

ceira

technologies inc.

Executable Enterprise Modeling with UML

Presented to:

**OMG Workshop on UML in the
Enterprise**

December 3-6, 2001

By:

Michael Latta

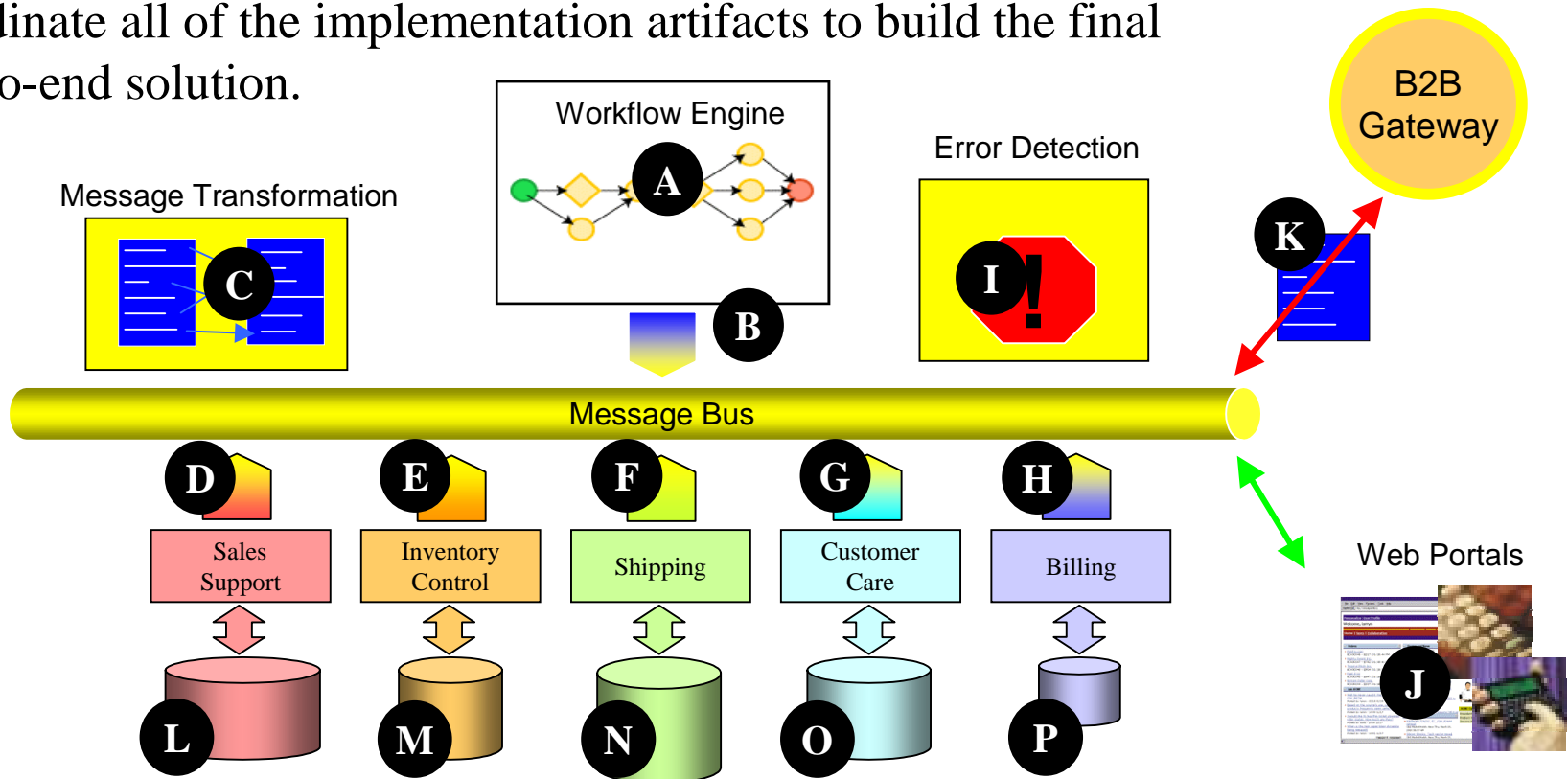
Yngvar D. Tronstad

- **Architecture**
- **OO Modeling**
 - **Structure**
 - **Behavior**
 - **State**
- **Push**
- **Execution Engine**
- **J2EE Apps Server**
- **JMS/JCA**
- **Lessons Learned**

Problem space and Challenges

An example integration environment, using TIBCO middleware

Configured piece by piece, it requires organizations to carefully coordinate all of the implementation artifacts to build the final end-to-end solution.



- **UML OO modeling support**
- **Collaborative Modeling Environment**
 - Different but related modeling domains/facets
 - Consistency checking
 - Multi- user support
 - Persistence
 - Versioning and Configuration Management
- **Model based design with validation**
- **Direct deployment and execution of the models**



- **The Model and the Implementation will not drift apart**
 - Automate the generation of the execution system from the model itself
 - Allows verification prior to deployment
- **Removes Ambiguity**
 - Forces semantics to be defined
 - Makes behavior more visible than when in code
- **Provides medium for Communication**
 - Business Owners and Designers can more easily relate
 - Highlights general process flow over details

What is an Executable Model?

- **A model complete enough to be executed**
- **Can be augmented by custom code, but only at the leaf level of the behavior tree**
- **Defines structure of business objects and their behavior, “smart objects” that can be distributed**
- **Embraces interface to external systems**
- **Scales to support the demands of an enterprise class customer**

How to provide an Executable Model

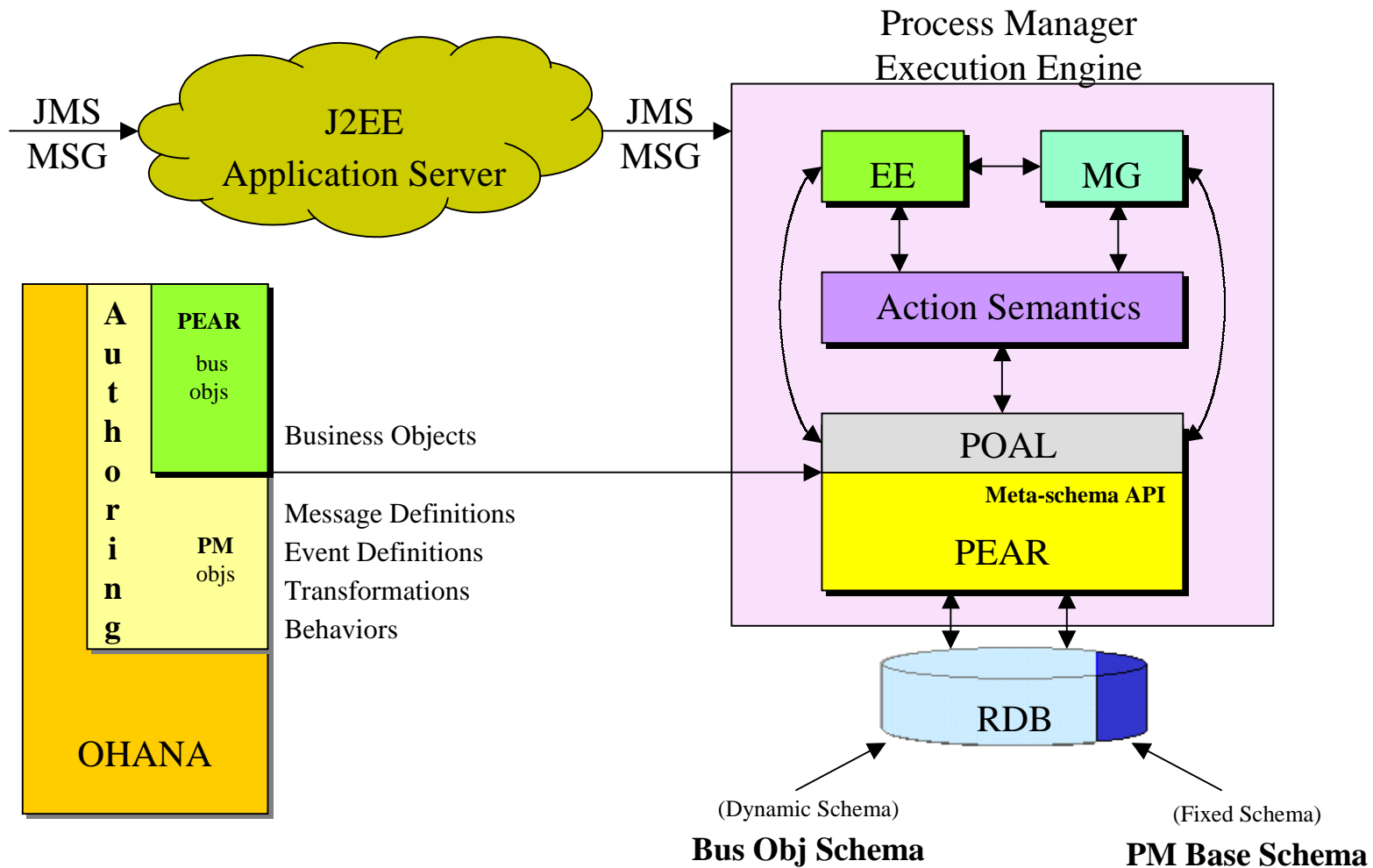
- **Start with UML**
- **Add missing semantics**
- **Clarify variations**
- **Generate database from structure model**
- **Execute behavior in process server**

Challenge in Executable Models: Add Missing Semantics

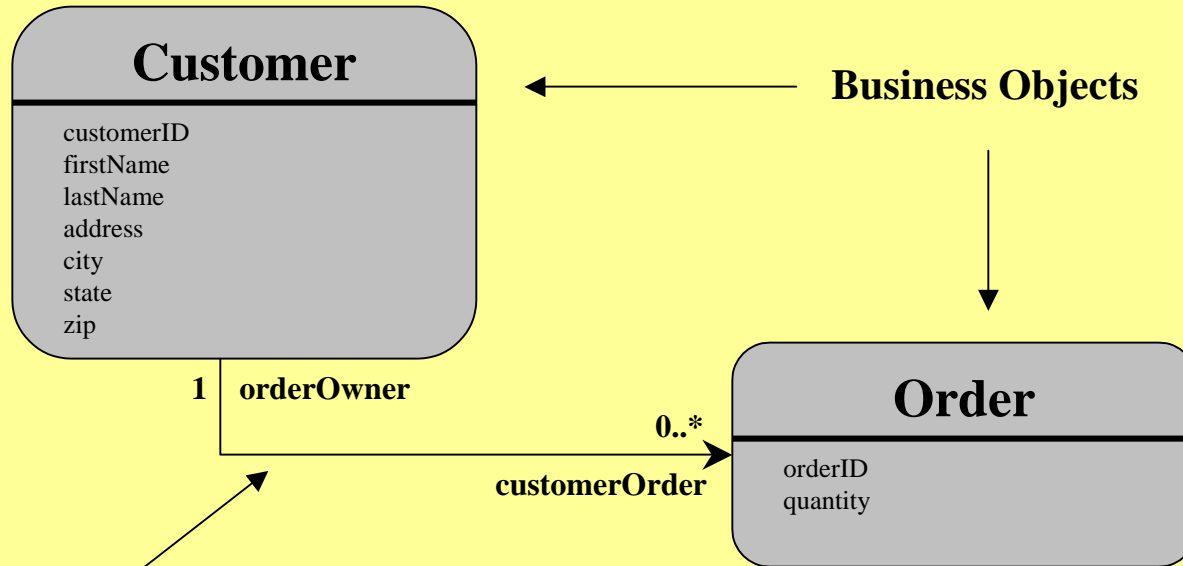
- **Define concrete semantics between state machine and their context**
 - How are state machines started?
 - Can an instance have multiple active state machines?
- **How is an incoming message converted to an event to be delivered to a state machine**
- **How is a generated event converted to an outgoing message**
- **Ambiguity in handling SynchronState**

Challenge in Executable Models: Clarify Variations

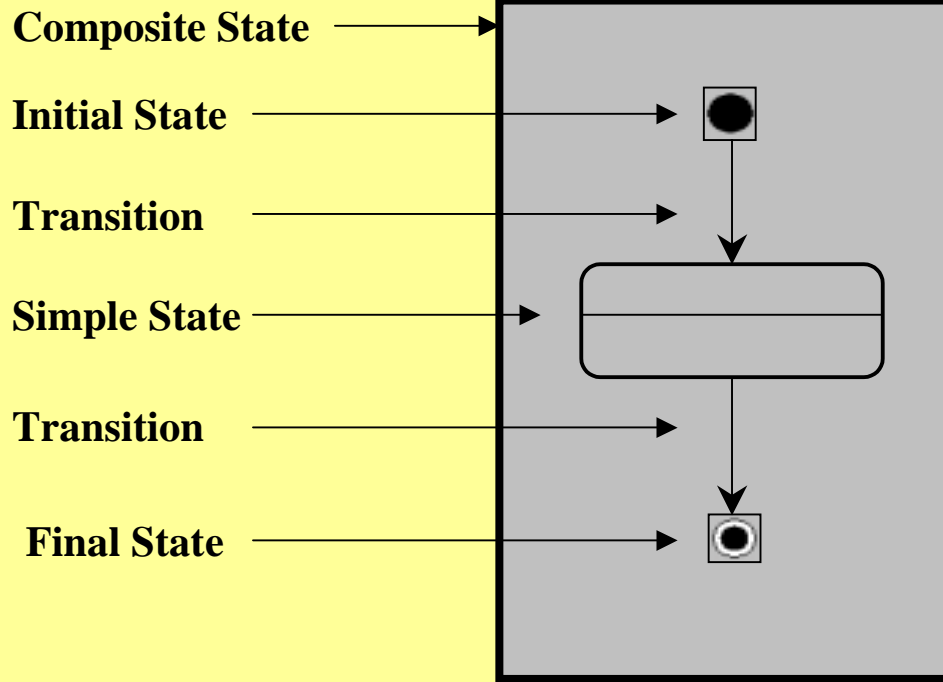
- **Resolving transition selection**
- **Processing transitions in series or concurrently**
- **Transaction boundaries around state machine semantics**
- **Mapping of UML structure models to RDBMS**
- **Selection of action representation (1.x or Action Semantics) and the concrete syntax for actions**
- **Selection of primitive actions, functions, data types**
- **Implementation of custom code actions**



Class Diagram

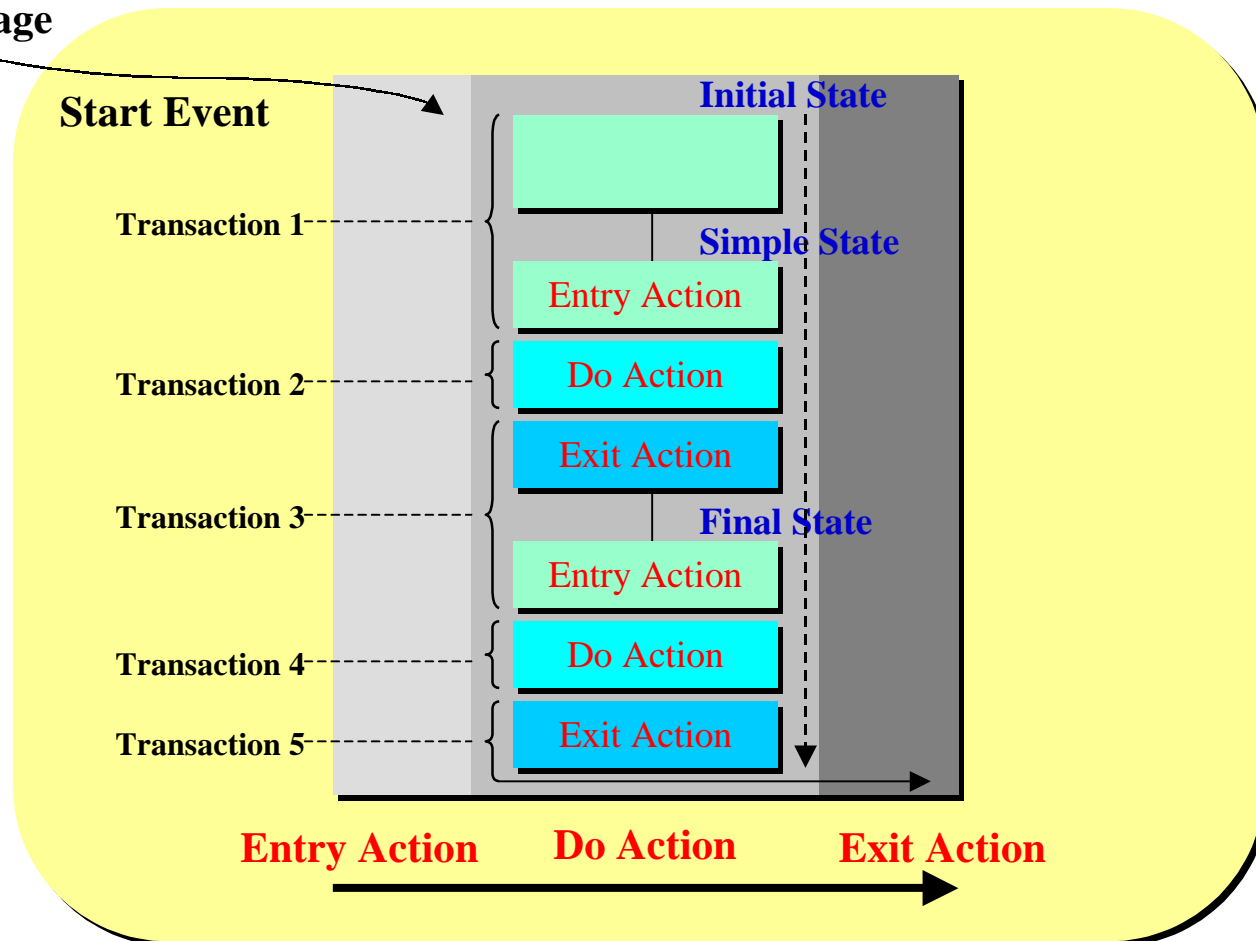


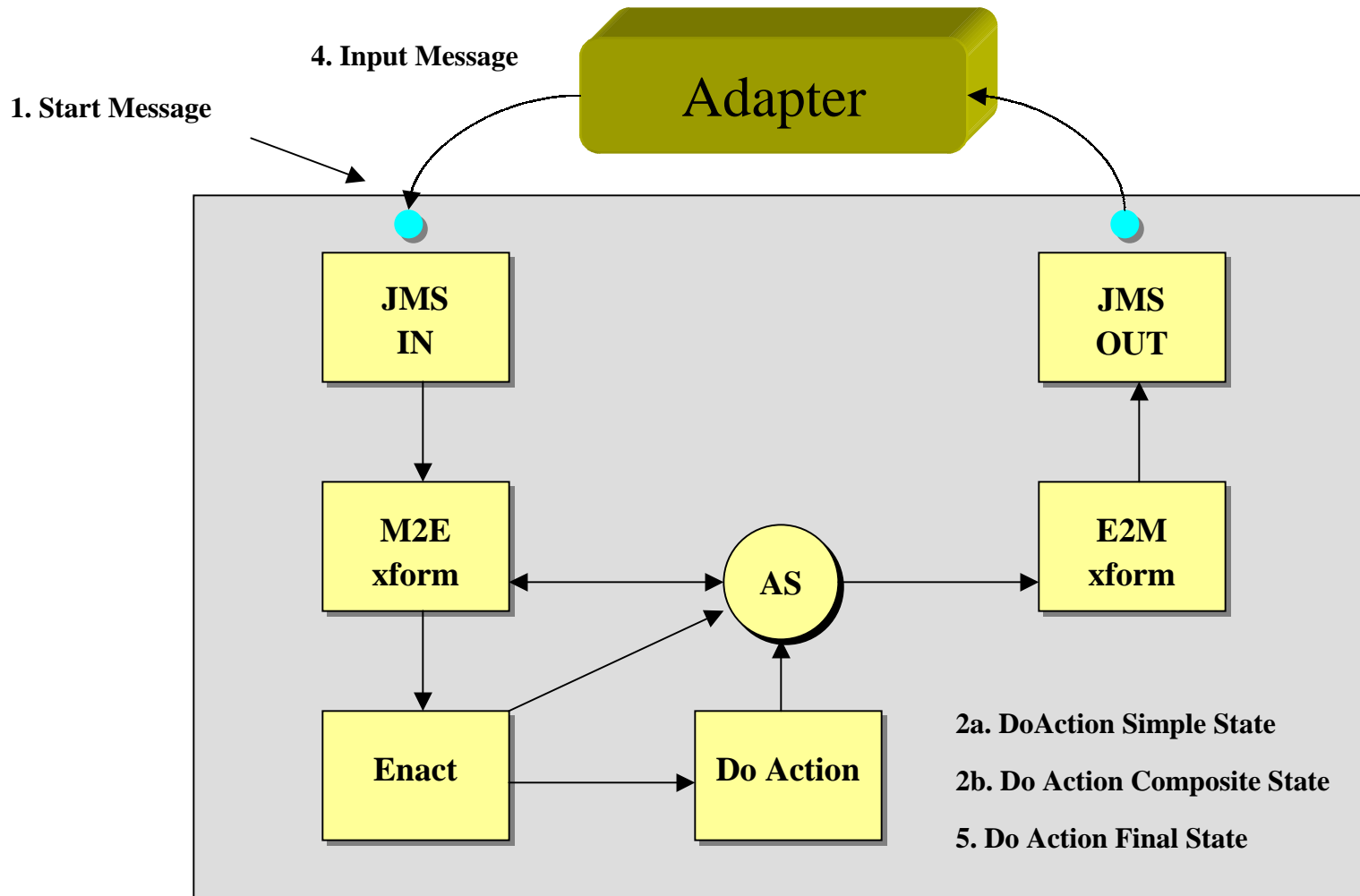
Object



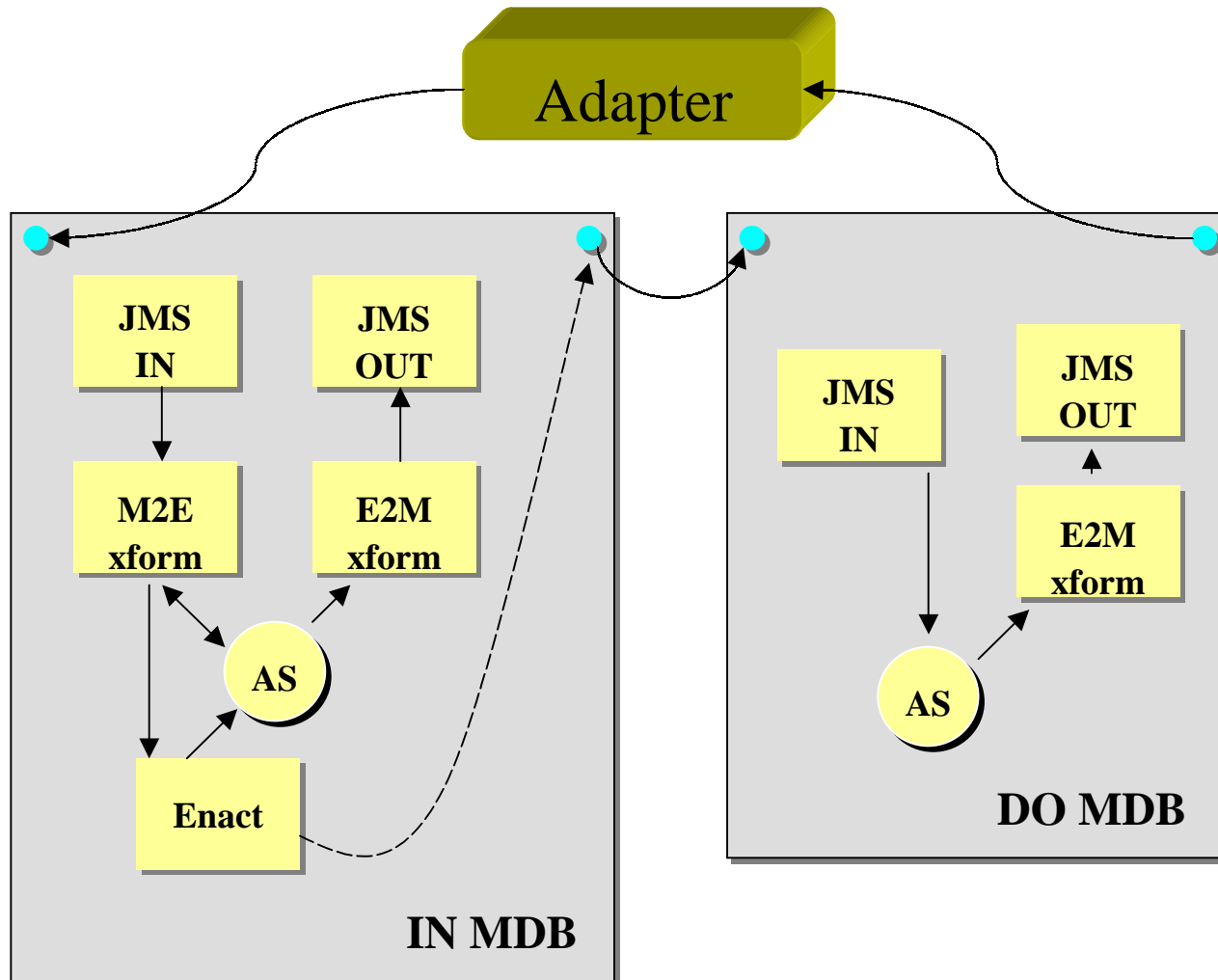
Transaction Model

Start Message

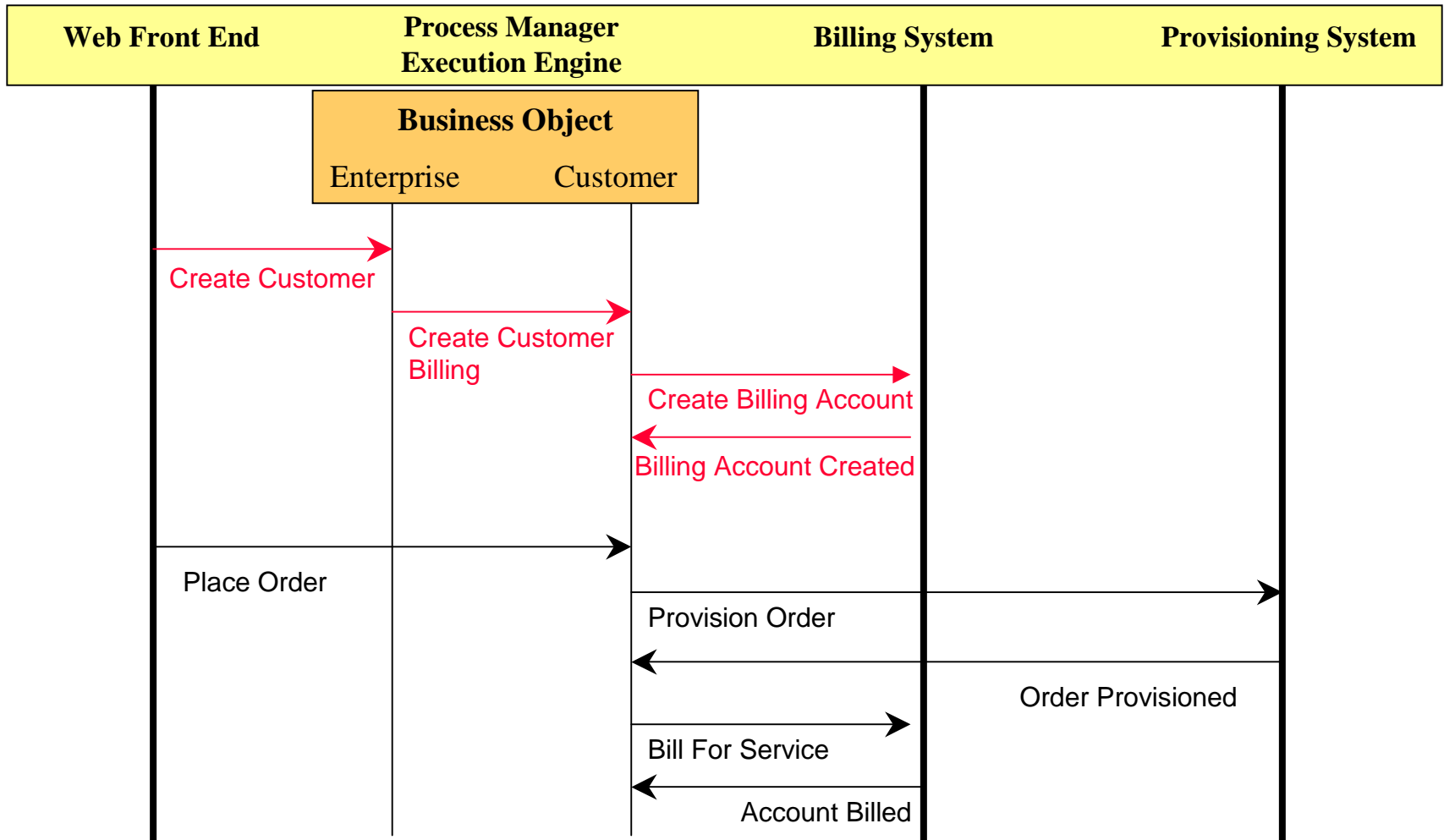


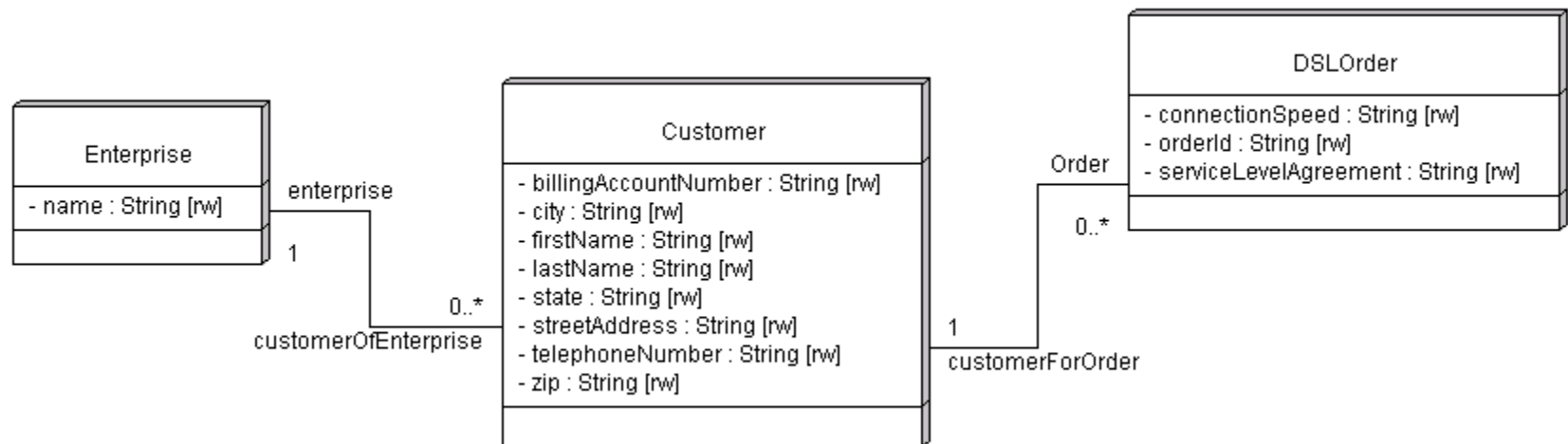


Message Driven Beans Support Transaction Model



Message Traffic Example

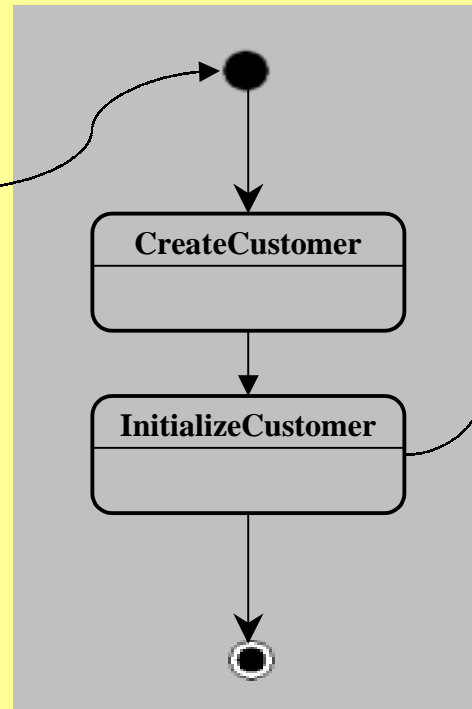




Enterprise

Create Customer

1. StartEvent
CreateCustomer
(from web)



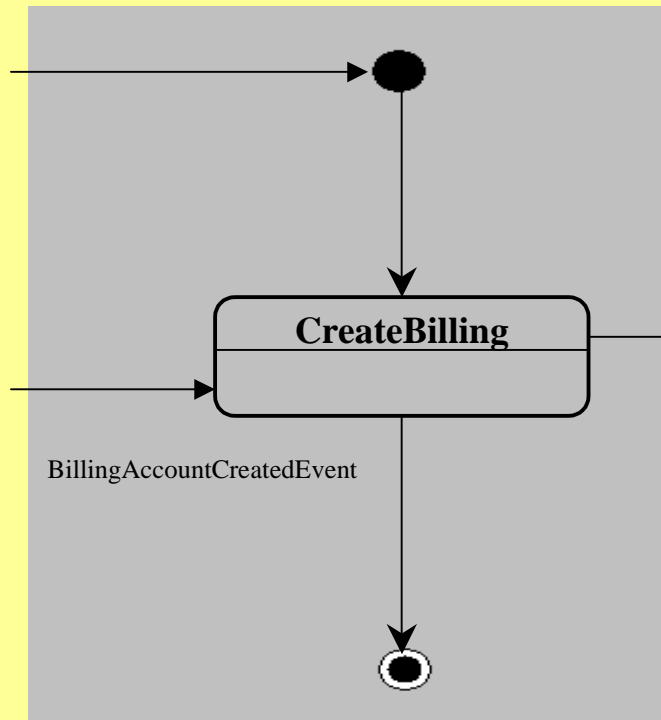
2. Send Message
CreateCustomerBilling
(to Customer)

Customer

Create Billing

Receive StartEvent
CreateCustomerBilling
(from Enterprise)

4. Receive Message
BillingAccountCreated
(from Billing Stub)



3. Send Message
BillingAccountCreation
(to Billing Stub)

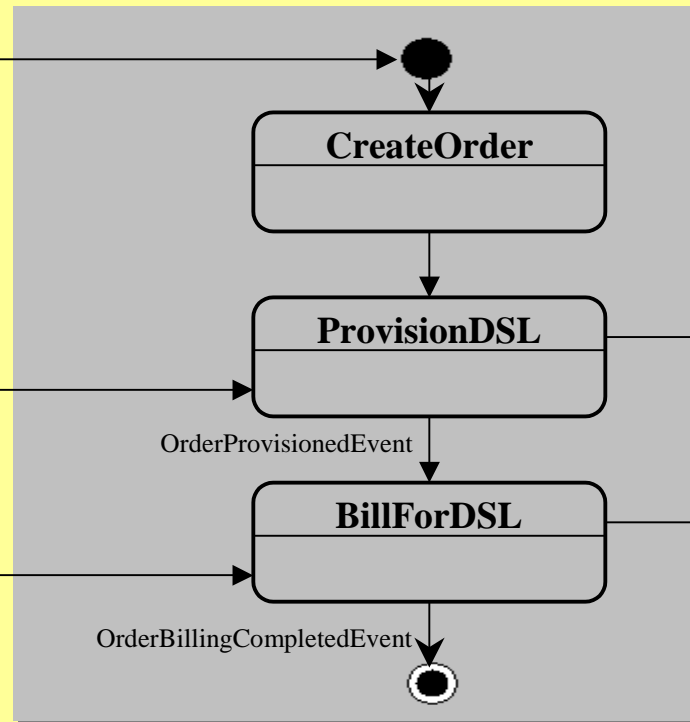
Customer

Place Order

**5. Receive StartEvent
PlaceCustomerOrder**
(from Web)

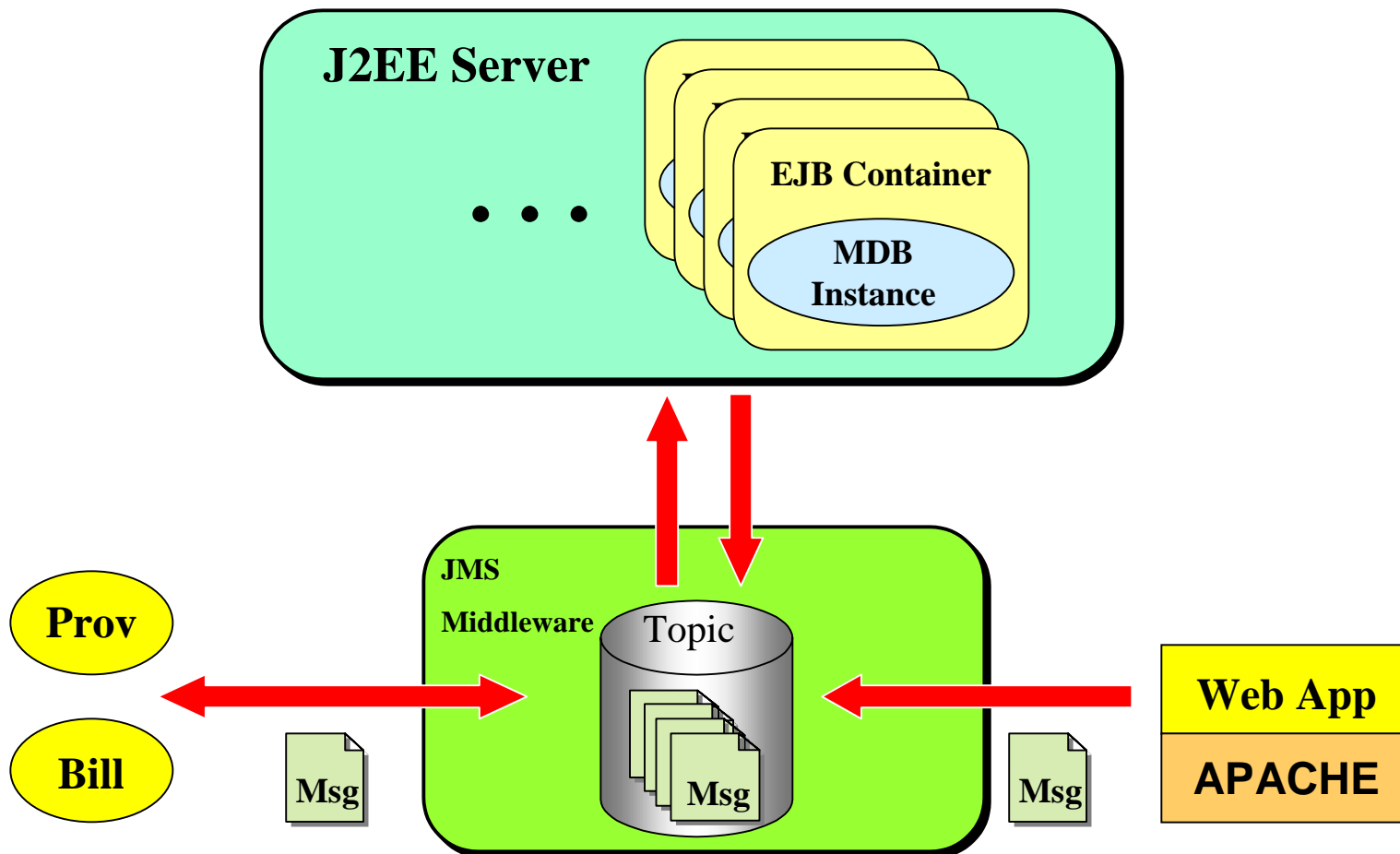
**7. Receive Message
OrderProvisioned**
(from Provision Stub)

**9. Receive Message
OrderBillingCompleted**
(to Billing Stub)



**6. Send Message
ProvisionOrder**
(to Provision Stub)

**8. Send Message
BillForOrder**
(to Billing Stub)



- **Process Manager**
 - **UML Based OO Authoring and CM**
 - **Execution of Smart Distributed Objects**
- **J2EE Application Server**
 - **Scalability**
 - **Transaction control**
- **JMS/JCA**
 - **Distribution, Delivery, Status**

Demonstration is available



Show Me !