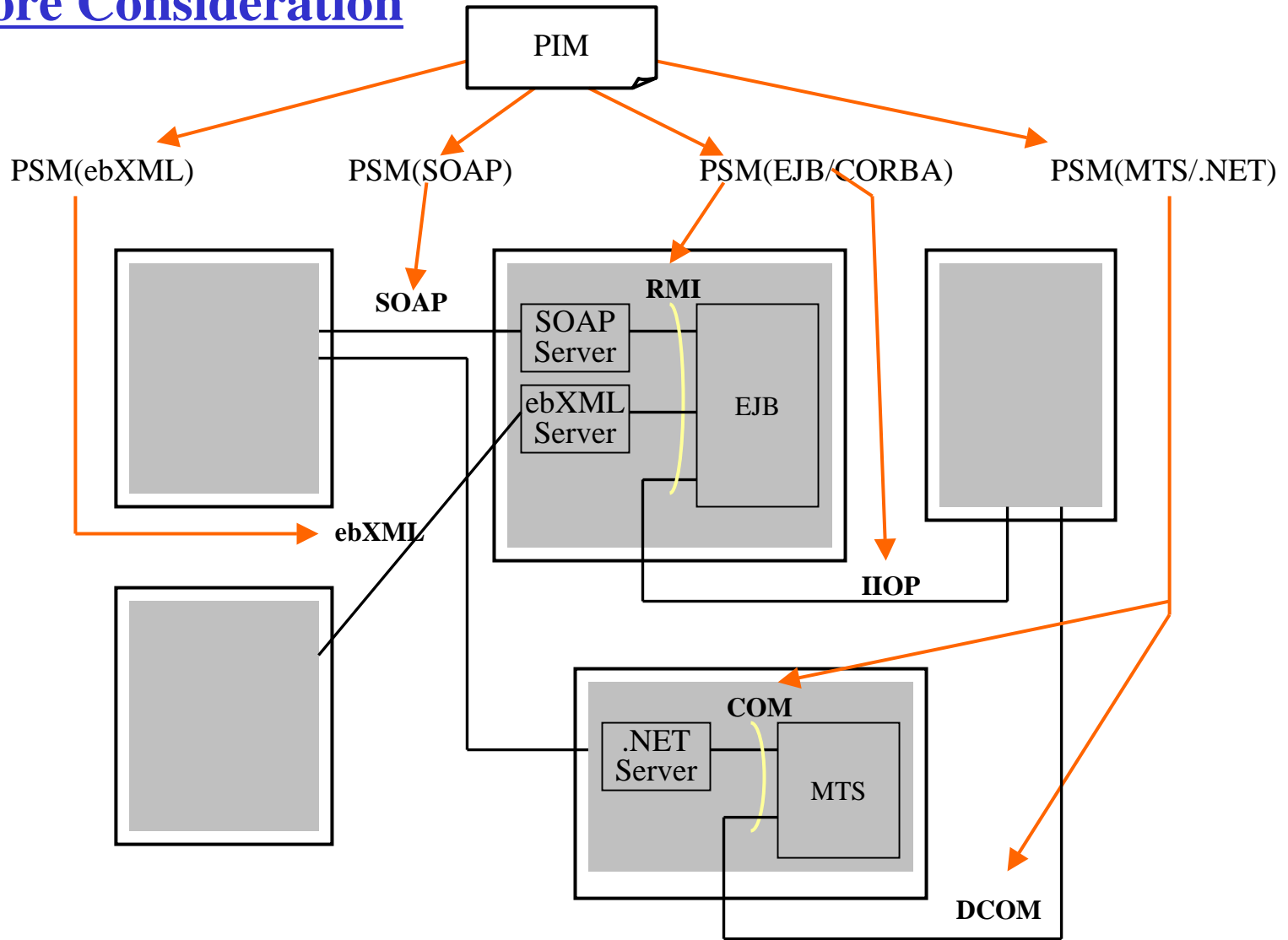
</definitions>

```

# Additional Consideration

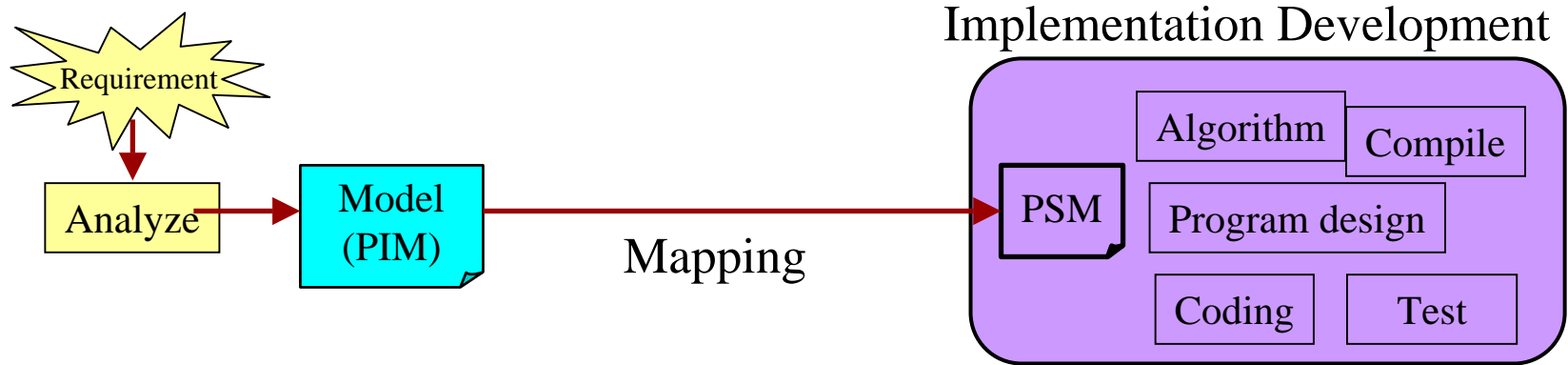


# More Consideration



# Toward Realization of MDA

# To Realize MDA



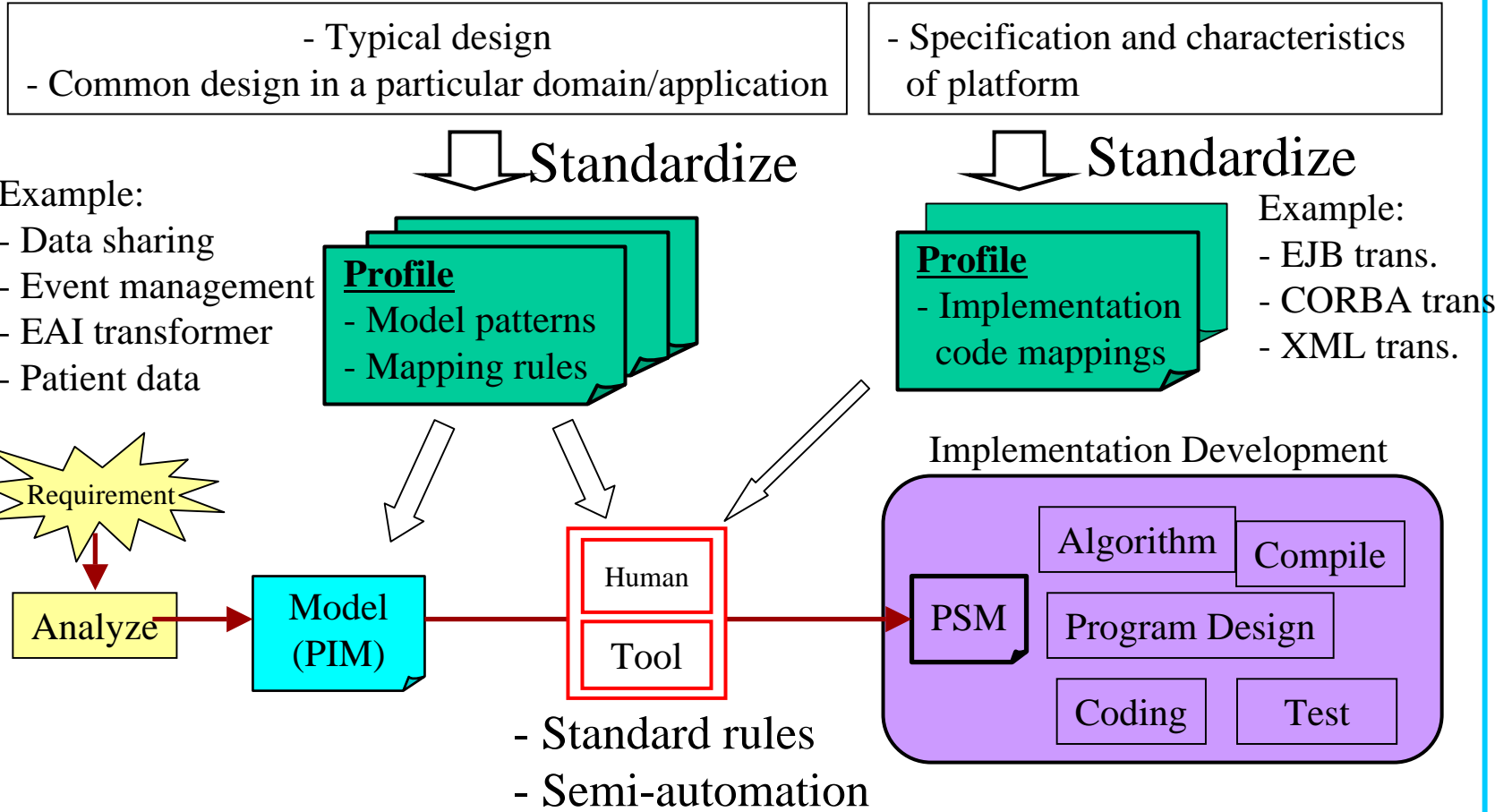
## ● How to map PIM to PSM?

- Do we depend on "Experience and talent of architects" or "Effort and tear of system developers" as we do now?
- "Secret" of MDA:
  - Define/standardize common/typical mapping rules
  - Aim semi-automatic PSM generation.

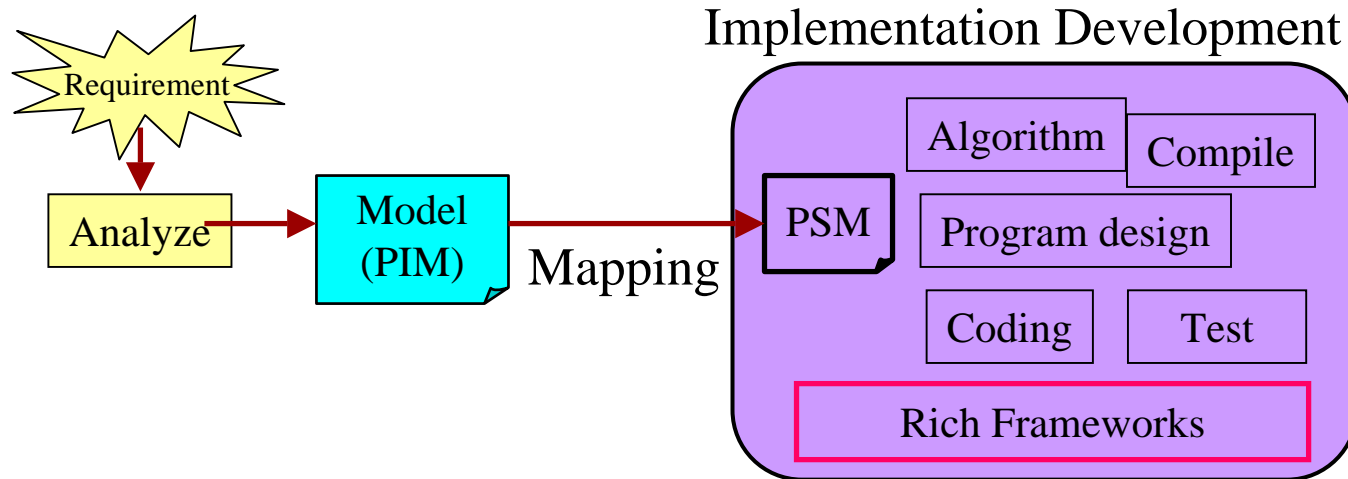
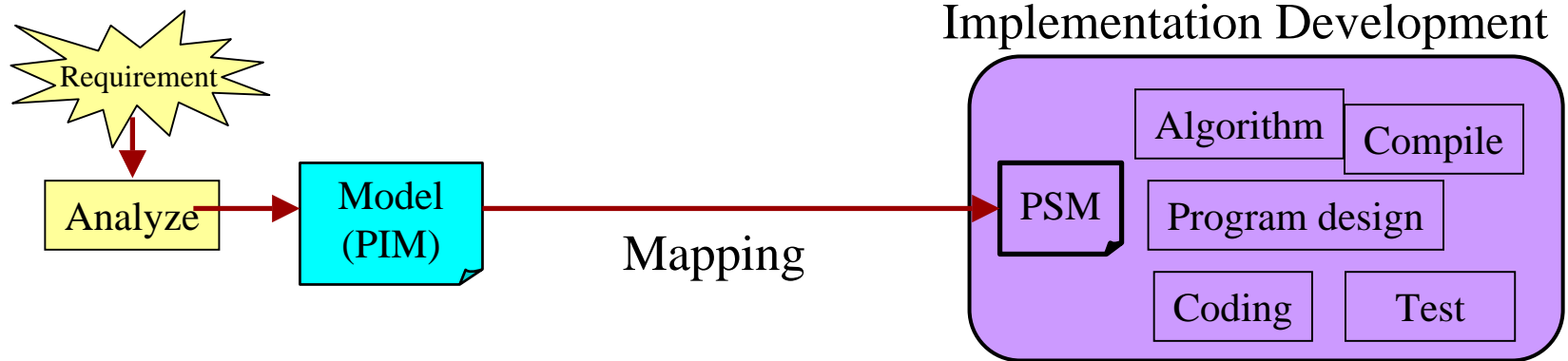
## ● Need to shorten the "Distance" between PIM and PSM

- Rich frameworks for application

# Mapping from PIM to PSM

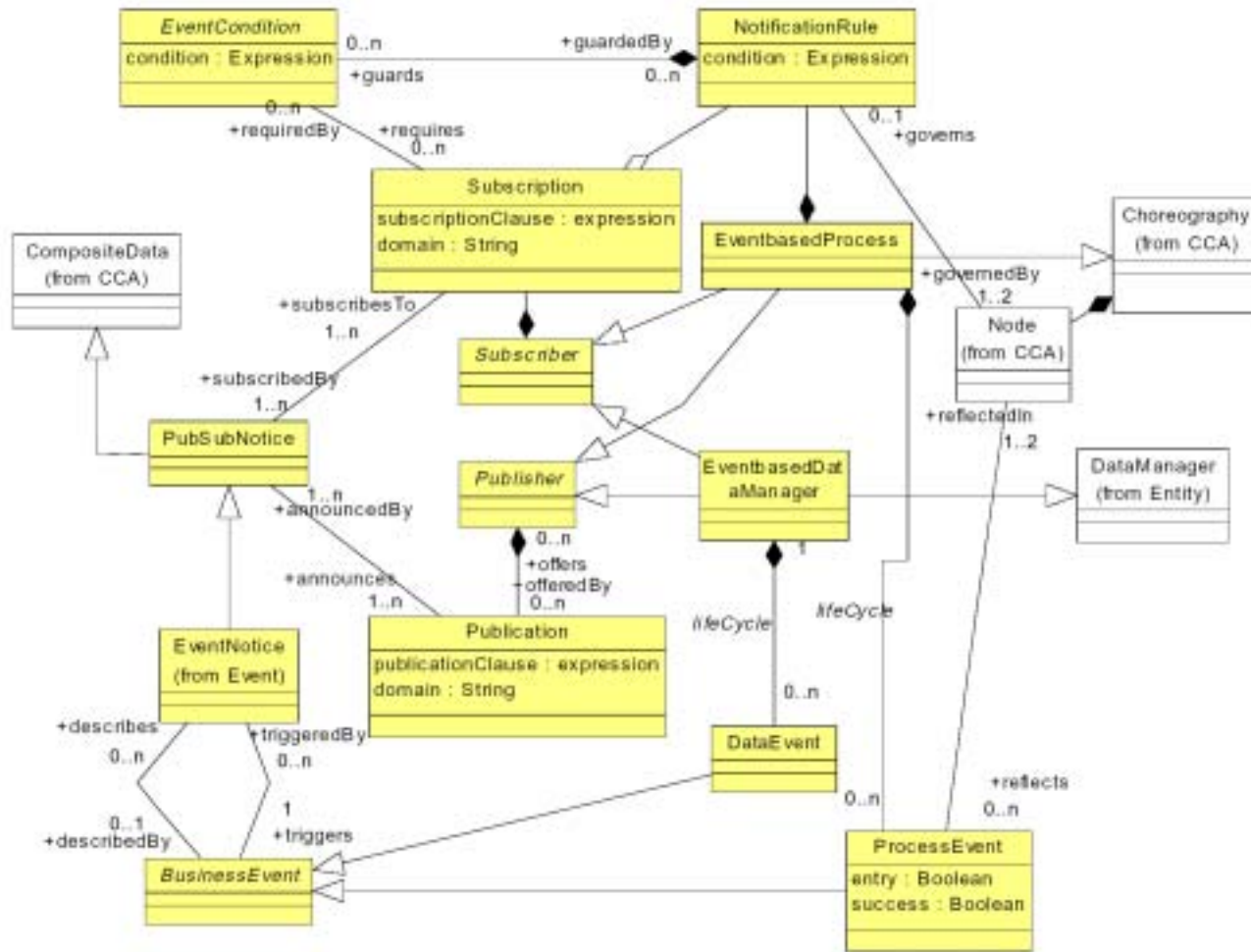


# Shorten the "Distance" between PIM and PSM



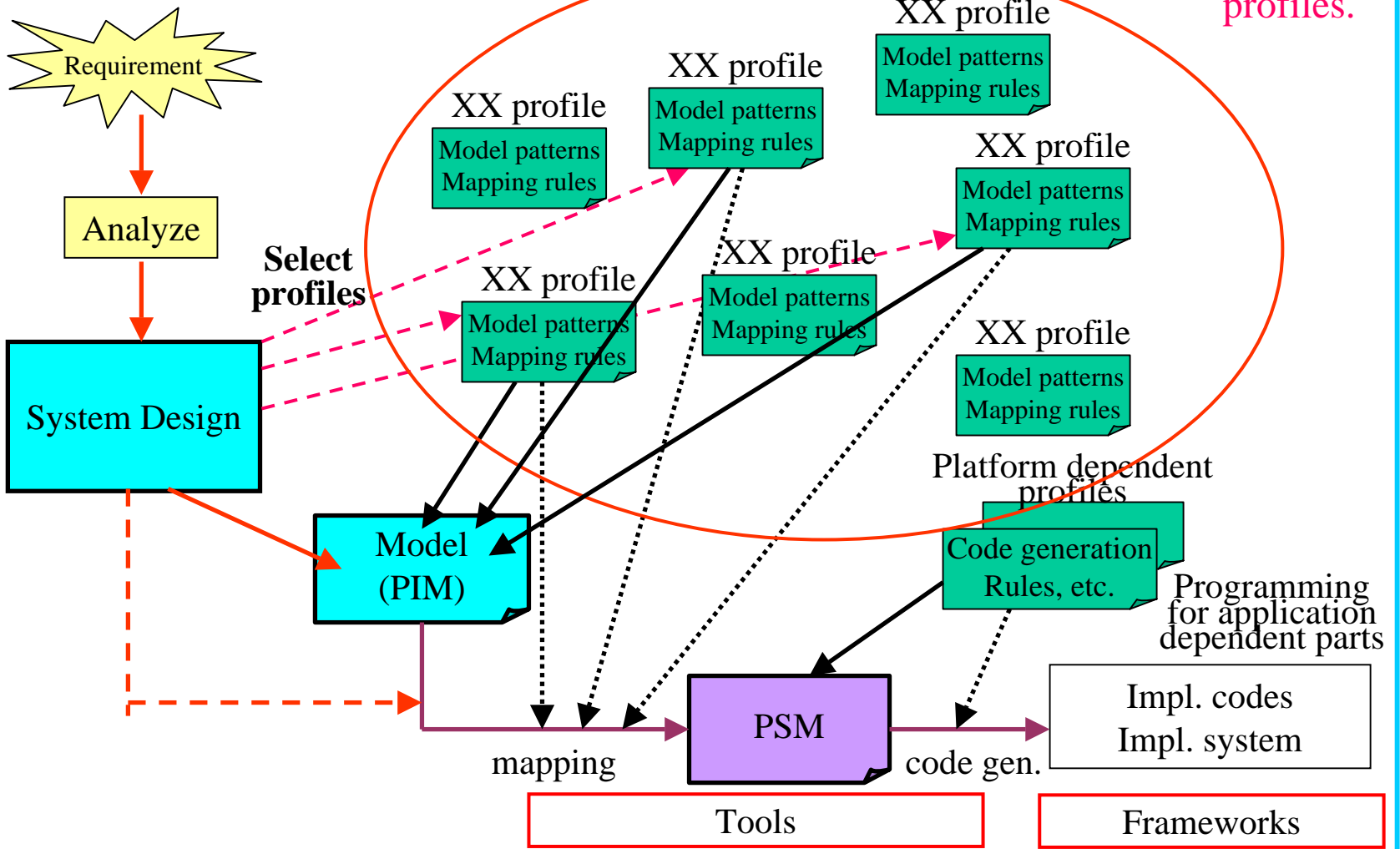
# Example of Profile

(from Event Profile in "UML Profile for EDOC")



# Ultimate Goal

The most important is to develop and heap standardized profiles.



## Note: Various profiles - already standardized, in process, under discussion

### OMG(standardized)

- UML Profile for EAI (Enterprise Application Integration)
- UML Profile for EDOC (Enterprise Distributed Object Computing)
- UML Profile for Schedulability, Performance and Time
- UML Profile for CORBA

- CCA (Component Collaboration Architecture)
- Entities Profile
- Events Profile
- Business Process Profile
- Relationship Profile

### OMG(in process)

- UML Profile for Modeling Quality of Service and Fault Tolerance Characteristics and Mechanisms
- UML for Systems Engineering

### JCP(standardized)

- UML Profile for EJB (JCP)

### Others (discussing, topics, rumor)

- |                                           |                                      |
|-------------------------------------------|--------------------------------------|
| - UML Profile for WSDL                    | - UML Profile for .NET               |
| - UML Profile for XML Schema              | - UML profile for Interaction design |
| - UML Profile for Persistence Model       | - UML Profile for Database Design    |
| - UML Profile for Reverse Engineering     | - UML profile for hypermedia         |
| - UML Profile for Framework Architectures | - UML for Ontology Development       |
| - UML Profile for DCL                     | - UML profile for DAML               |
| - UML Profile for Business Modeling       | - UML Profile for Web applications   |
| - UML profile for Business Analysis       |                                      |

# Summary

# MDA Summary

- PIM and PSM
- Two kinds of mappings:  
PIM=>PSM and PSM=>Implementation
- For PIM creators, standardized application specific profiles are provided. Standard mappings to PSM are also defined.
- Aiming semi-automatic PIM=>PSM transformation and automatic PSM=>Implementation transformation.
- Directly connecting to actual implementation development, and/but system design is platform independent.
- The most important thing for realization of MDA is development of wide range of standardized profiles.

# Conclusion

- Middleware will continue to evolve and proliferate with emerging technologies.
  - CORBA, Java and .NET will also evolve.
  - Web Services will evolve.
  - New middleware may appear.
  - Users want their IT systems up to date using such state-of-art technologies.
- Business requirements will also evolve and need to be quickly implemented in enterprise IT systems.
- PIM and PSM should to be independently designed and developed corresponding to business evolution and technology evolution, without breaking consistency, and in the way improving development productivity.
- MDA is the Key to this vision.

*END*

Note: All names in this presentation, including company names and product names, are used identification purpose only, may be trademark or registered trademark of their respective holders.