



TC184/SC5 Ad Hoc meeting  
Oct. 30, 2003, Washington DC

# PSLX-04

## XML Specifications for PSLX

Yasuyuki Nishioka, Prof. Dr.

Hosei University,

PSLX Consortium

nishioka@k.hosei.ac.jp





# Outline

- Part 1: Define object model for XML specifications using core components, and describe XML Schema for PSLX
- Part 2: Define functional specifications of interfaces, and messages to be exchanged through the interface
- Part 3: General messaging protocols and message binding alternatives to middleware that takes part in such services

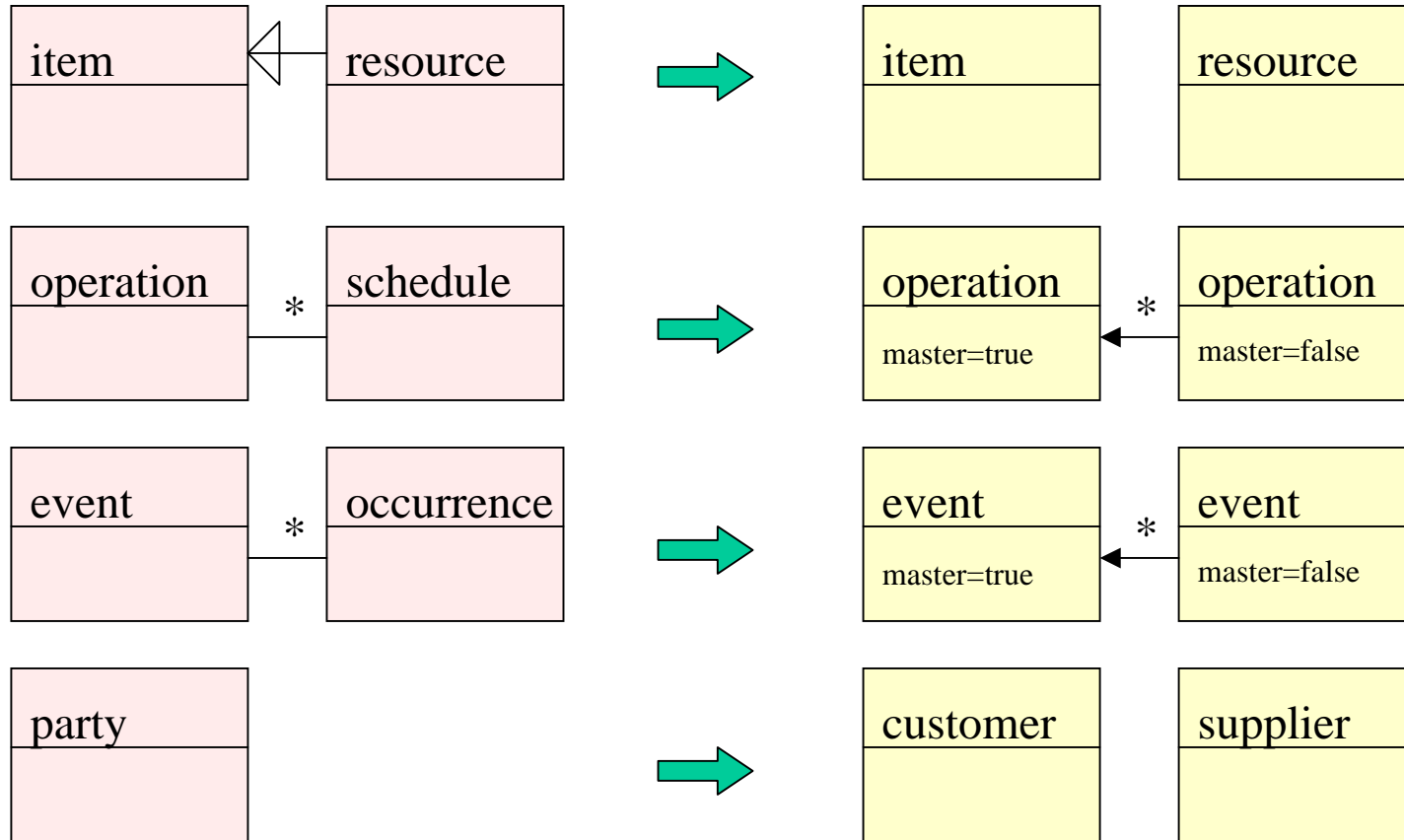


# Design concept

- The number of tag is as less as possible
- Tag name consists of one term
- Common attribute is independent as a tag
- Tag location in a structure is significant
- Single and multiple occurrence are cared
- Using pointer data in an application level
- No cyclic definition in the schema level
- An object corresponds to a tag element



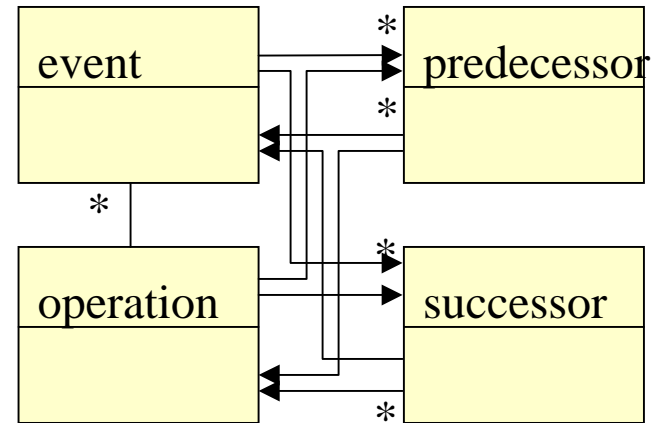
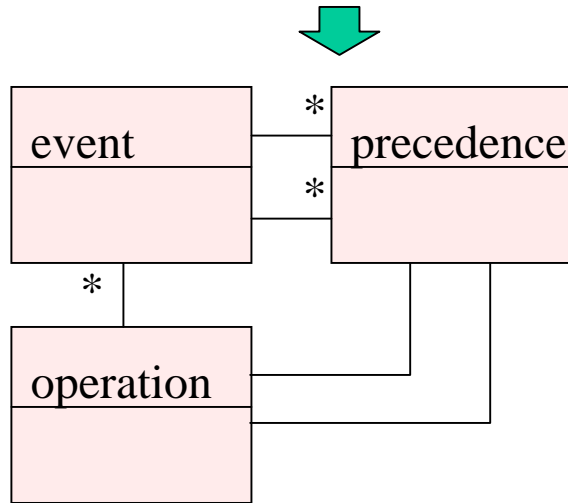
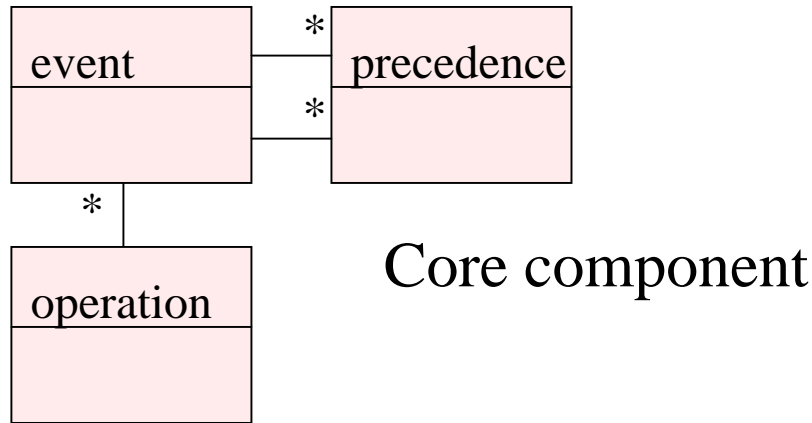
# XML schema object (1)



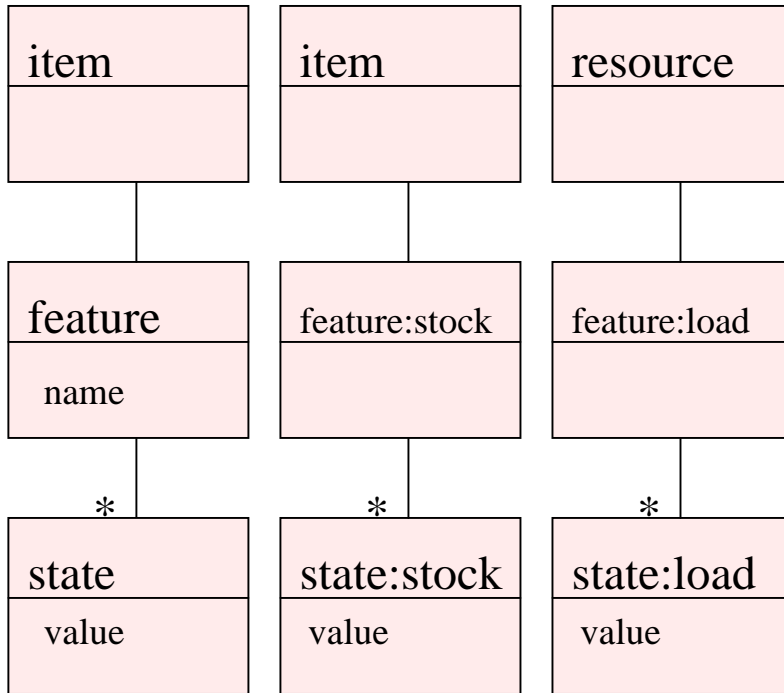
Core component

XML schema object

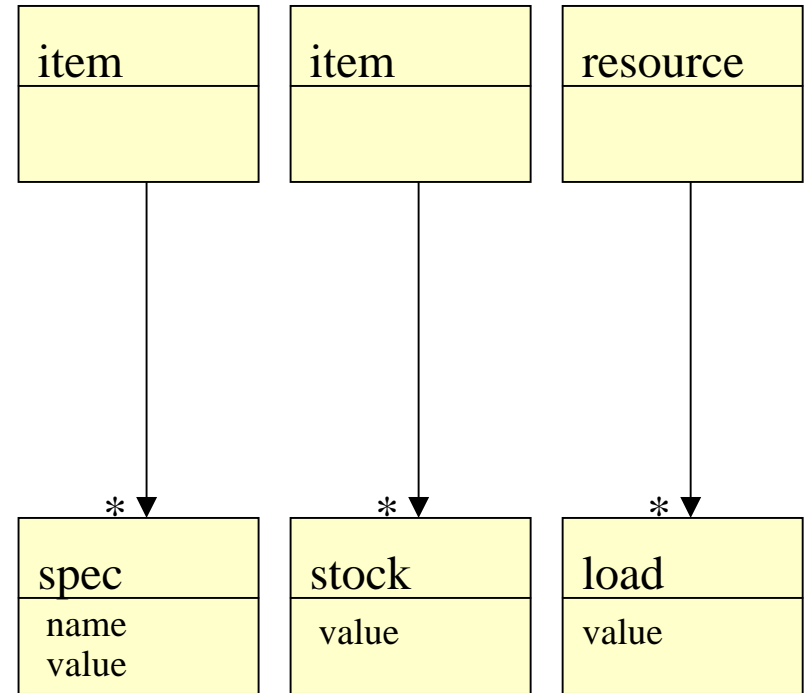
# XML schema object (2)



# XML schema object (3)

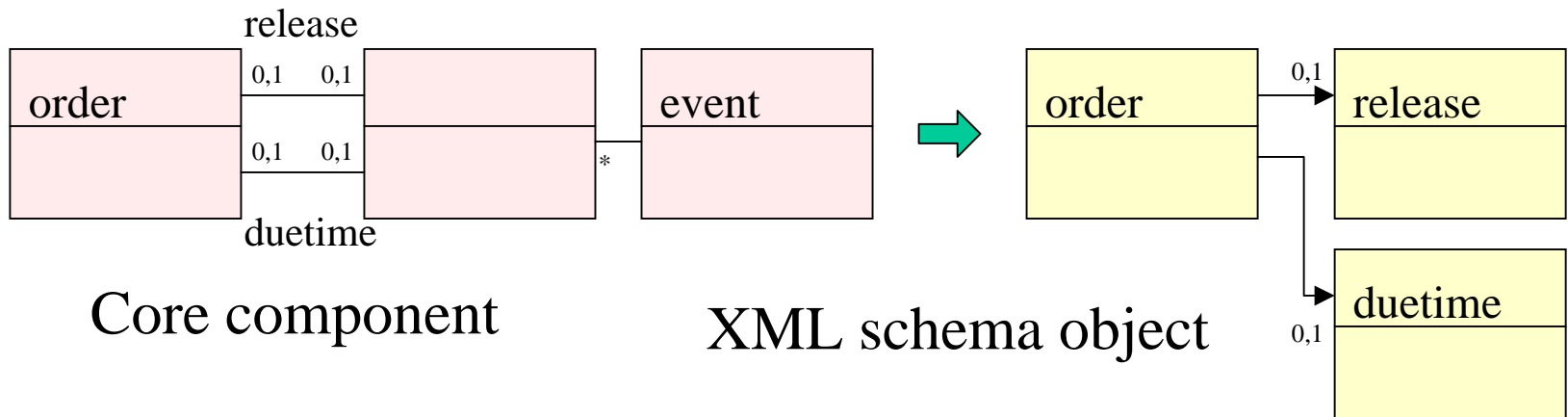
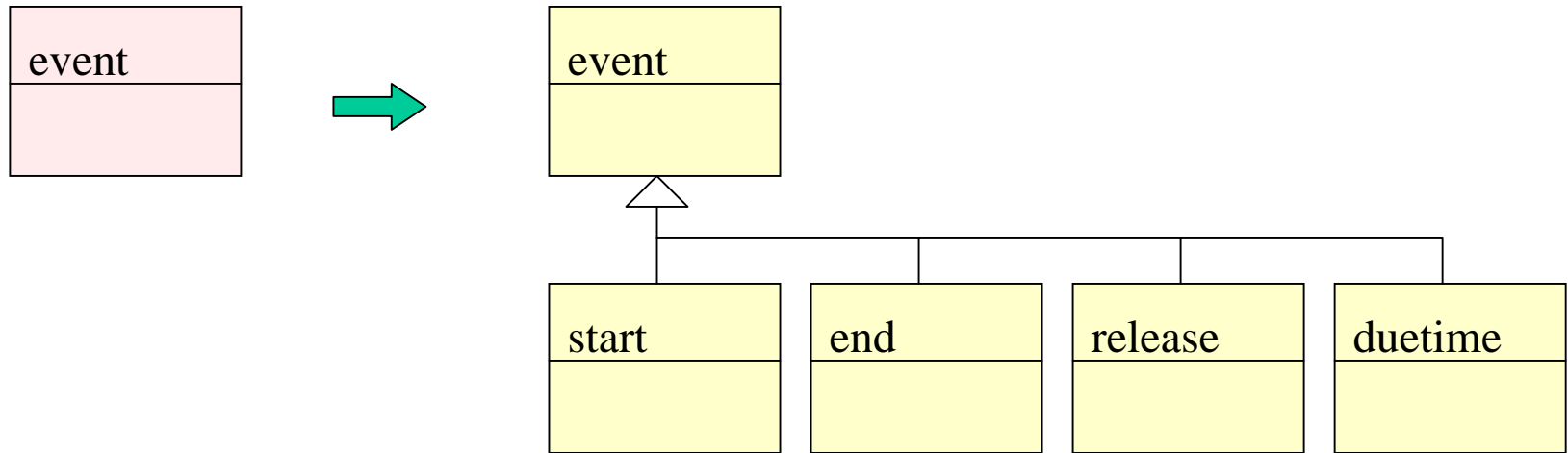


Core component



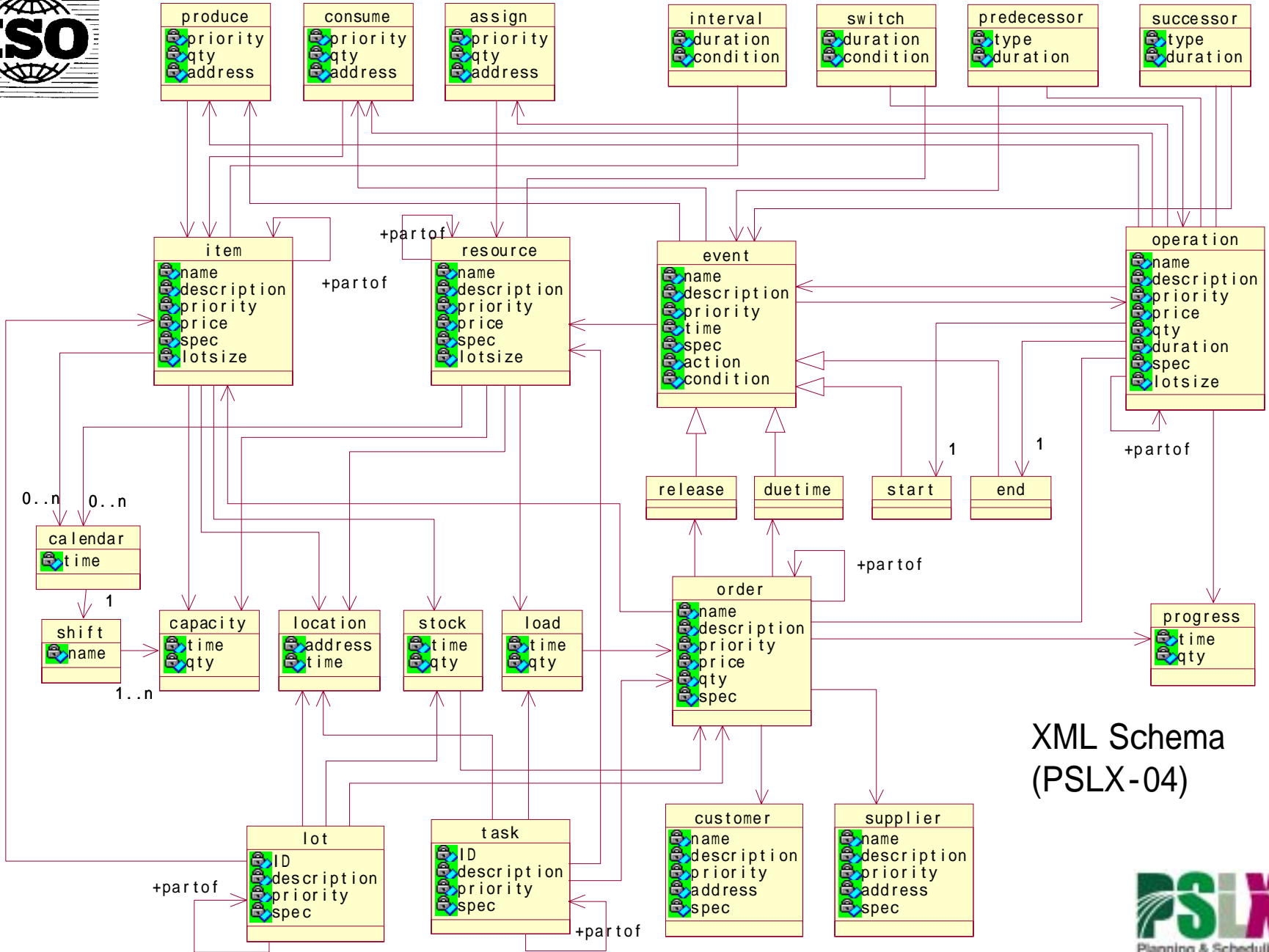
XML schema object

# XML schema object (4)



Core component

XML schema object



## XML Schema (PSLX-04)



# Sample XML specification



```
<?xml version="1.0" encoding="UTF-8" ?>
- <pslx type="response" id="4" ref="4" action="getStock" xmlns="http://www.pslx.org/XMLSchema.xsd">
  <profile current="2003-03-28T23:58:49.000+09:00" />
  - <item name="3U134 65431">
    <display name="CONNECOTOR O-RING" />
    - <stock type="disc">
      <time value="2003-03-28T23:58:49.000+09:00" />
      <qty value="810.0" />
      <rate value="1.0" />
    </stock>
  </item>
  - <item name="3W621 65422">
    <display name="CONNECTOR O-RING" />
    - <stock type="disc">
      <time value="2003-03-28T23:58:49.000+09:00" />
      <qty value="150.0" />
      <rate value="1.0" />
    </stock>
  </item>
  - <item name="3W621 65432">
    <display name="CONNECTOR" />
    - <stock type="disc">
      <time value="2003-03-28T23:58:50.000+09:00" />
      <qty value="300.0" />
      <rate value="1.0" />
    </stock>
  </item>
  - <item name="45570 69080">
    <display name="0A460/62702 65510用" />
    - <stock type="disc">
      <time value="2003-03-28T23:58:50.000+09:00" />
      <qty value="442.0" />
      <rate value="1.0" />
    </stock>
  </item>
  - <item name="48640 69080">
    <display name="UNION COLLAR" />
    - <stock type="disc">
      <time value="2003-03-28T23:58:50.000+09:00" />
    </stock>
  </item>
  - <order name="GP移植品" />
  - <stock type="disc">
    <time value="2003-03-28T23:58:50.000+09:00" />
    <qty value="192.0" />
    <rate value="1.0" />
  </stock>
  - <item name="A9020 69170">
    <display name="UNION O RING" />
    - <stock type="disc">
      <time value="2003-03-28T23:58:50.000+09:00" />
      <qty value="1700.0" />
      <rate value="1.0" />
    </stock>
  </item>
  - <item name="a9020 69180 JIK">
    <display name="UNION O RING" />
    - <stock type="disc">
      <time value="2003-03-28T23:58:50.000+09:00" />
      <qty value="1000.0" />
      <rate value="1.0" />
    </stock>
  </item>
  - <order name="3U134 65431" item="3U134 65431">
    <qty value="648.0" />
  </order>
  - <order name="48640 69080" item="48640 69080">
    <qty value="150.0" />
  </order>
  - <order name="62110 69080" item="62110 69080">
    <qty value="150.0" />
  </order>
  - <order name="A9020 69170" item="A9020 69170">
    <qty value="1000.0" />
  </order>
  - <order name="a9020 69180 JIK" item="a9020 69180 JIK">
    <qty value="12000.0" />
  </order>
</pslx>
```



# PSLX tag set

<pslx>	<char>	<produce>	<event>	< >
<profile>	<address>	<consume>	<ev>	< >
<error>	<description>	<assign>	<start>	< >
<color>	<time>	<predecessor>	<end>	< >
<display>	<duration>	<successor>	<release>	< >
<unit>	<spec>	<partof>	<duetime>	< >
<translate>	<location>	<pegging>	<customer>	< >
<scale>	<progress>	<tracking>	<supplier>	< >
<qty>	<capacity>	<lotsize>	<item>	< >
<price>	<load>	<tasksize>	<resource>	< >
<base>	<stock>	<condition>	<lot>	< >
<priority>	<shift>	<action>	<task>	< >
	<calendar>	<changeover>	<operation>	< >
		<interval>	<order>	< >



# Primitive data representation

## Numerical data representation

```
<qty value="100" unit="piece"/>
```

## Character data representation

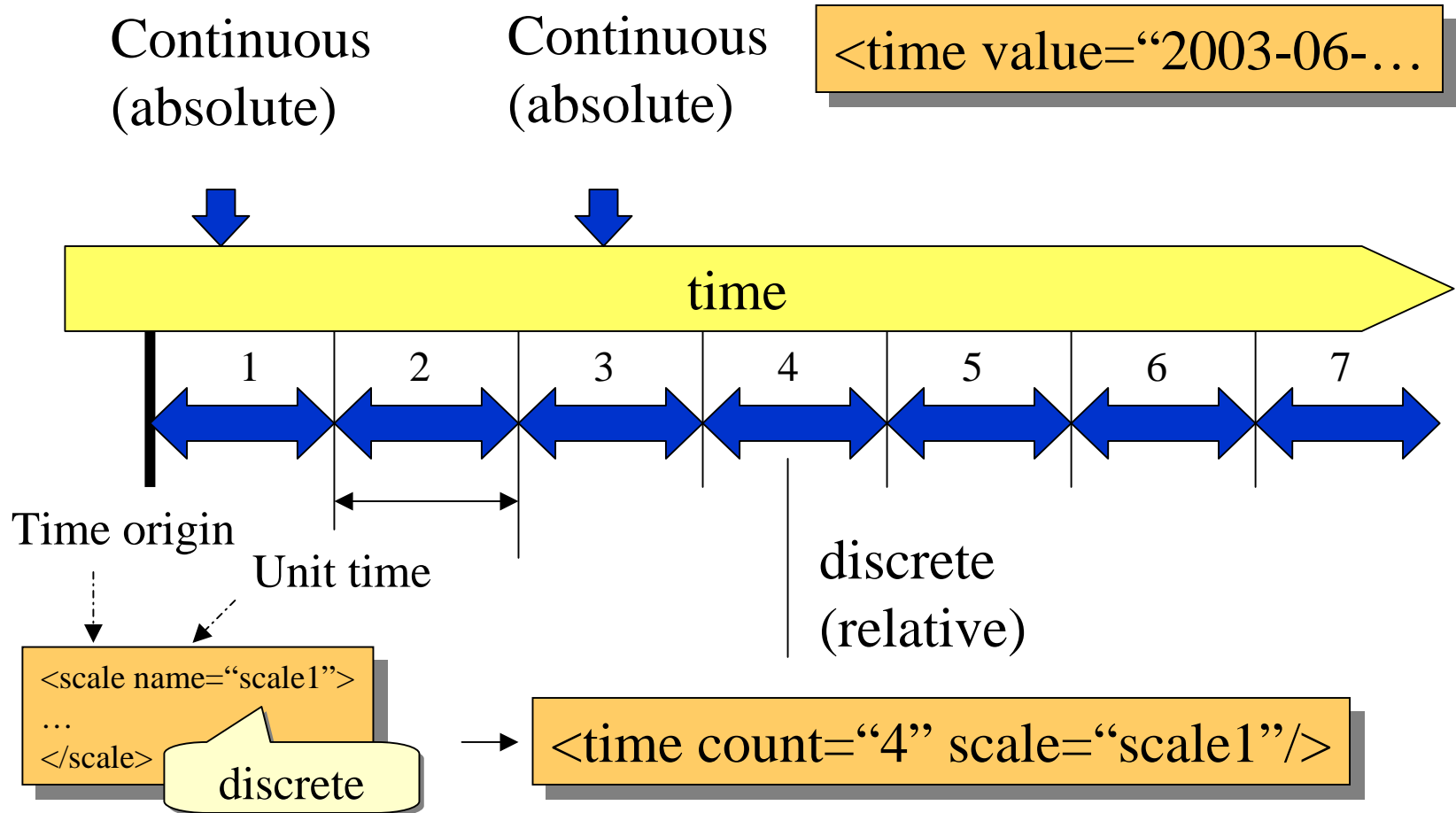
```
<char value="true"/>
```

## Date and time data representation

```
<time value="2003-06-18T13:00:00.000"/>
```



# Continuous and discrete time





# Work order representation

Operation ID

Start date

End date

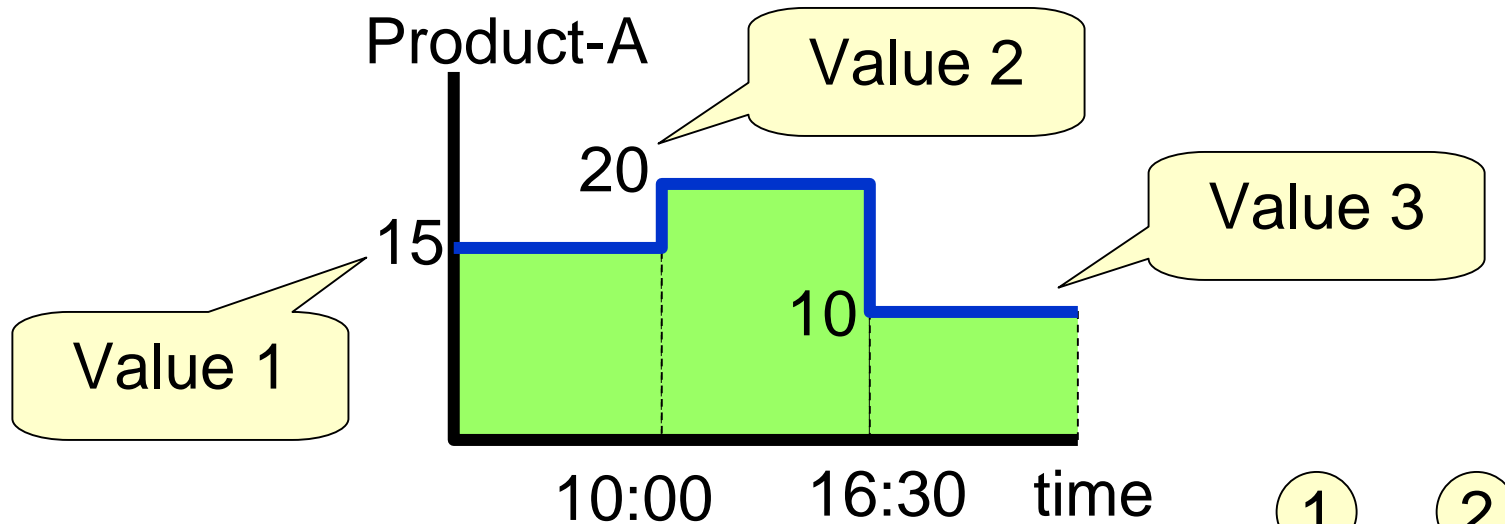
```
<operation name="YU001">  
<start><time value="2003-06-18T13:00:00"/></start>  
<end> <time value="2003-06-18T13:50:00"/></end>  
<assign resource="X02-A"/>  
</operation>  
<order name="0805001" operation="YU001">  
<qty value="20"/>  
</order>
```

machine

quantity

Order ID

# Stock level representation

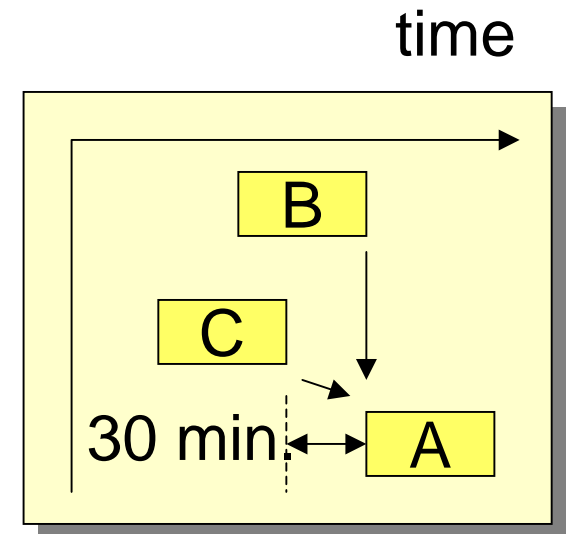
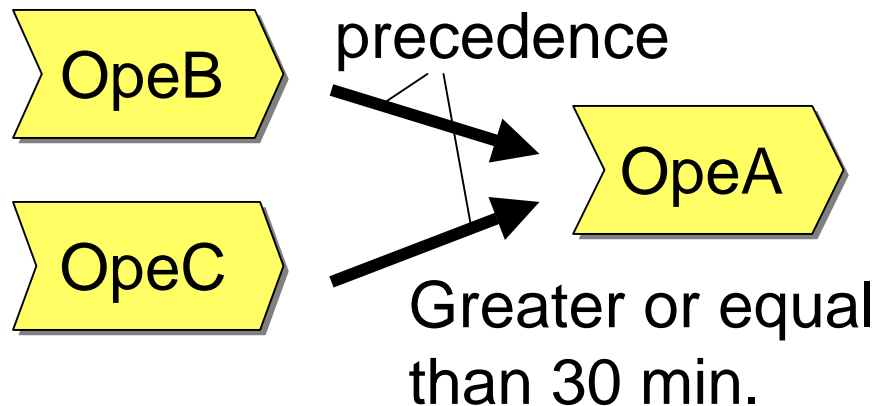


```

<item name="Product-A">
  <stock><time ref="init"/><qty value="15"/></stock>
  <stock><time value="2003-06-18T10:00:00"/><qty value="20"/></stock>
  <stock><time value="2003-06-18T16:30:00"/><qty value="10"/></stock>
</item>
  
```

# Precedence relations

```
<operation name="OpeA">  
<predecessor operation="OpeB"/>  
<predecessor operation="OpeC">  
  <duration value="PT30M"/>  
</predecessor>  
</operation>
```

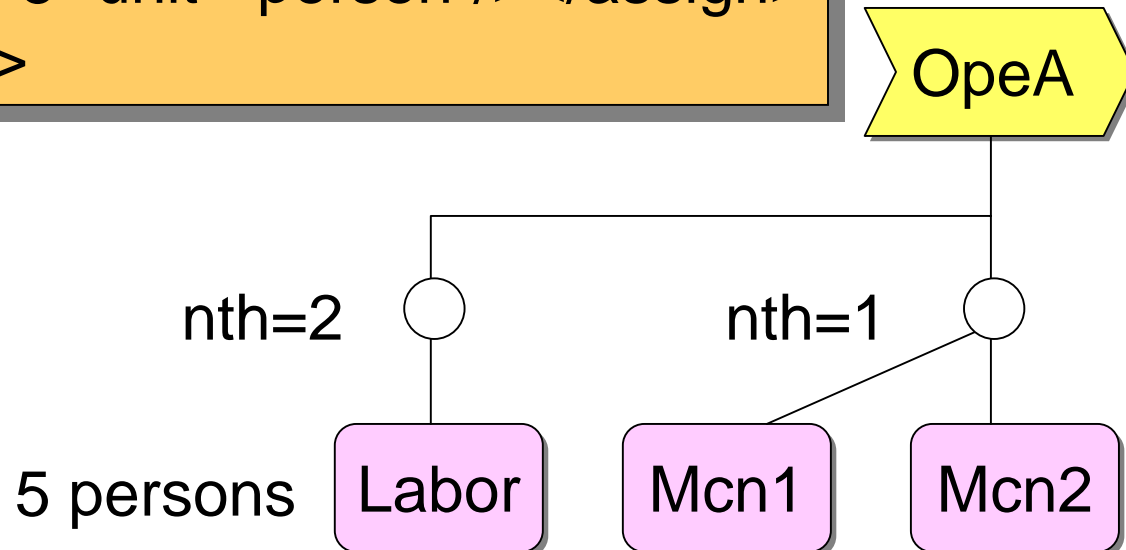


Schedule sample

# Resource assignment

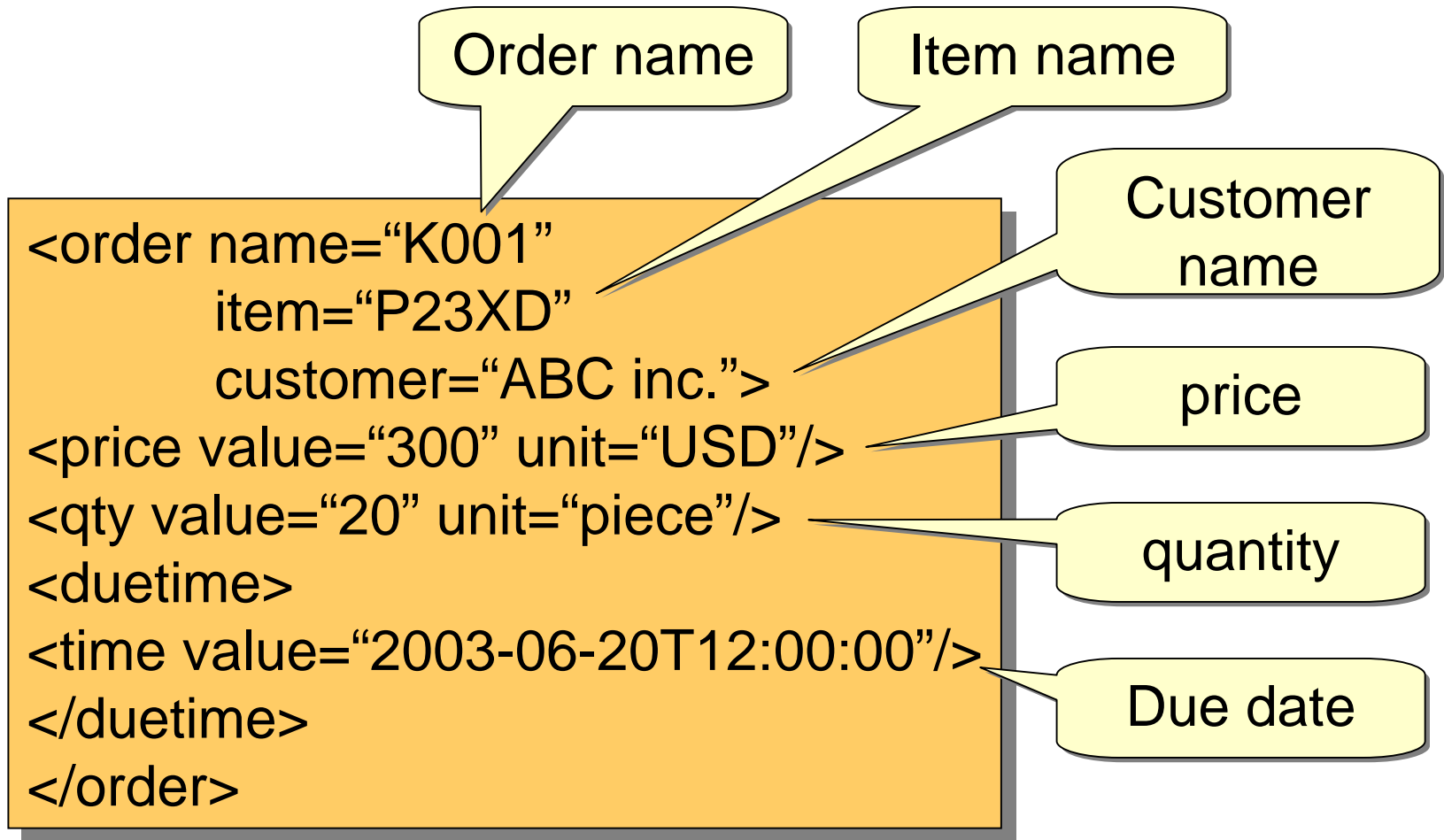
```
<operation name="OpeA">  
<assign resource="Mcn1" nth="1"/>  
<assign resource="Mcn2" nth="1"/>  
<assign resource="Labor" nth="2">  
<qty value="5" unit="person"/></assign>  
</operation>
```

} alternative resources





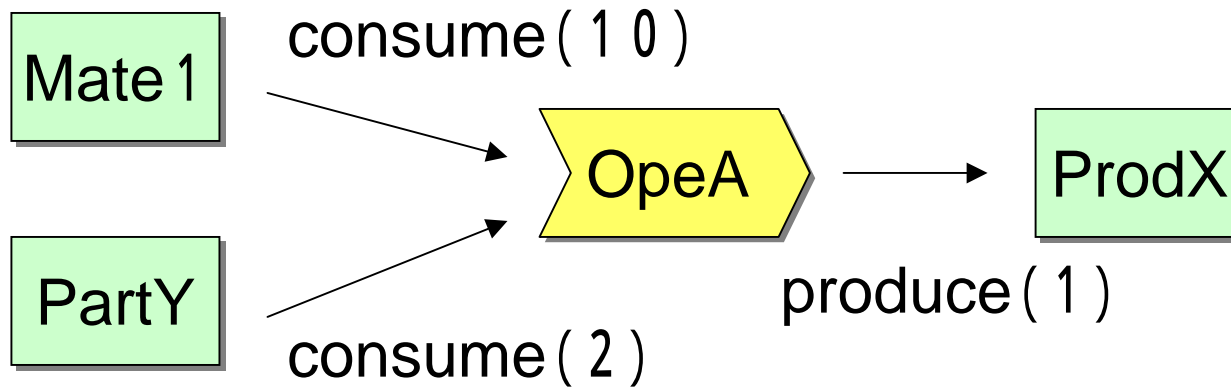
# Customer order representation





# Producing/consuming items

```
<operation name="OpeA">  
<consume item="Mate 1"><qty value="10"/></consume>  
<consume item="PartY"><qty value="2"/></consume>  
<produce item="ProdX"><qty value="1"/></produce>  
</operation>
```





# Outline

- Part 1: Define object model for XML specifications using core components, and describe XML Schema for PSLX
- Part 2: Define functional specifications of interfaces, and messages to be exchanged through the interface
- Part 3: General messaging protocols and message binding alternatives to middleware that takes part in such services



# Message category (interface)

Interface name	Interface name	Interface name
initSchedule	setOrder	setCapacity
makeSchedule	getOrder	getCapacity
setSchedule	setOption	setLot
getSchedule	getOption	getLot
setParty	setProgress	setTask
getParty	getProgress	getTask
setProduct	setStock	
getProduct	getStock	
setProcess	setLoad	
getProcess	getLoad	



# Type of message

- Request message

- Message to request a particular job to an agent

- Response message

- Message to response to a request from a client by the agent

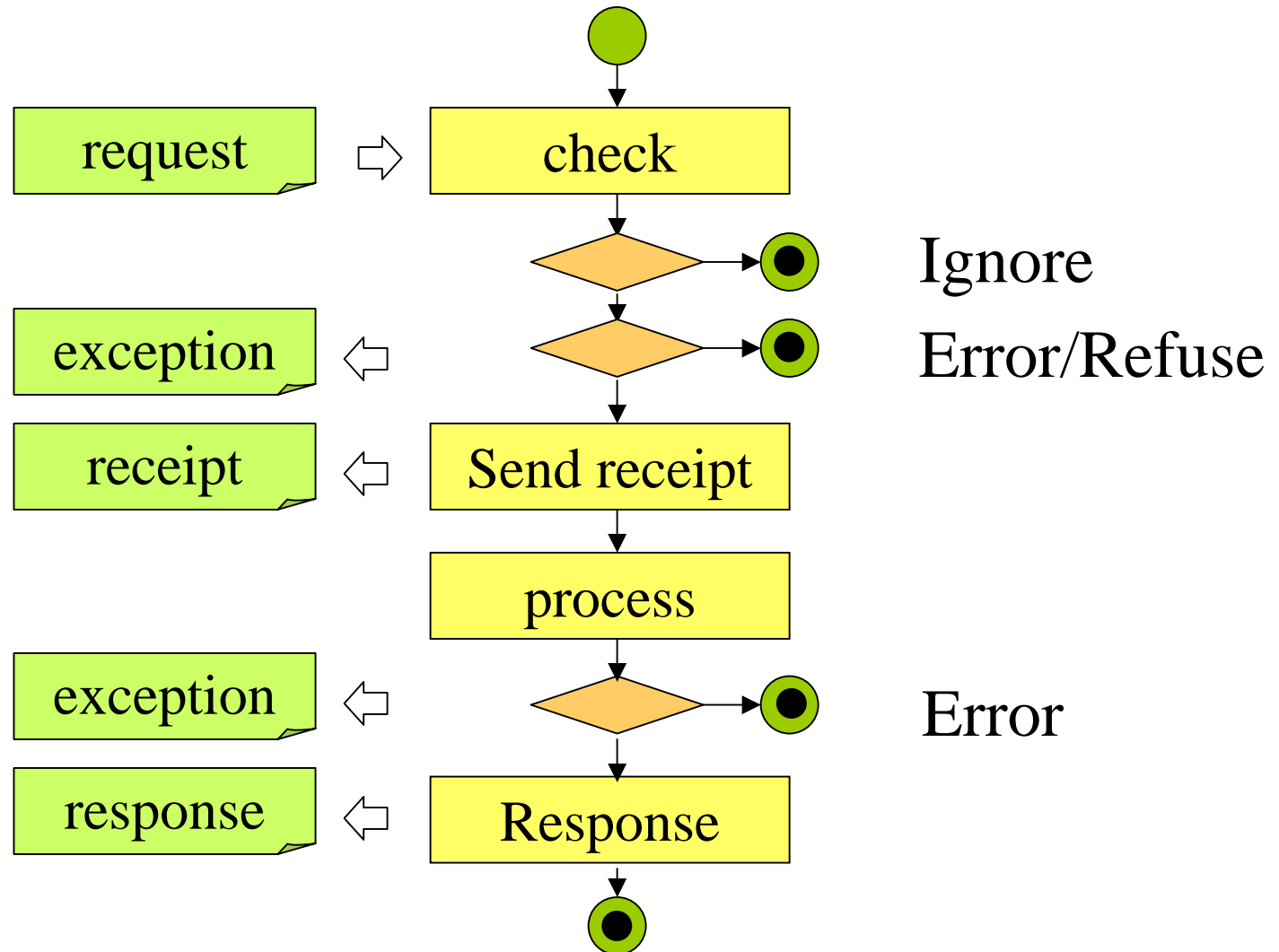
- Receipt message

- Message to send that a request message is received by the agent

- Exception message

- Message to send that a request message has an exception

# Type of message



# Basic rules of messages

## Request message

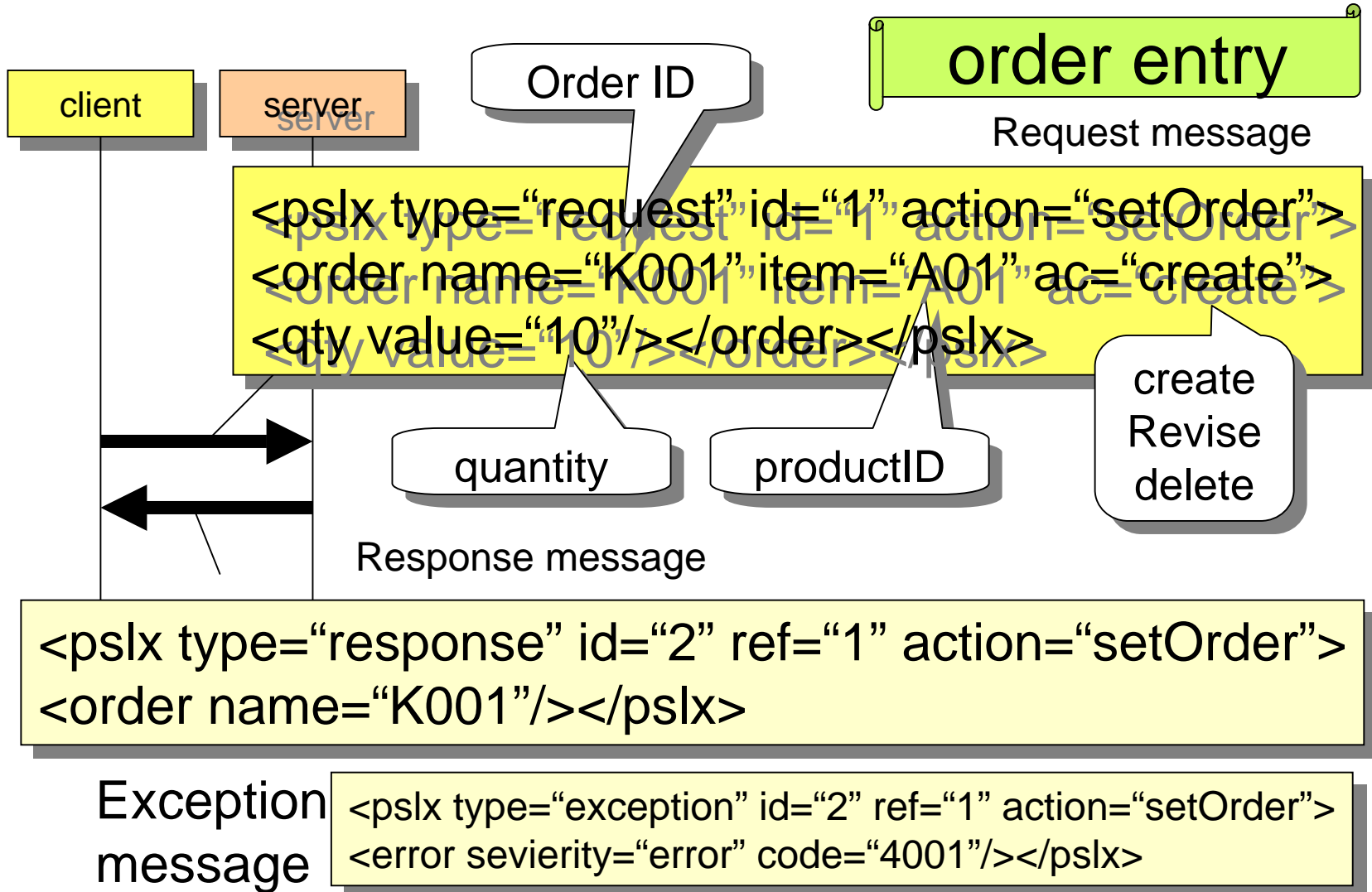
```
<pslx type="request" id="messageID" action="interface name">  
...  
PSLX data contents  
...  
</pslx>
```

## Response message

```
<pslx type="response" id="messageID" ref="refMessageID"  
action="interface name">  
...  
PSLX data contents  
...  
</pslx>
```

Type of message

# setOrder



# getOrder

## duetime inquiry

Request message

```
<pslx type="request" id="1" action="getOrder">  
<query select="#duetime"/>  
<order name="K001"/></pslx>
```

Order ID

Response message

duetime

```
<pslx type="response" id="2" ref="1" action="getOrder">  
<order name="K001">  
<duetime><time value="2003-06-20T16:00:00"/></duetime>  
</order></pslx>
```

# getSchedule

## schedule inquiry

Request message

```
<pslx type="request" id="1" action="getSchedule">  
<query select="#start"/>  
<operation name="#query">  
<assign resource="machine1"/></operation></pslx>
```

Response message

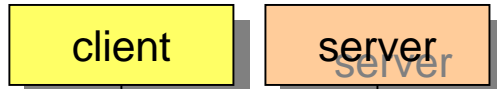
```
<pslx type="response" id="2" ref="1" action="getSchedule">  
<operation name="P005"><start><time value="2003-06-18T10:00:00"/></start>  
<operation name="P002"><start><time value="2003-06-18T16:10:00"/></start>  
<operation name="P007"><start><time value="2003-06-18T17:50:00"/></start>  
</pslx>
```

Schedule of  
machine1

# setProgress

progress report

Request message



```
<pslx type="request" id="1" action="setProgress">  
<operation name="P001" ac="revise">  
<progress status="completed"><qty value="190"/>  
</progress></operation></pslx>
```

P001 was completed

Num of yield

```
<pslx type="response" id="2" ref="1" action="setProgress">  
<operation name="P001"/></pslx>
```

Response message



# How to inquiry data

List name and price of all products,  
that size is greater than 130

Item table in DB

name	size	price
ProdA	150	2000
ProdB	100	2100
ProdC	190	2500

```
SELECT name, price FROM item  
WHERE size>130
```

SQL specification



```
<item name="#query">  
<spec name="size"><qty><min value="130"/></qty></spec>  
<query select="price"/>  
</item>
```

PSLX specification



```
<item name="ProdA">  
<price value="2000"/></item>  
<item name="ProdC">  
<price value="2500"/></item>
```



# How to inquiry data

Use #query if you inquiry by any conditions

Describe field items you want in the result.  
Name is default

Restriction by <min> and <max> tag for each data field

```
<item name="#query">  
<query select="price"/>  
<spec name="size"><qty><min value="130"/></qty></spec>  
</item>
```

All data described inside the tag are used as conditions



# Outline

- Part 1: Define object model for XML specifications using core components, and describe XML Schema for PSLX
- Part 2: Define functional specifications of interfaces, and messages to be exchanged through the interface
- Part 3: General messaging protocols and message binding alternatives to middleware that takes part in such services



# Sync vs. Async sequence

- Sync communication
  - sender should wait the response from the target agent after sending a message. sender cannot do any other job until the response will come.
  - receiver is always waiting for a message and responds immediately.
- Async Communication
  - sender can do other jobs after sending a message. When the response will come, sender stops its job to deal with the response.
  - receiver respond to a message after its current jobs are completed. Therefore, you may wait the response for long time depending on the load level of the agent.



# Binding to HTTP (or RPC)

## Sync sequence

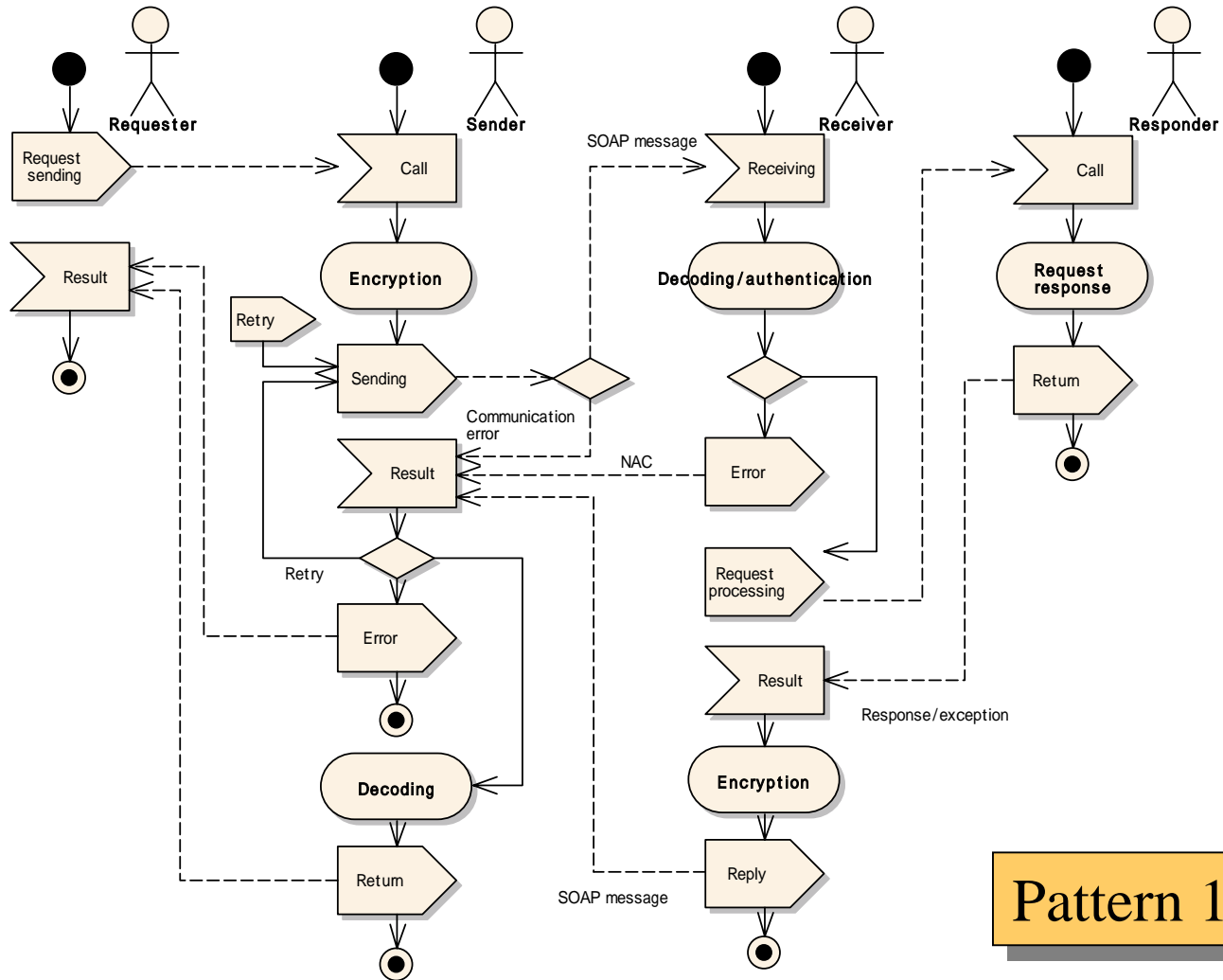
Pattern no	send	receive
1	Request message	Response message
2	Request message	Exception message

## Async sequence

3	Request message	---
4	Response message	---
5	Receipt message	---
6	Exception message	---
7	---	Request message
8	---	Response message
9	---	Receipt message
10	---	Exception message



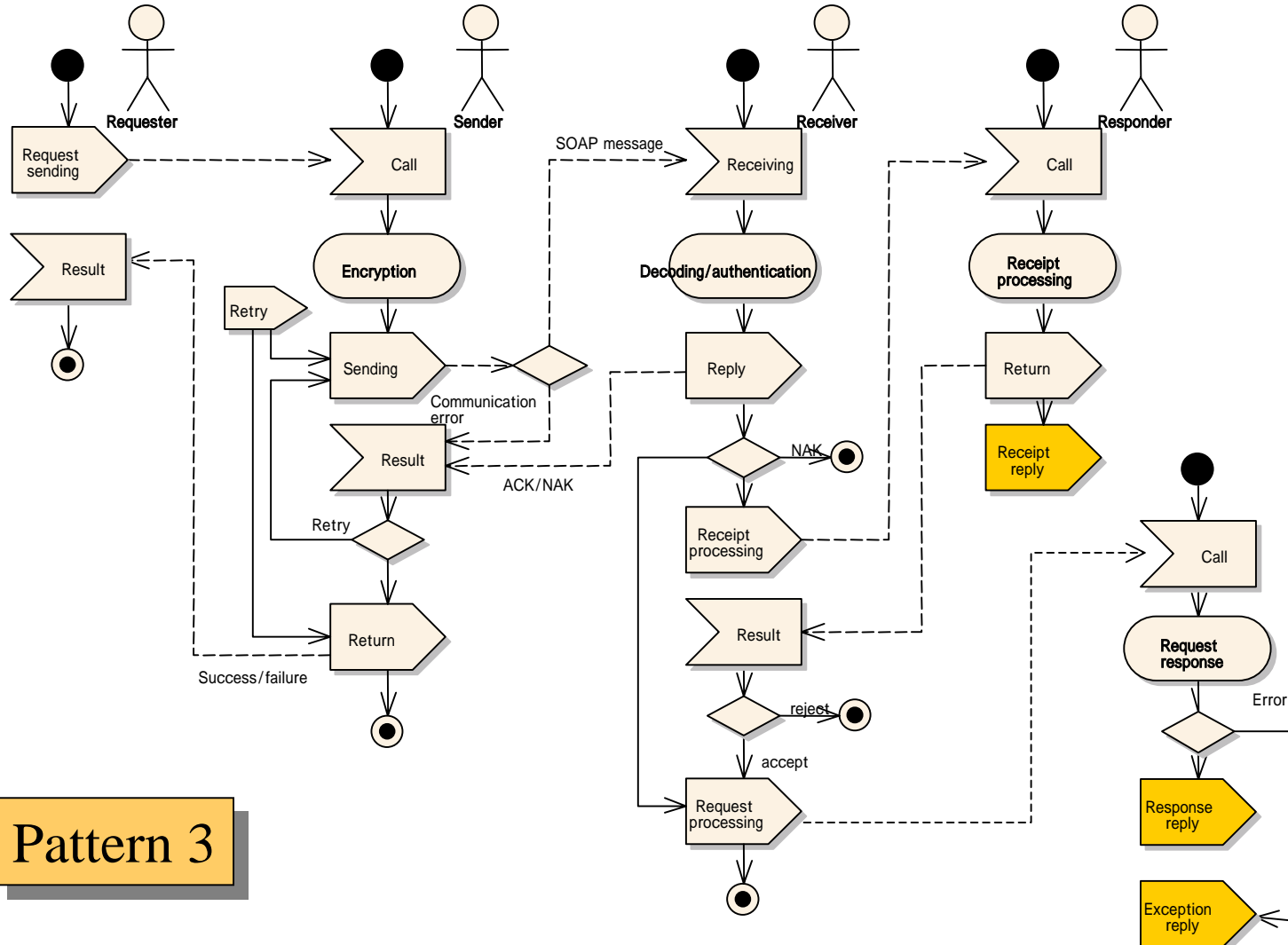
# Sample of sync sequence



Pattern 1,2



# Sample of async sequence



Pattern 3



# Middleware selection

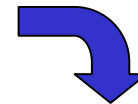
- **WebService**
  - Many Tools are available, simple mechanism
  - No security concern in B2B environment
- **ebXML Message Service**
  - Global standard bodies are support. Secure communication
  - Few examples in practical use. Few tools.
- **RosettaNet RNIF**
  - Many industrial cases studies. Many commercial products.
  - Inconsistent to global standards. Expensive to use.



# Binding to Webservice

```
<soapenv:Envelope>  
  <soapenv:Body>  
    <pslxMethodSync>  
      <pslxBody>...  
      PSLX messages...  
    </pslxBody>  
    <toLocal>programID</toLocal>  
  </pslxMethodSync>  
</soapevsn:Body>  
</soapenv:Envelope>
```

request



response

```
<soapenv:Envelope>  
  <soapenv:Body>  
    <pslxMethodSyncResponse>  
      <pslxMethodSyncReturn>  
        PSLX messages  
      </pslxMethodSyncReturn>  
    </pslxMethodSyncResponse>  
  </soapenv:Body>  
</soapenv:Envelope>
```

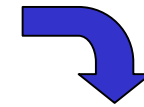
Sync sequence

# Binding to Webservice

```
<soapenv:Envelope>
  <soapenv:Body>
    <pslxMethodAsync>
      <messageID>messageID</messageID>
      <refID>referenceID</refID>
      <pslxBody>PSLXmessages</pslxBody>
      <toLocal>receiverProgrammID</toLocal>
      <fromURI>senderServerURI</fromURI>
      <fromLocal>senderProgramID</fromLocal>
      <clientID>senderClientID</clientID>
    </pslxMethodAsync>
  </soapenv:Body>
</soapenv:Envelope>
```

request

Async sequence (push)



response

```
<soapenv:Envelope>
  <soapenv:Body>
    <pslxMethodAsyncResponse>
      <pslxMethodAsyncReturn>
        messageID
      </pslxMethodAsyncReturn>
    </pslxMethodAsyncResponse>
  </soapenv:Body>
</soapenv:Envelope>
```



# Binding to Webservice

Tag name	type	sync	acync request	async response
toLocal	string	V	V	V
fromURI	string	-	O	-
fromLocal	string	-	V	-
clientID	string	-	O	O
messageID	string	-	O	O
refID	string	-	-	O
pslxBody	string	O	O	O

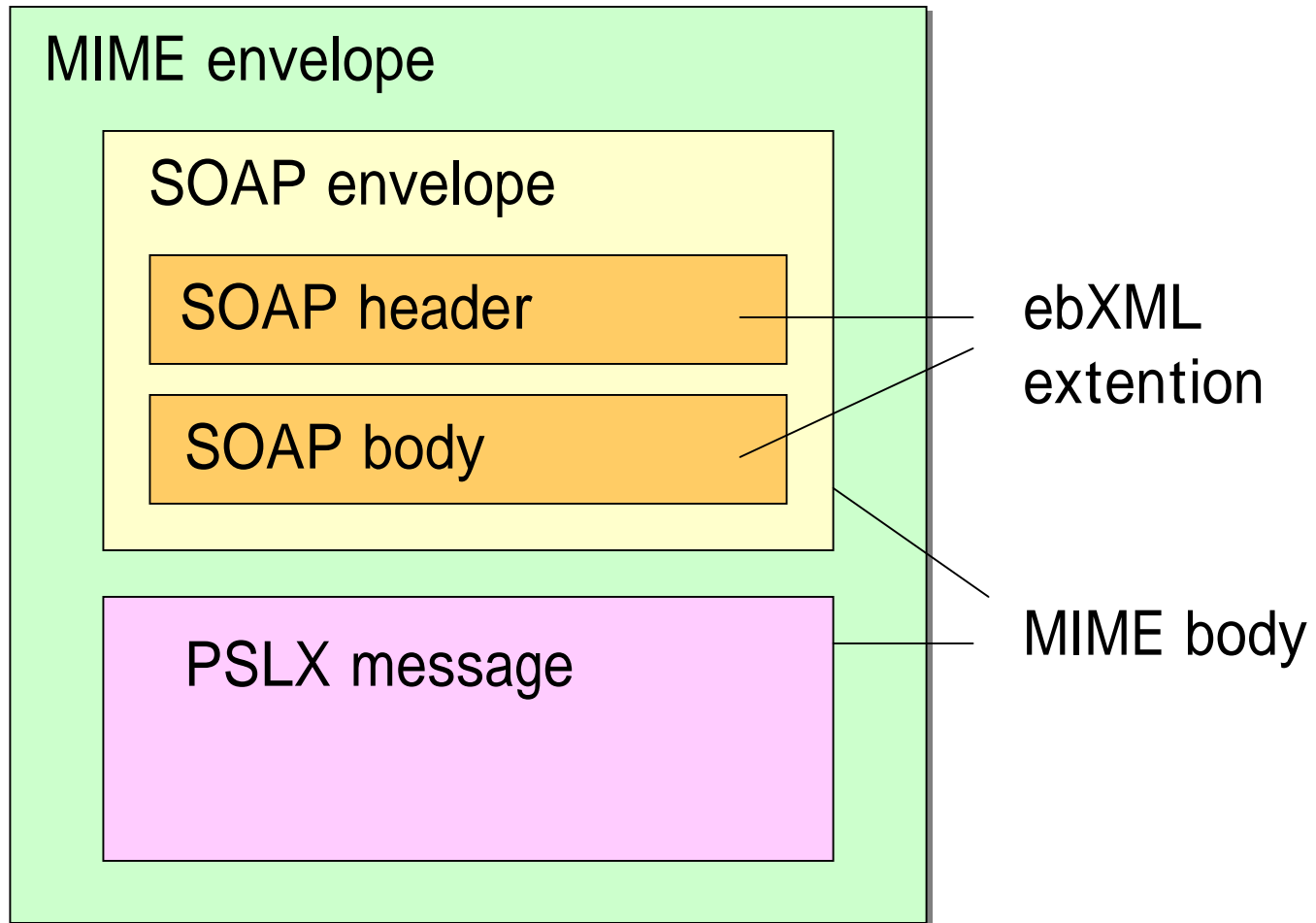
O ... necessary

V ... optional

- ... invalid



# Binding to ebXML-MS





# Binding to ebXML-MS

```
<soapenv:Envelope>
  <soapenv:Body>
    <MessageHeader>
      <From>
        <PartyID type="URI">senderServerURI</PartyID>
        <PartyID type="local">senderProgramID</PartyID>
        <PartyID type="client">senderClientID</PartyID>
      </From>
      <To>
        <PartyID type="URI">reserverServerURI</PartyID>
        <PartyID type="local">reseiverProgramID</PartyID>
        <PartyID type="client">receiverClientID</PartyID>
      </To>
      <Service>PslxServise</Service>
      <Action>interfaceName</Action>
      <MessageData>
        <MessageId>messageID</MessageId>
        <RefToMessageId>referenceID<RefToMessageId>
      </MessageData>
    </MessageHeader>
  </soapenv:Body>
</soapenv:Envelope>
```

} Sender  
information

} Receiver  
information

} Name of interface

} Message  
identifier



# Binding to ebXML-MS

Tag name	type	request	response
From:PartyID:URI	string	O	-
From:PartyID:local	string	V	-
From:PartyID:client	string	O	-
To:PartyID:URI	string	O	O
To:PartyID:local	string	V	V
To:PartyID:client	string	-	O
Service	string	O	O
Action	string	O	O
MessageId	string	O	O
RefToMessageId	string	-	O



Thank you

<http://www.pslx.org>