CSCI427/927 Systems Development

Service Design, Composition, Interoperation
SoaML

Acknowledgement: the materials in the slides are adapted from SoaML specification

Service Oriented Architecture

- Service Oriented Architecture (SOA) is a way of describing and understanding organizations, communities and systems
  - Maximize agility, scale, and interoperability
- People, organizations, and systems provide services to each other.
  - Be more efficient and agile
  - Enable to offer our capabilities to others in exchange for some value => establishing a community, process, or market place.

Service Oriented Architecture

- A service is value delivered to another through a well-defined interface and available to a community (which may be the general public).
- A service **results in work provided** to one by another.
- SOA is an architectural paradigm for defining how people, organizations, and systems provide and use services to achieve results.
- SOA supports service design, composition and interoperation.

SOA modelling language

- It is important to provide modelling support in which business analysts and application architects can exchange views and share understanding.

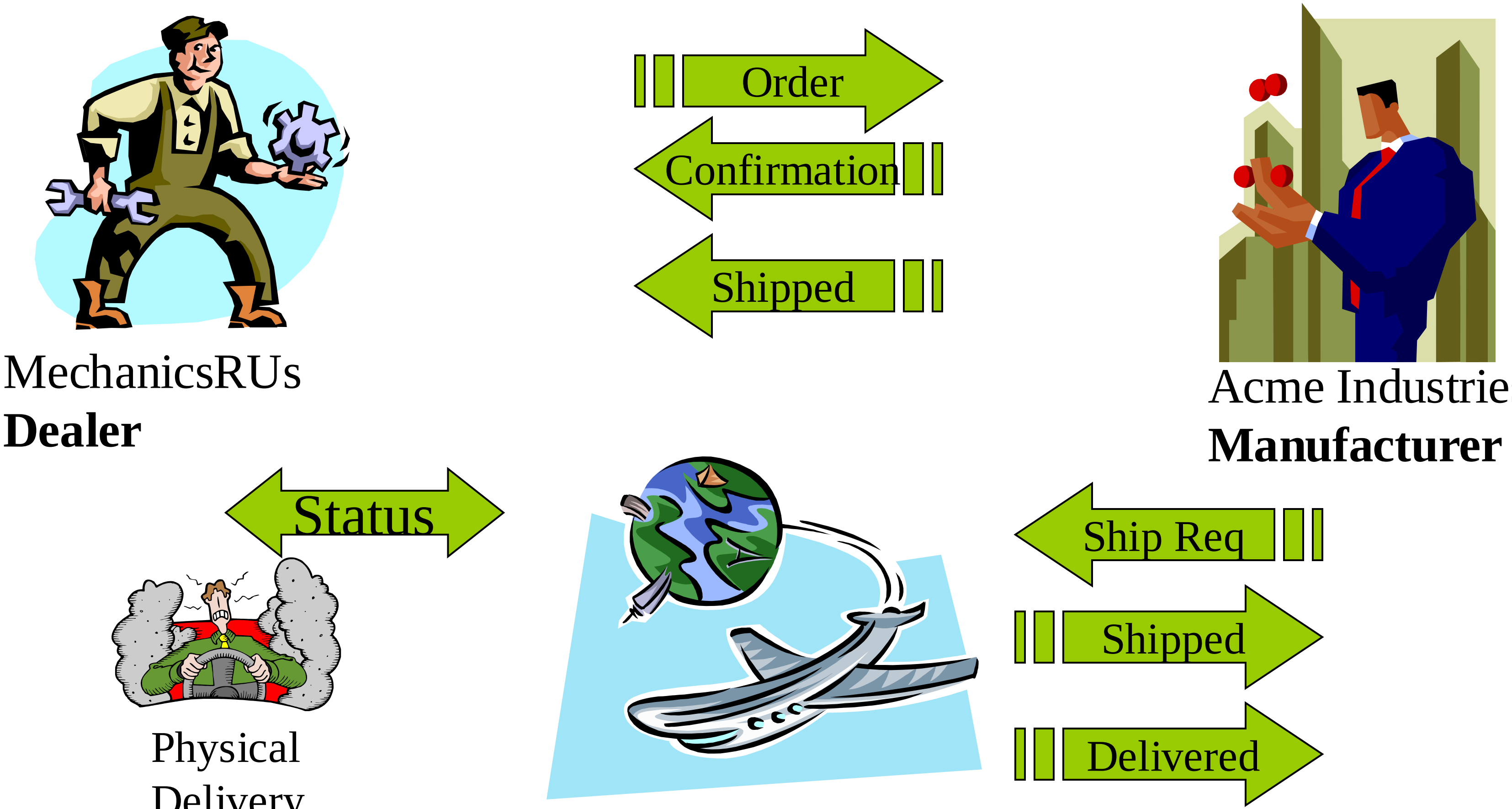
- Several modelling languages for SOA
  - E.g. SOMF, SOMA, SoaML.

- SoaML is a Object Management Group’s (OMG) standard
  - It is a UML profile and metamodel for the modelling and design of services within an SOA

Key concepts

- Service Architecture: a specification of a community
  - Participants – roles
  - Service contracts – collaboration (provide and consume services)
- Service contract: a specification of a service
  - Role – provider and consumer
  - Interfaces
  - Choreography (protocol, behaviour)

SOA Marketplace Example



MechanicsRUs
Dealer

Order

Confirmation

Shipped

Acme Industries
Manufacturer

Status

Physical
Delivery

Ship Req

Shipped

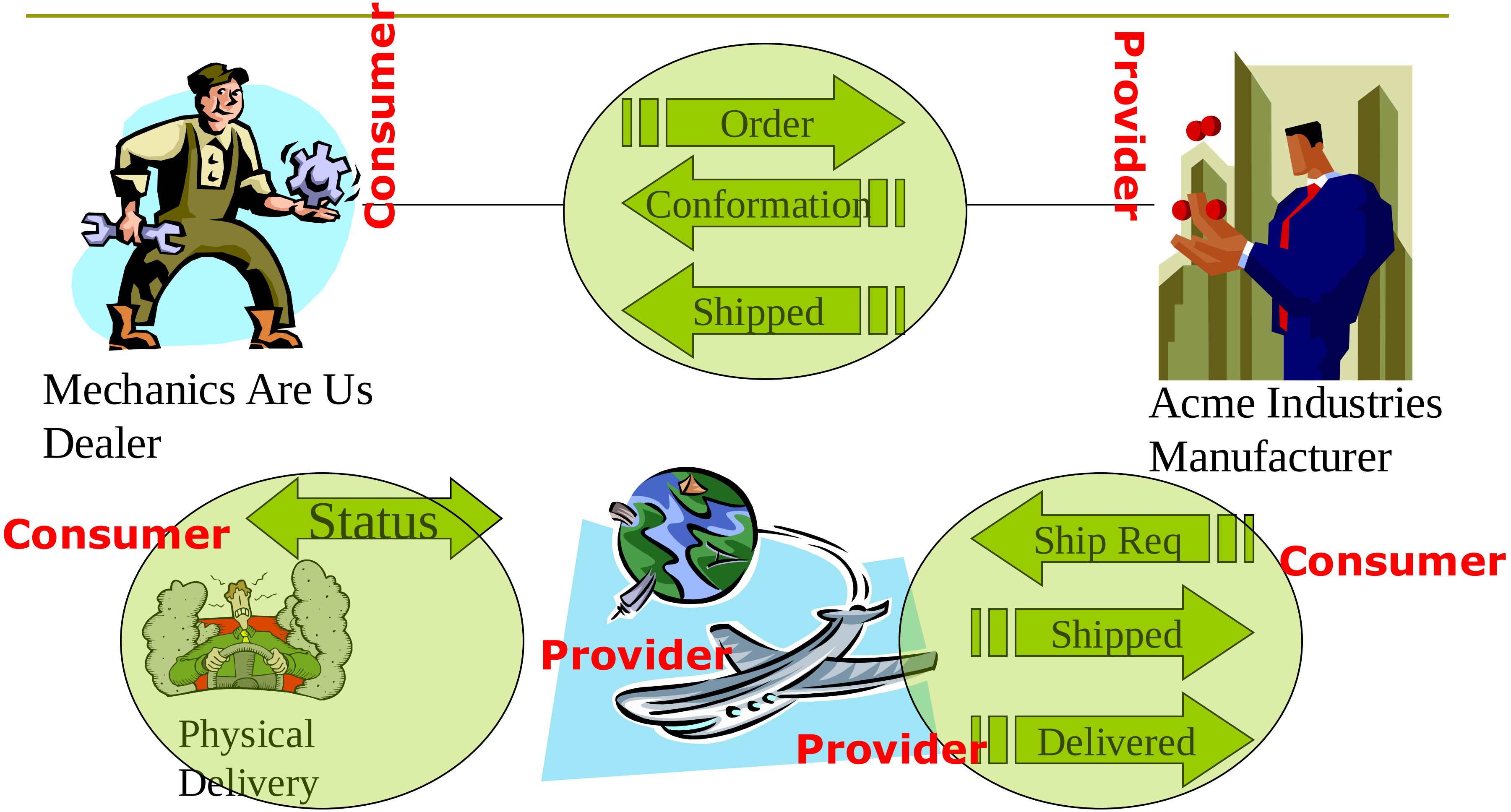
Delivered

GetItThere Freight
Shipper

SOA Marketplace

- The marketplace is a **community of independent** dealers, manufacturers, and shippers.
- Dealers, manufactures, shippers **have their own** business processes, rules and information.
- They want to work together **cohesively**, and they do ***not*** want to redesign their business processes or systems.
- We need to define a service oriented architecture (SOA) for the community:
  - open (allowing new business entities to participate)
  - agile

Marketplace Services



Consumer

Order

Conformation

Shipped

Provider

Mechanics Are Us
Dealer

Acme Industries
Manufacturer

Consumer

Status

Physical
Delivery

Provider

Ship Req

Shipped

Delivered

Consumer

Provider

GetItThere Freight
Shipper

UML Collaboration

A collaboration is shown as a dashed ellipse icon containing the name of the collaboration. The internal structure of a collaboration as comprised by roles and connectors may be shown in a compartment within the dashed ellipse icon. Alternatively, a composite structure diagram can be used.

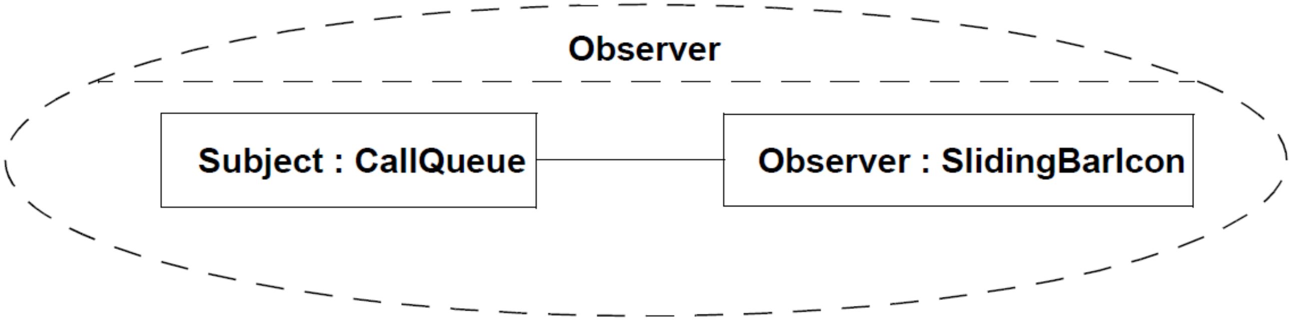
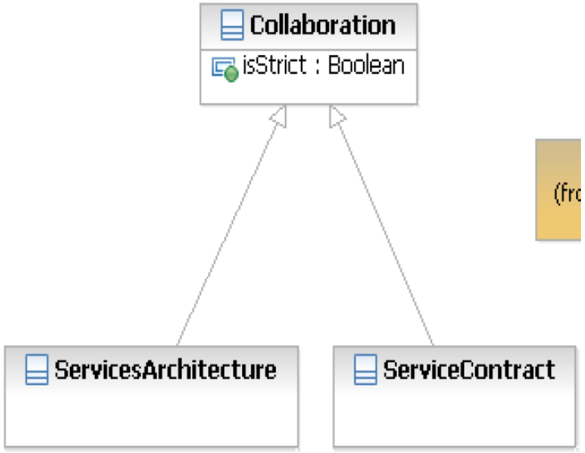
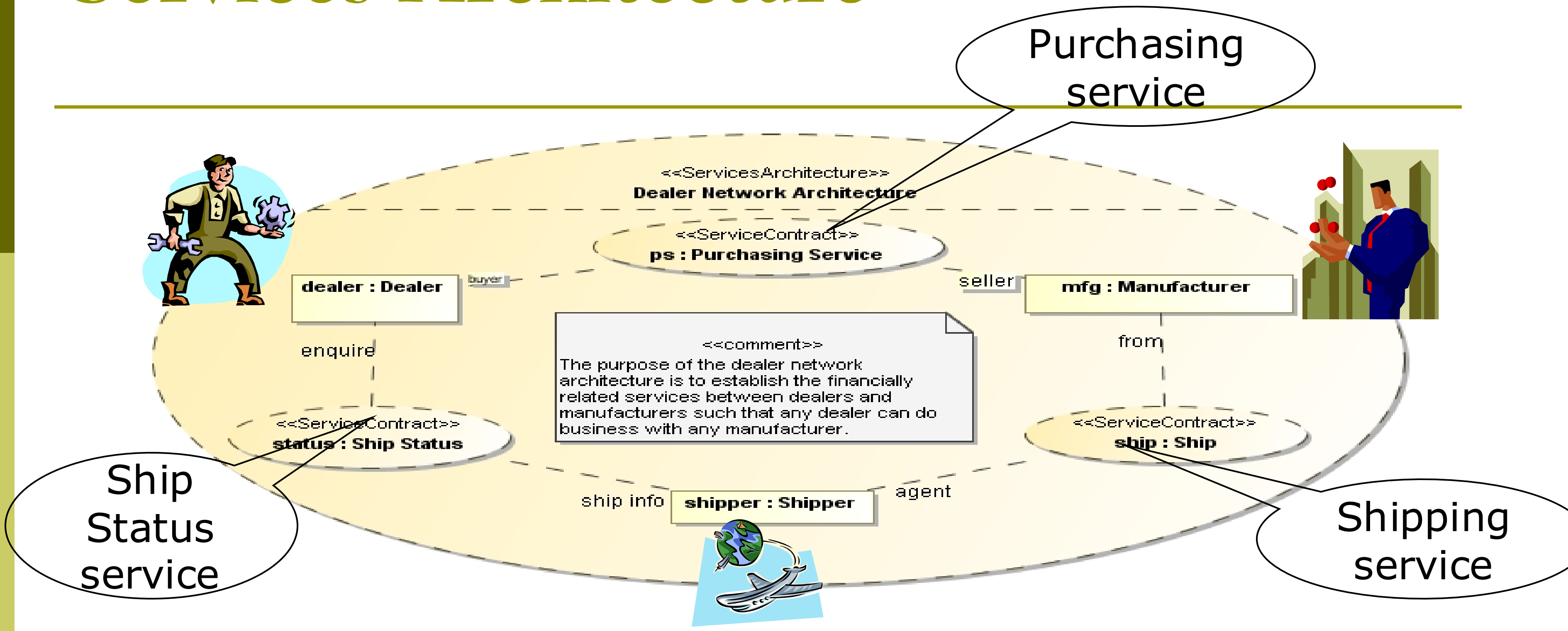


Figure 9.11 - The internal structure of the Observer collaboration shown inside the collaboration icon (a connection is shown between the Subject and the Observer role).



Services Architecture

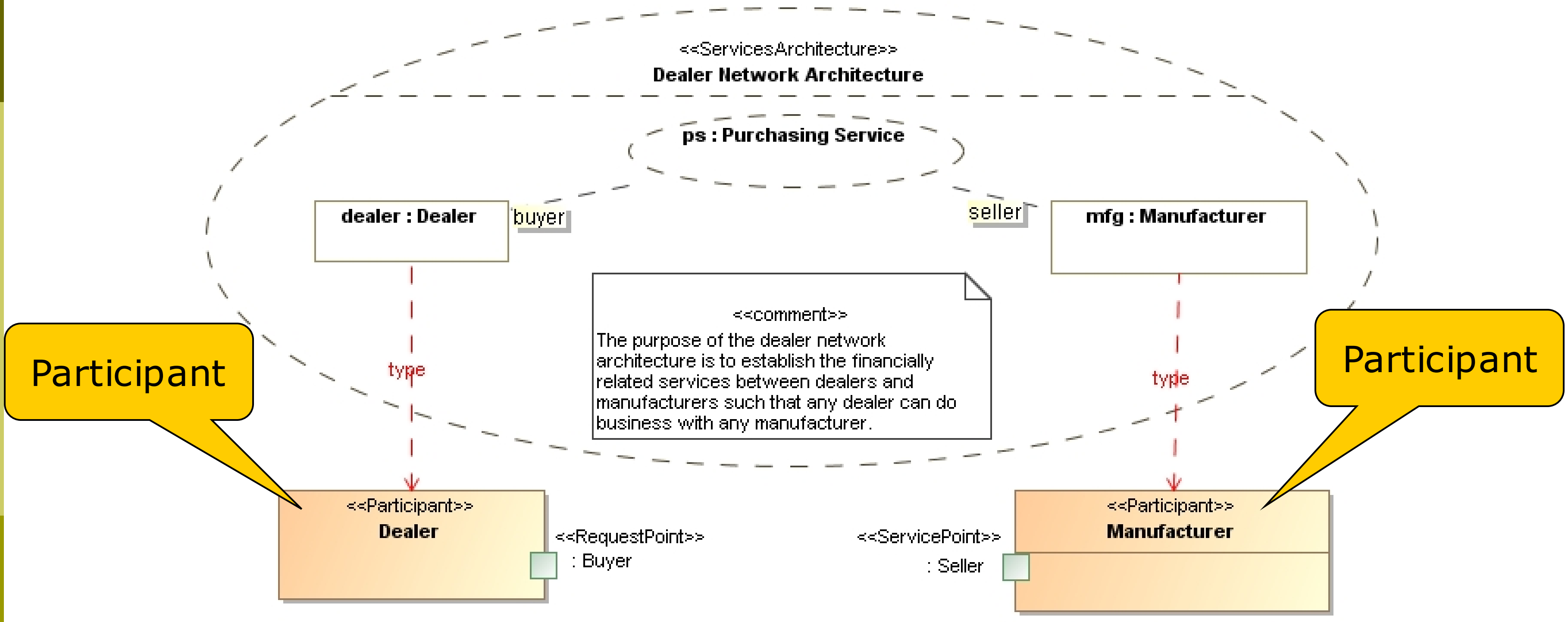


A ServicesArchitecture (or SOA) is a **network of participant** roles *providing* and *consuming services* to fulfill a purpose.

The services architecture puts a set of services in context and shows how participants work together for a community or organization without required process management.

A *community* ServicesArchitecture is defined using a **UML Collaboration**.

Participants



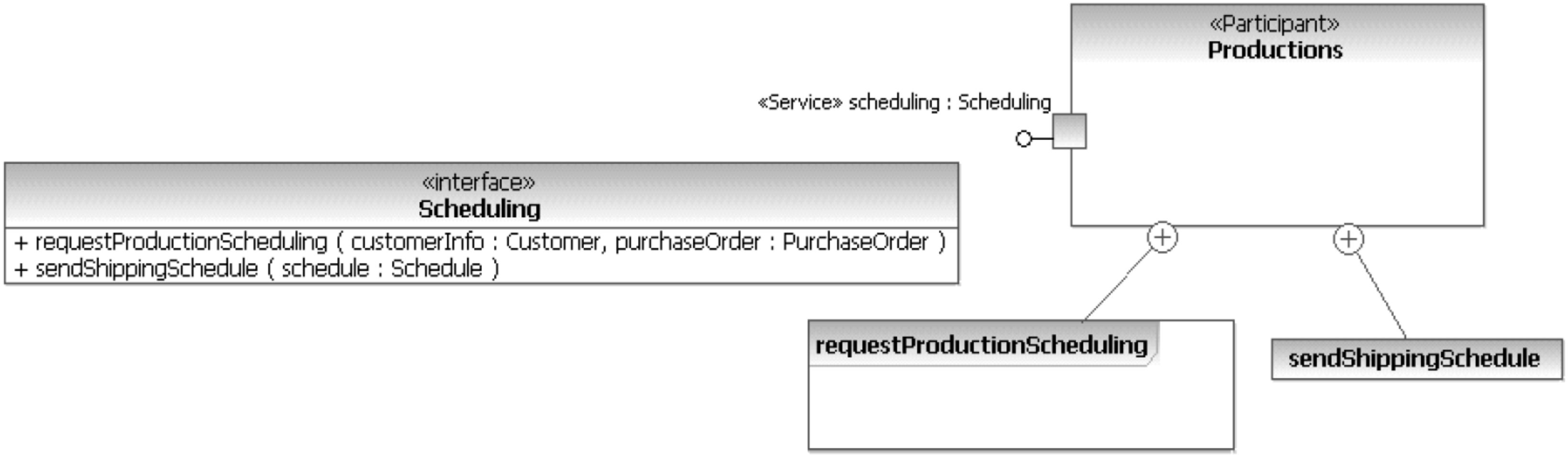
<<comment>>
The purpose of the dealer network architecture is to establish the financially related services between dealers and manufacturers such that any dealer can do business with any manufacturer.

Participants:

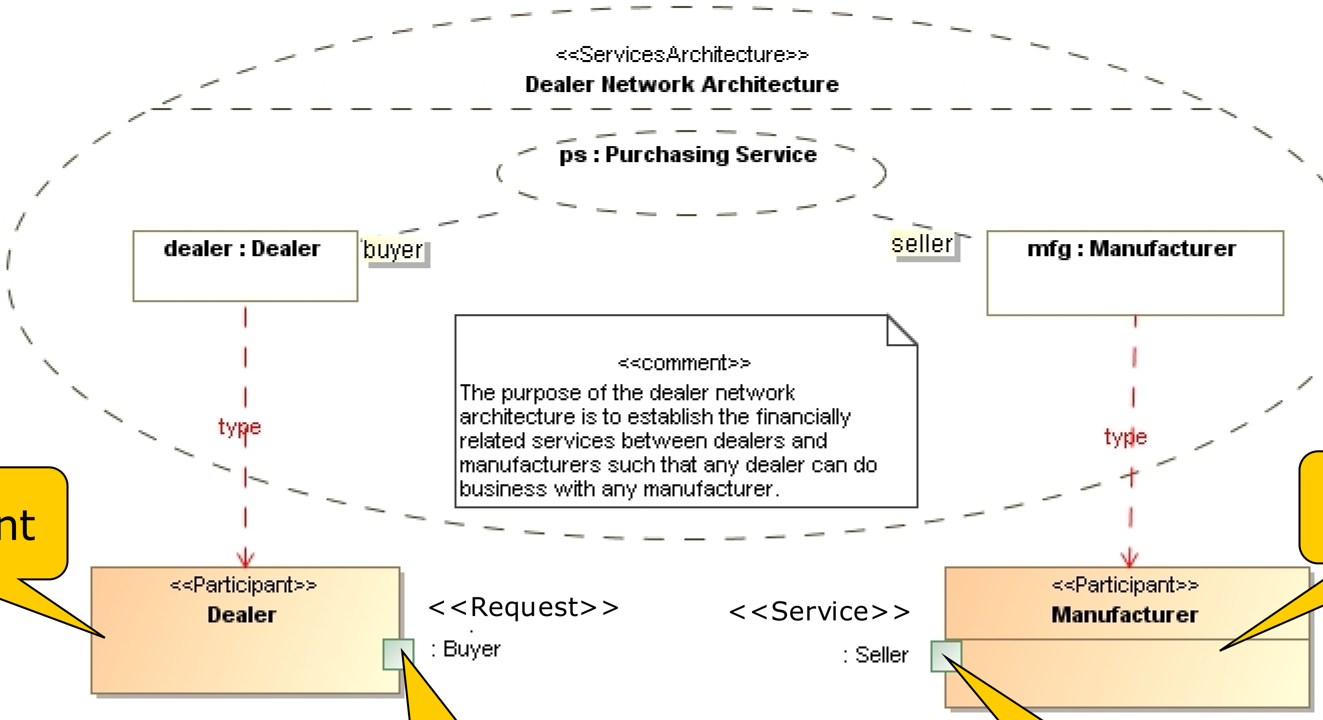
- represent logical or real people or organizational units that participate in services architectures and/or business processes.
- provide and use services, defining their external contract

Participant ports

- Ports - participants **provide or consume services** via ports.
- A port is the part or feature of a participant that is the interaction point for a service - where it is provided or consumed.
- A port where a service is offered may be designated as a **“Service” port**
- A port where a service is consumed may be designated as a **“Request” port**.


«Participant»
Productions
«Service» scheduling : Scheduling
«interface»
Scheduling
+ requestProductionScheduling (customerInfo : Customer, purchaseOrder : PurchaseOrder)
+ sendShippingSchedule (schedule : Schedule)
requestProductionScheduling
sendShippingSchedule

Participants with Service Ports and Request Ports



<<comment>>
The purpose of the dealer network architecture is to establish the financially related services between dealers and manufacturers such that any dealer can do business with any manufacturer.

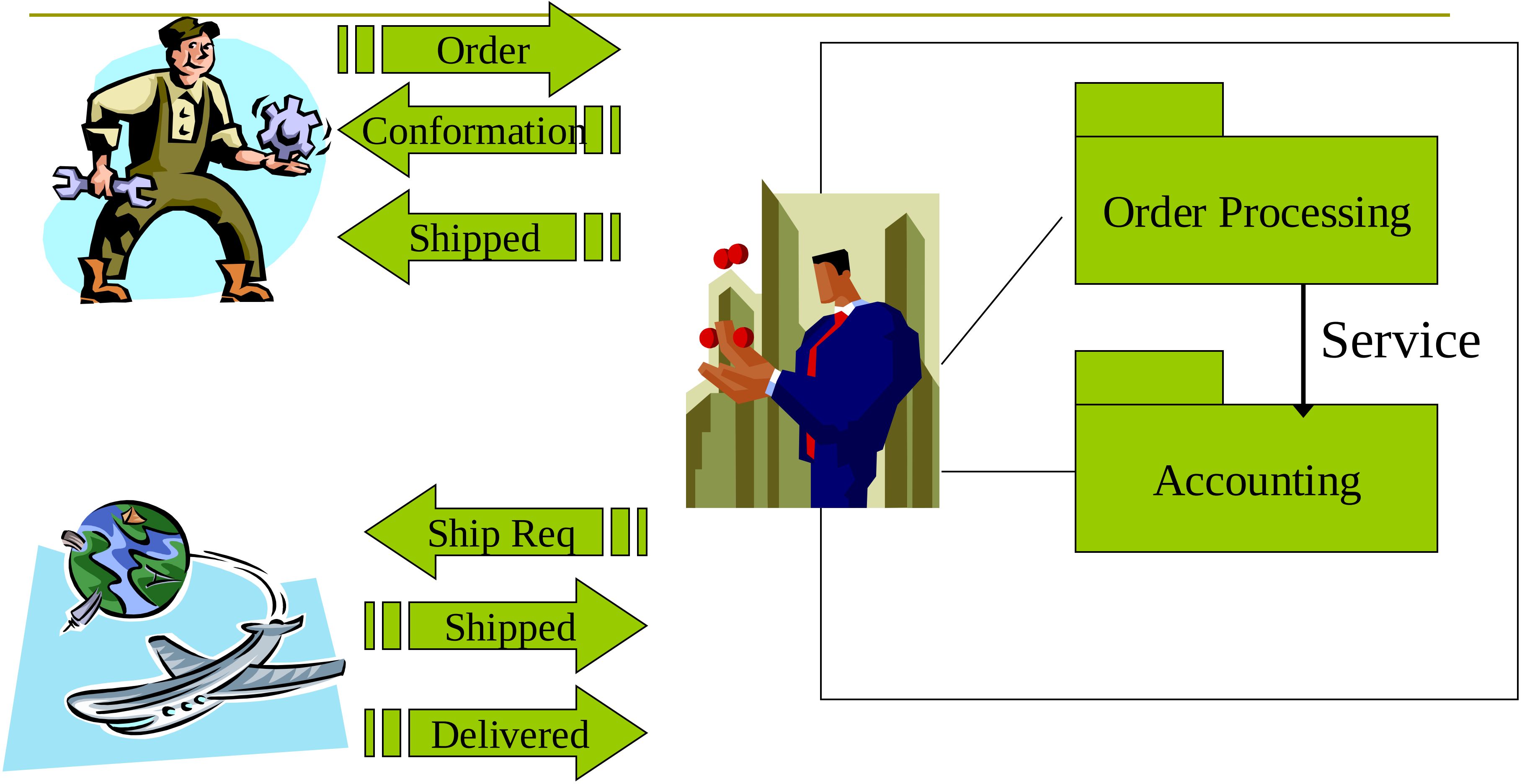
Participant

Participant

Request Port – port where the dealer uses the purchasing service

Service Port – port where the manufacturer offers the purchasing service

Inside the Manufacturer particpant



Order

Conformation

Shipped

Ship Req

Shipped

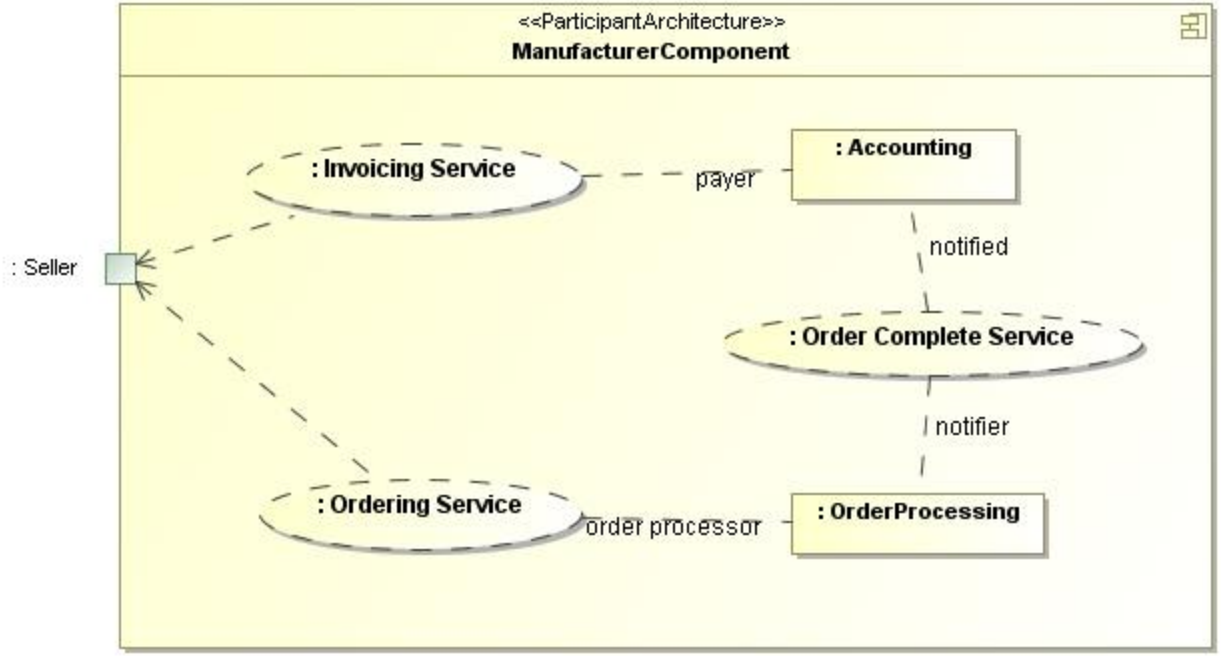
Delivered

Order Processing

Service

Accounting

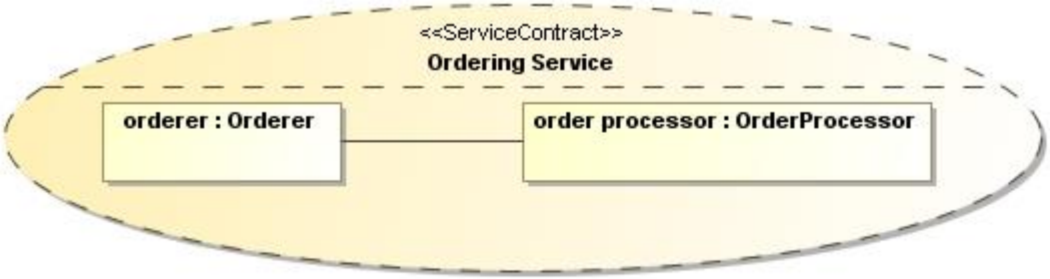
Services architecture for a participant



ParticipantArchitecture is the high-level services architecture of a participant that defines how a set of **internal** and **external** participants use services to implement the responsibilities of the participant.

A participant will also frequently have **a business process**.

ServiceContract

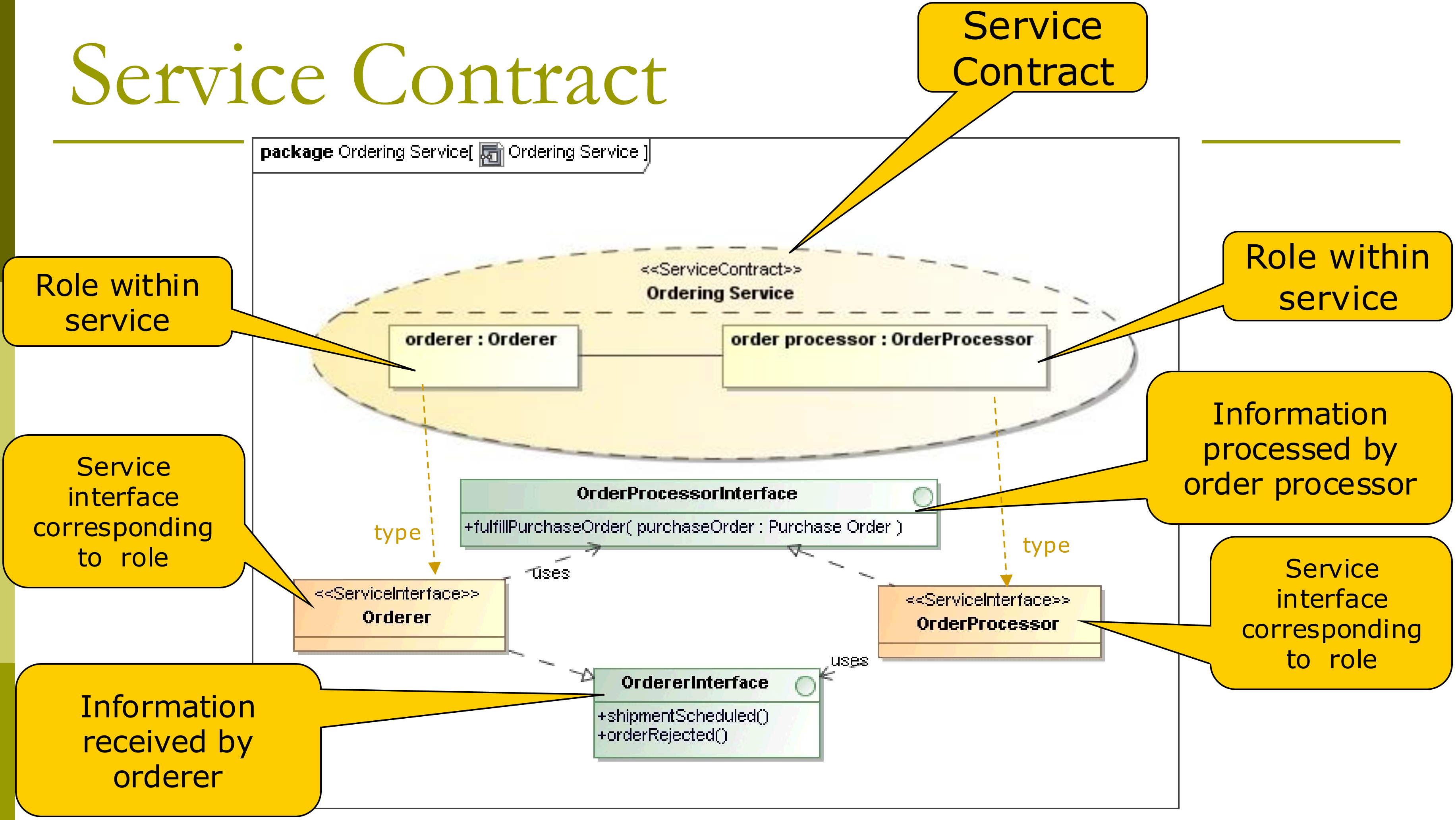


A ServiceContract

- Fully specifies the service (terms, conditions, interfaces, choreography)
- Is **binding** on both the providers and consumers of that service
- Is defined using a UML collaboration that is focused on the interactions involved in providing a service.

A participant plays a role in the larger scope of a ServicesArchitecture and also plays a role as the provider or user of services specified by ServiceContracts.

Service Contract

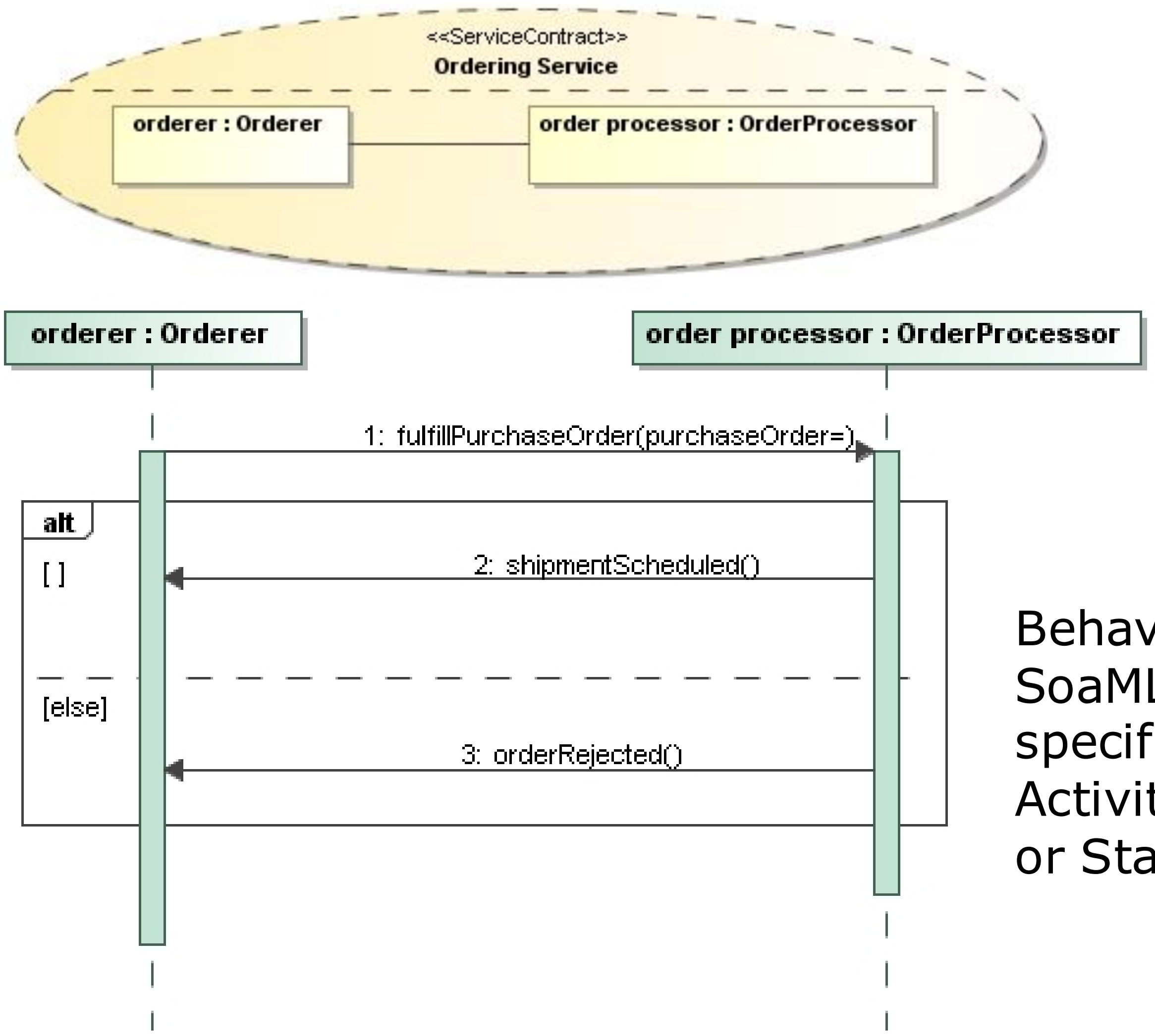

Service Contract
package Ordering Service[Ordering Service]
<<ServiceContract>>
Ordering Service
Role within service
Role within service
orderer : Orderer
order processor : OrderProcessor
Information processed by order processor
OrderProcessorInterface
+fulfillPurchaseOrder(purchaseOrder : Purchase Order)
type
type
uses
Service interface corresponding to role
<<ServiceInterface>>
Orderer
<<ServiceInterface>>
OrderProcessor
Service interface corresponding to role
uses
Information received by orderer
OrdererInterface
+shipmentScheduled()
+orderRejected()

The service contract specifies the details of the service – what information, assets and responsibilities are exchanged and under what rules

Behaviour of Service Contract

- Further details needed to describe the flow of information between the participants
  - The “behaviour” associated with a service contract
- The behaviour shows how the participants work together within the context of this service (not their internal process).
  - E.g. How messages “choreographed” in the service contract – what flows between who, when and why.

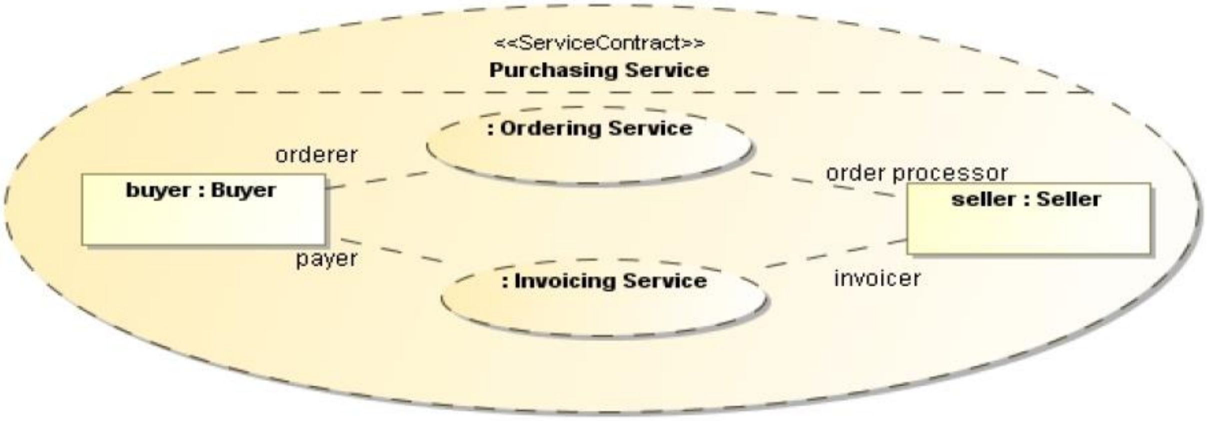
Simple Protocol Choreography for Ordering Service Contract



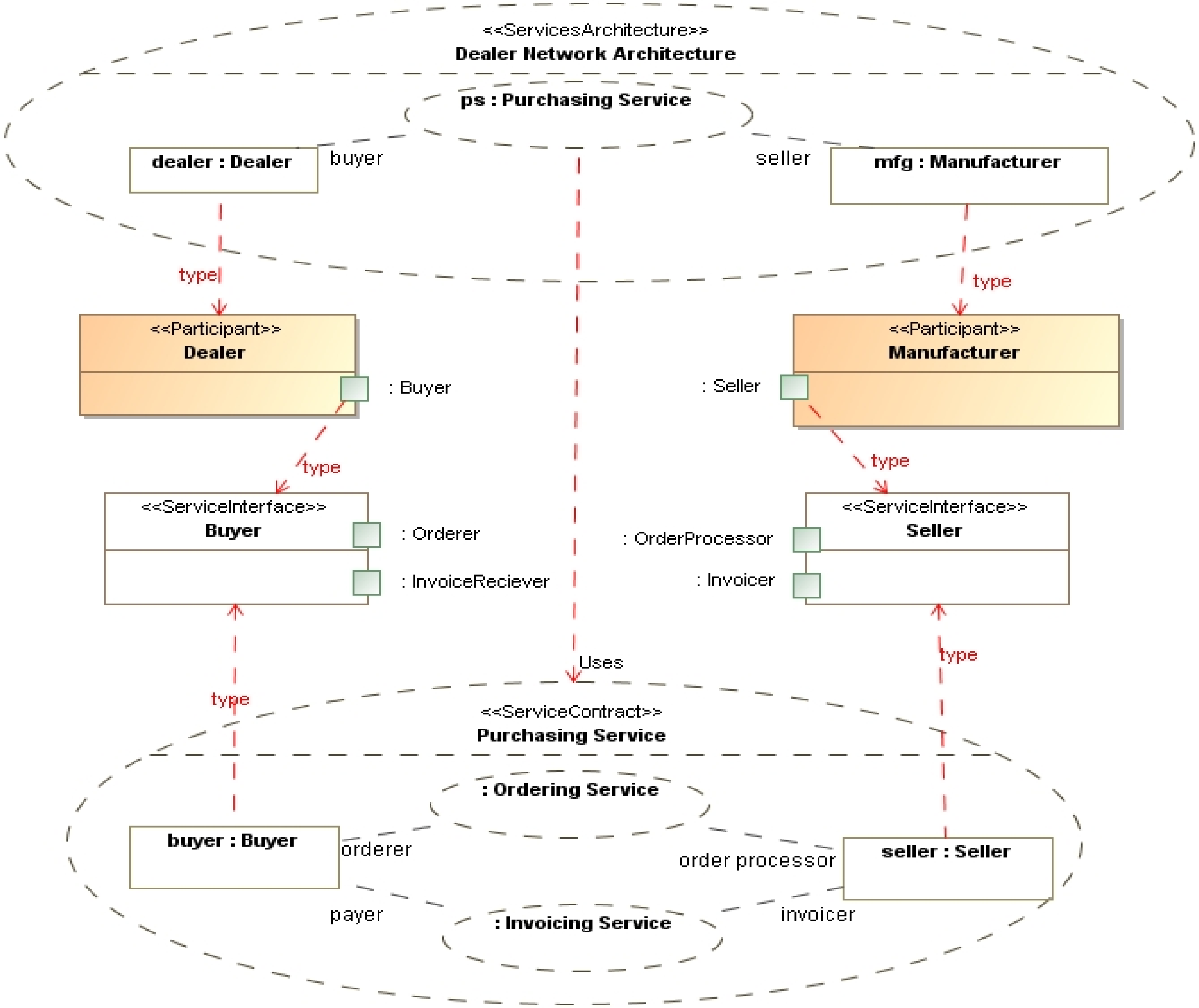
Behaviour in SoaML can also be specified with Activity Diagrams or State Machines

Service Composition

- The purchasing service is actually composed of two simpler services: Ordering service and Invoicing Service.



Relating services architecture to service contracts



Tool support for SoaML

- Some existing UML tools support SoaML
  - E.g. IBM Rational Software Architect, Rational Software Modeler, Sparx Systems' Enterprise Architect, MagicDraw)
  - SoaML specification https://www.omg.org/spec/SoaML/