Service Referrals in BPEL-based Choreographies

Gero Decker[1], Oliver Kopp[2], Frank Puhlmann[1]

HPI Hasso Plattner Institut
IT Systems Engineering | Universität Potsdam

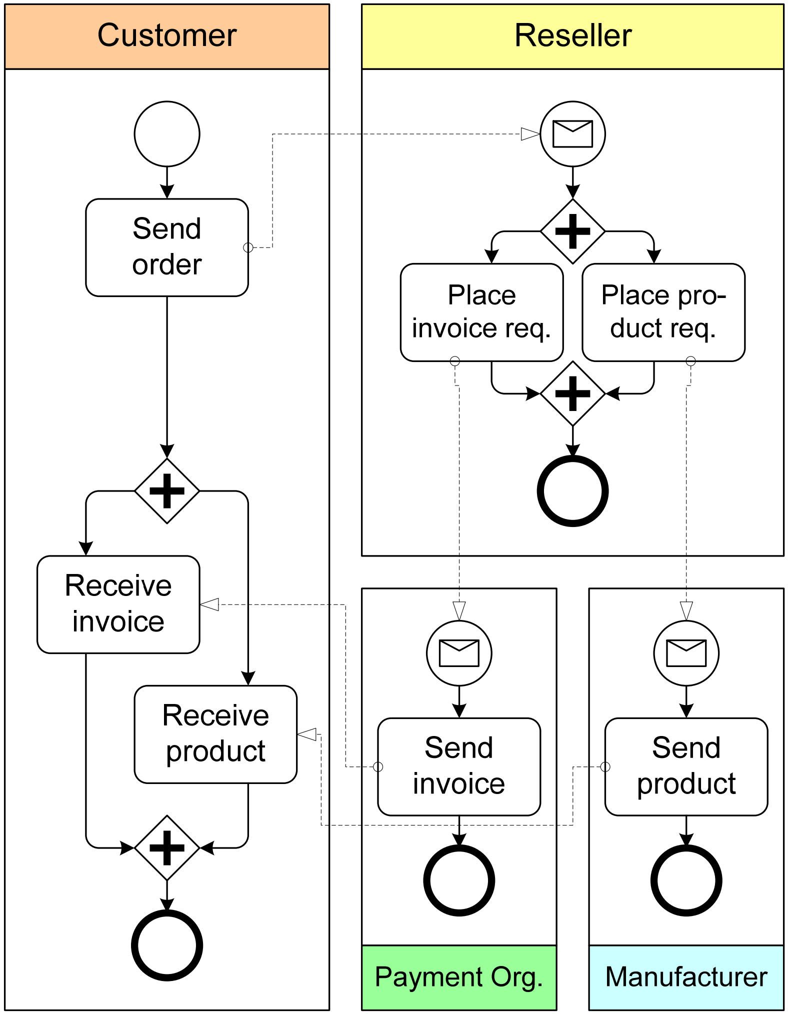
IAAS
Universität Stuttgart
Germany

Agenda

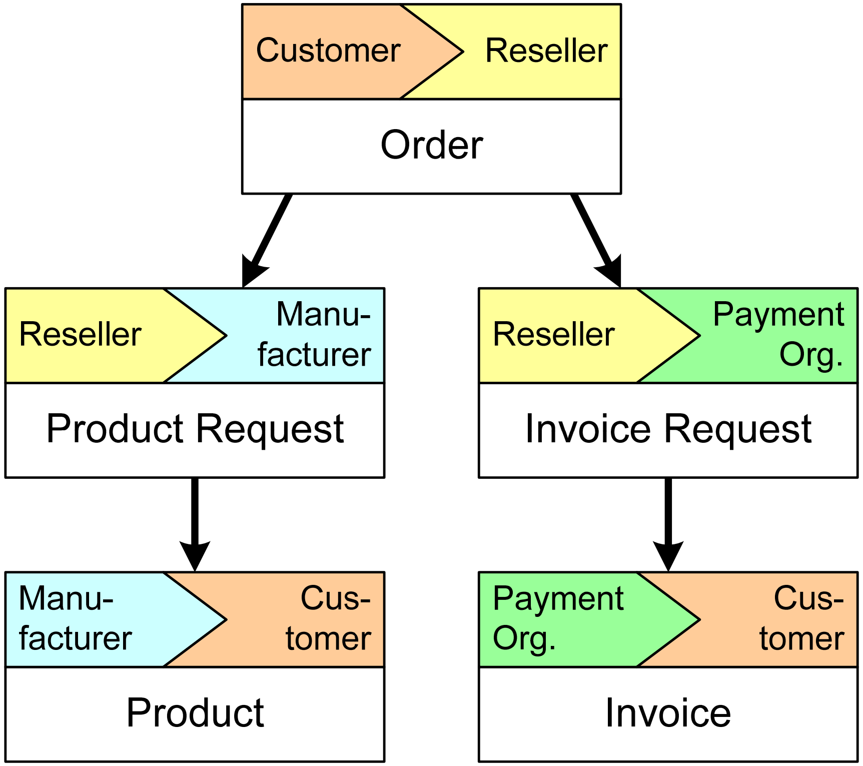
- Choreographies
- BPEL-based Choreographies
- Service Referrals in BPEL4Chor

The Reseller Choreography

- Interconnection model

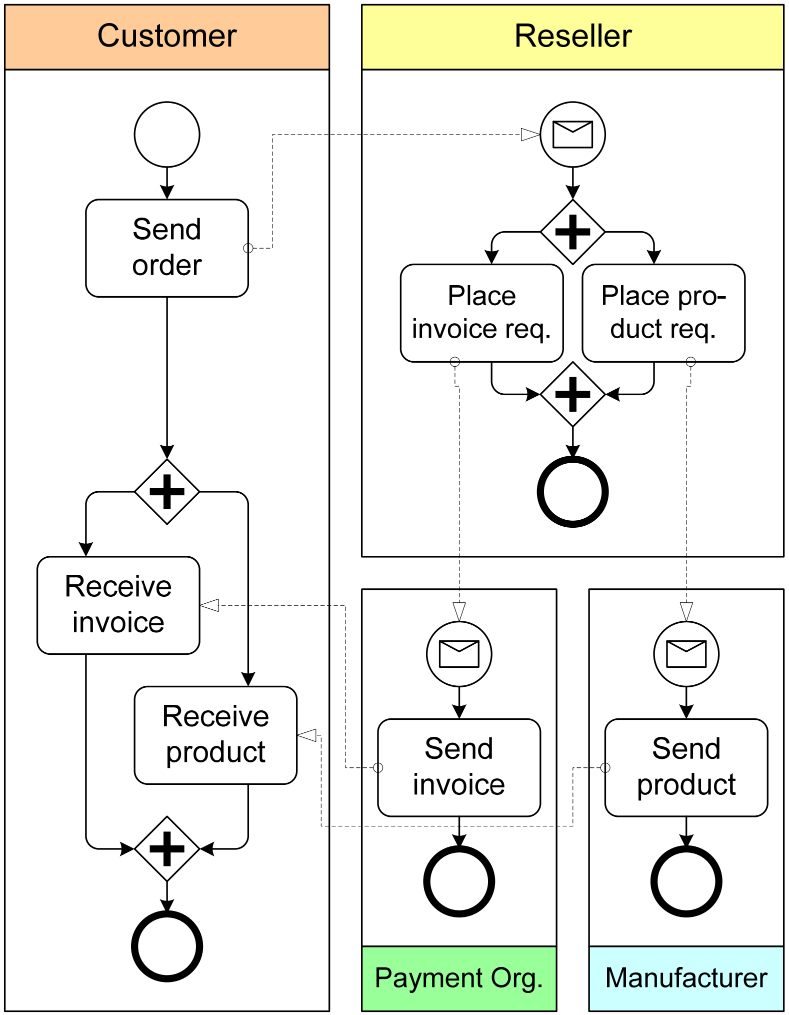


- Interaction model

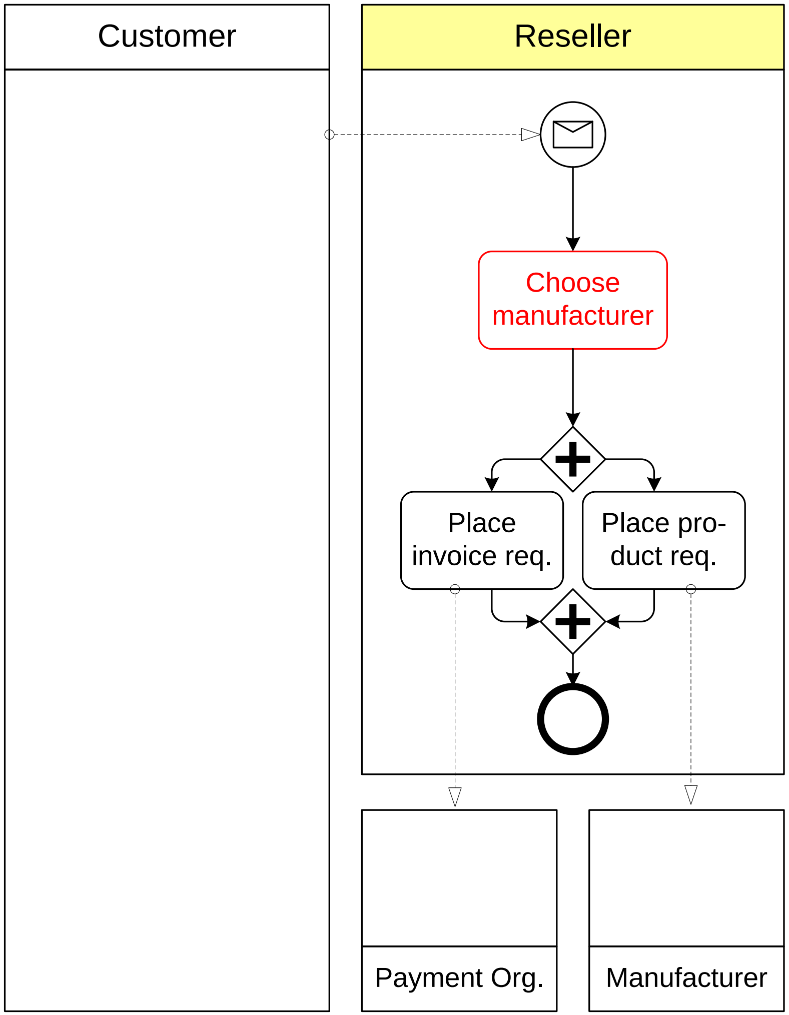


How to derive the orchestration?

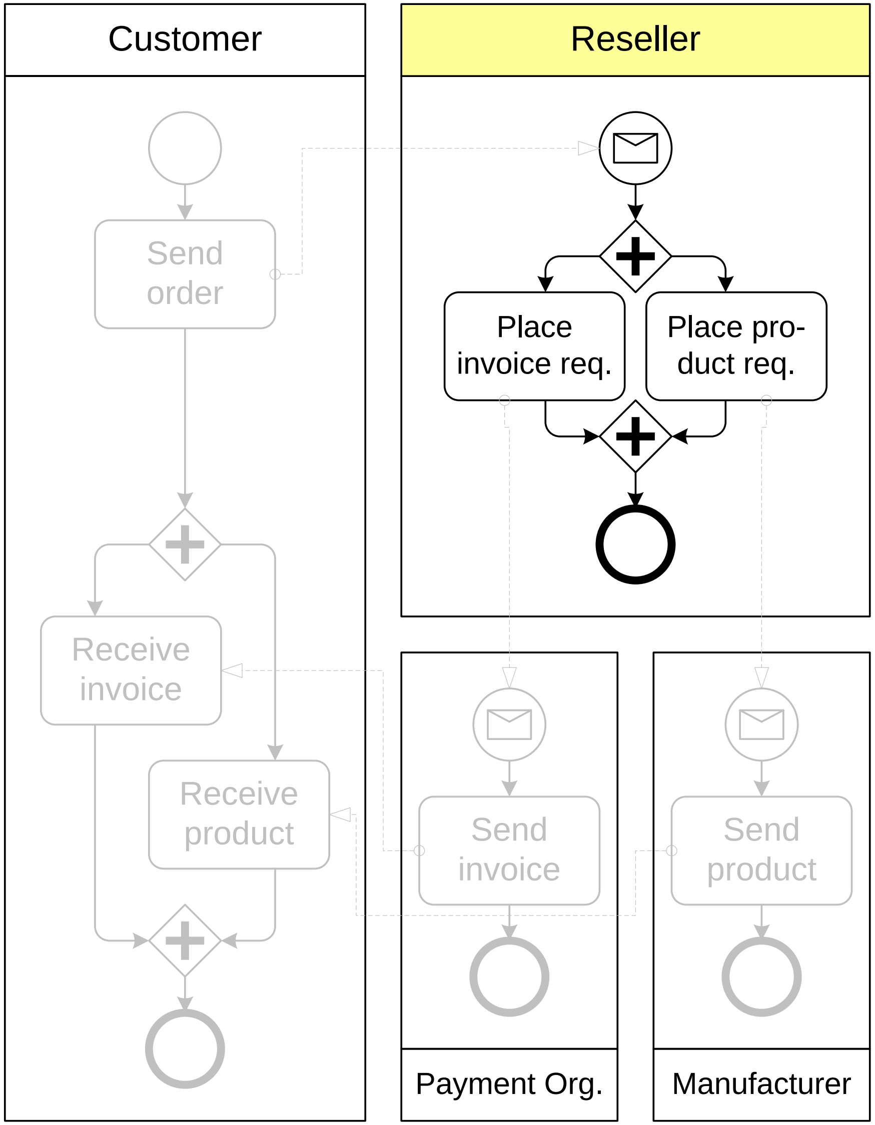
- Choreography



- Orchestration



Reseller Choreography in BPEL4Chor

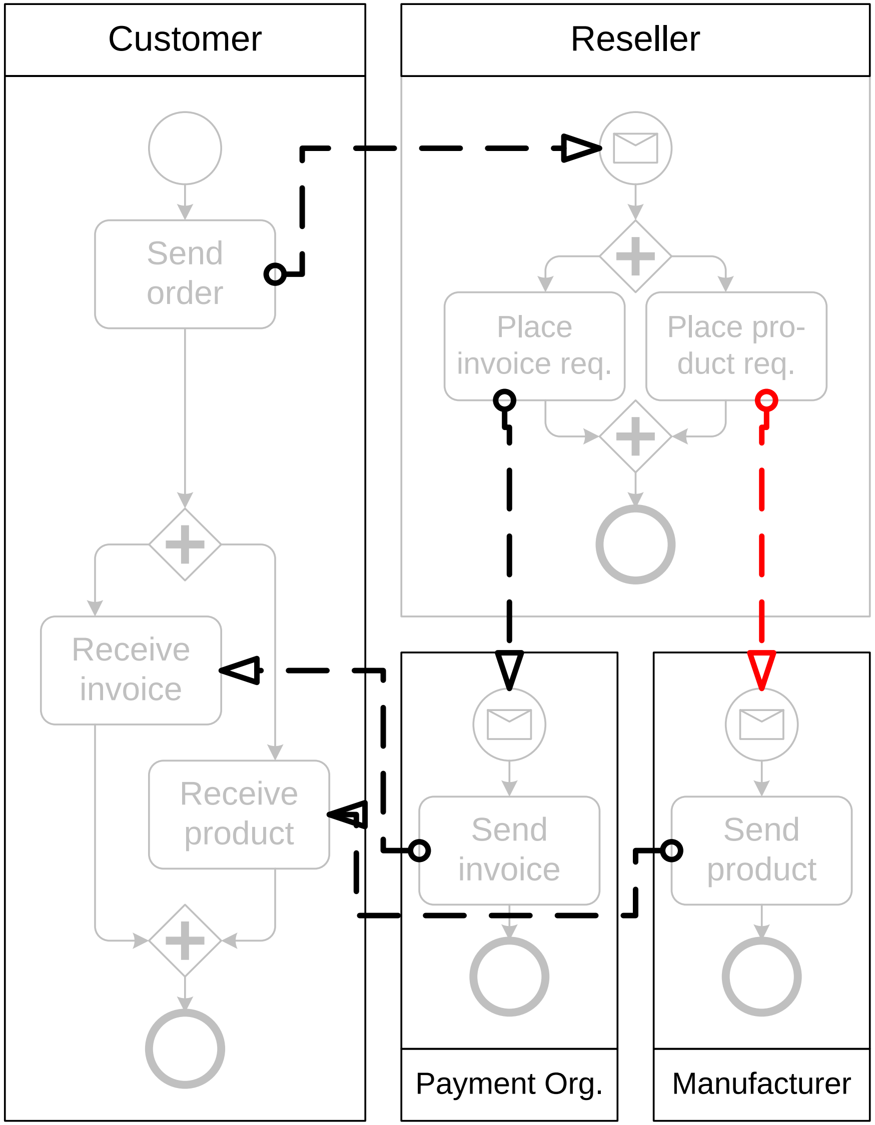


Resellers's behavior:

```
<receive
  name="getOrder" />
<flow>
  <invoke
    name="PlaceInvoiceReq" />
  <invoke
    name="PlaceProductReq" />
</flow>
```

- Behavior of the other participants specified analogous

The interconnections



The interconnection is formed by links:

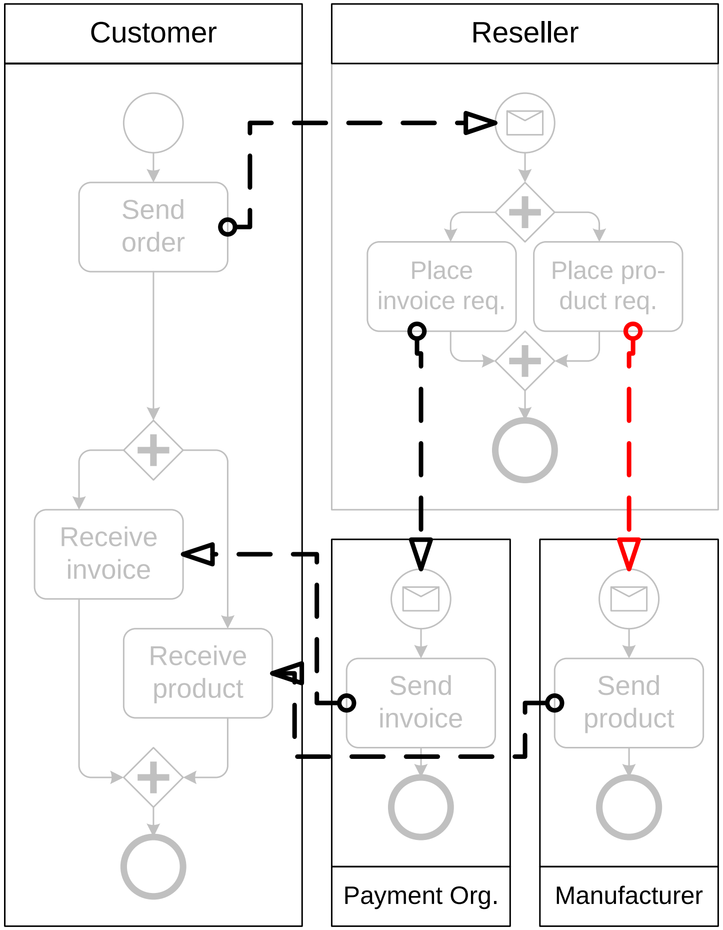
```
<messageLink
   name="productRequestLink"
   sender="reseller"
   sendActivity=
      "PlaceProductRequest"
   receiver="manufacturer"
   receiveActivity="getReq"
   messageName="productReq"

   participantRefs="customer"
/>
```

Presented by Oliver Kopp

© Gero Decker, Oliver Kopp, Frank Puhlmann

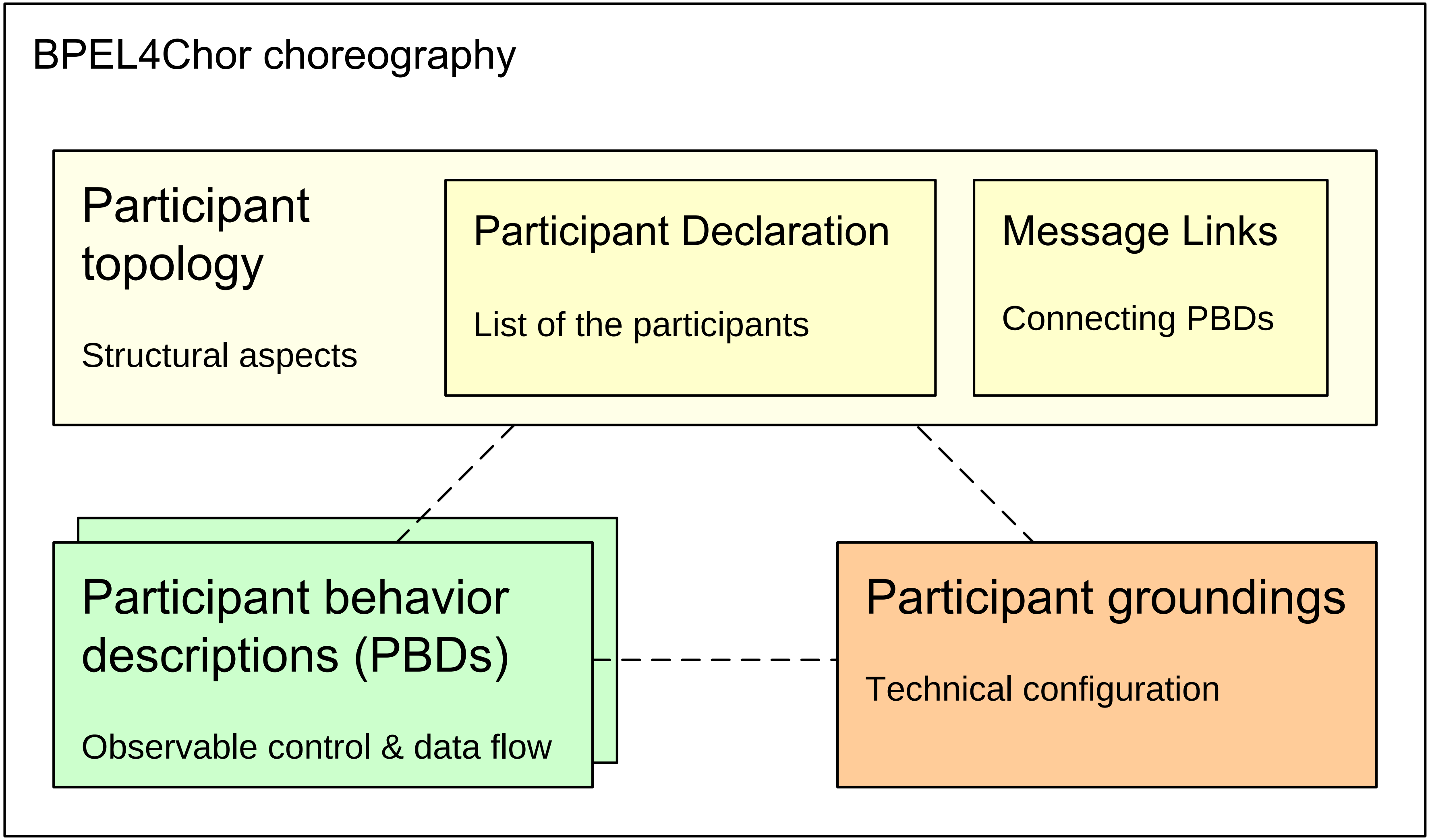
The execution

- Mapping message links to porttypes / operations
  - "Grounding"



```
<messageLink
   name="productRequestLink"
   portType="man:manufacturerPT"
   operation="placeProductRequest"
   />

<participantRef name="customer"
   WSDLproperty=
     "msgs:customerProperty" />
```

The complete picture of BPEL4Chor



BPEL4Chor choreography

Participant topology

Structural aspects

Participant Declaration

List of the participants

Message Links

Connecting PBDs

Participant behavior descriptions (PBDs)

Observable control & data flow

Participant groundings

Technical configuration

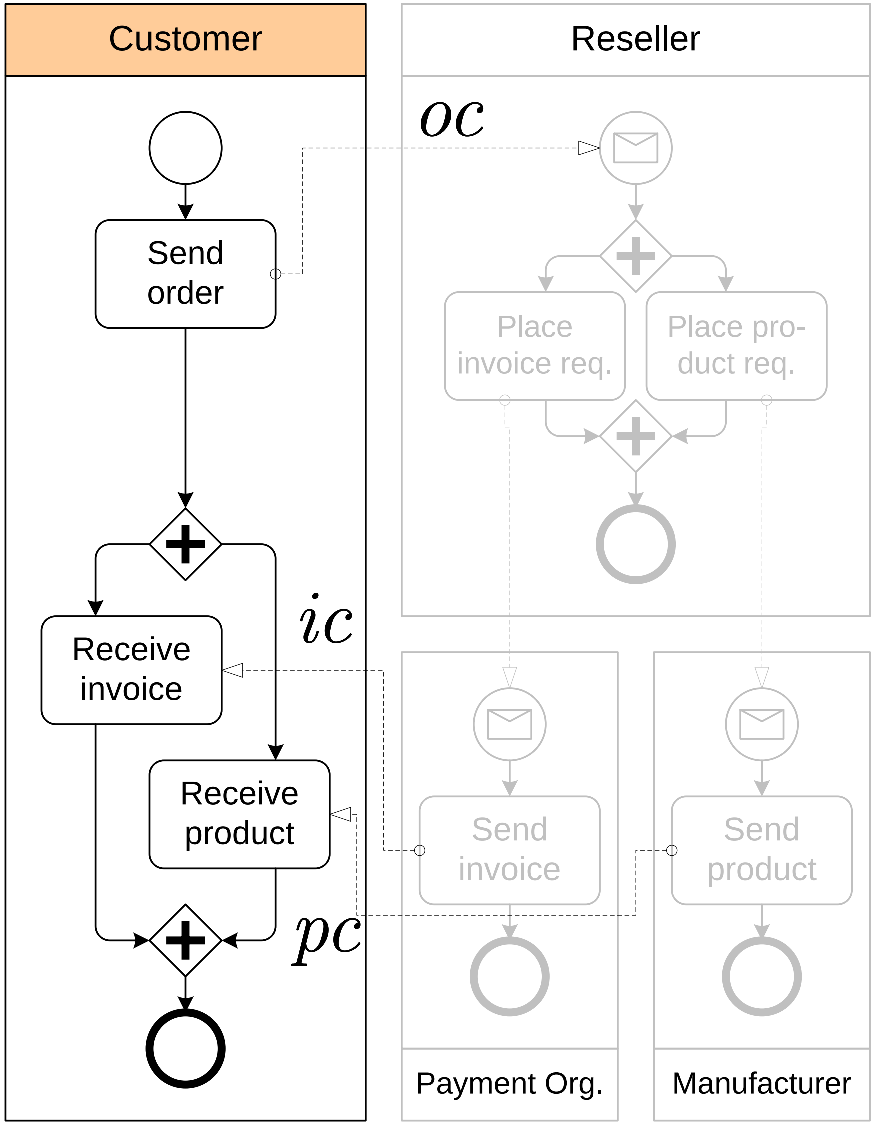
Reseller Choreography in Pi-Calculus

- Pi-Calculus because of link-passing mobility

- The complete system consists of the four participants executing in parallel

$$SYS \overset{def}{=} (C \mid R \mid P \mid M)$$

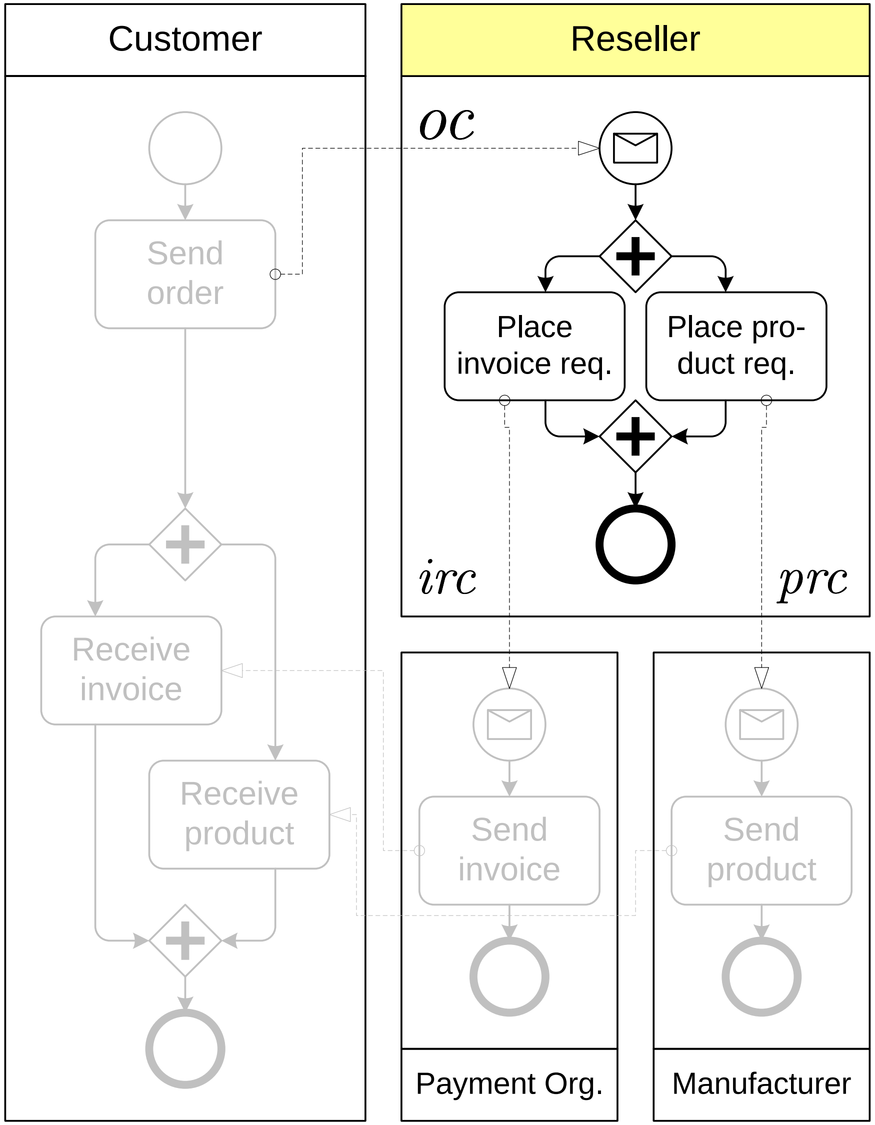
The Customer



- Customer creates order and callback-channels (ν)
- Uses public order channel “oc”

$$C \overset{def}{=} (\nu order, ic, pc)\ \overline{oc}\langle order, ic, pc\rangle.\ (ic(invoice)\ |\ pc(product))$$

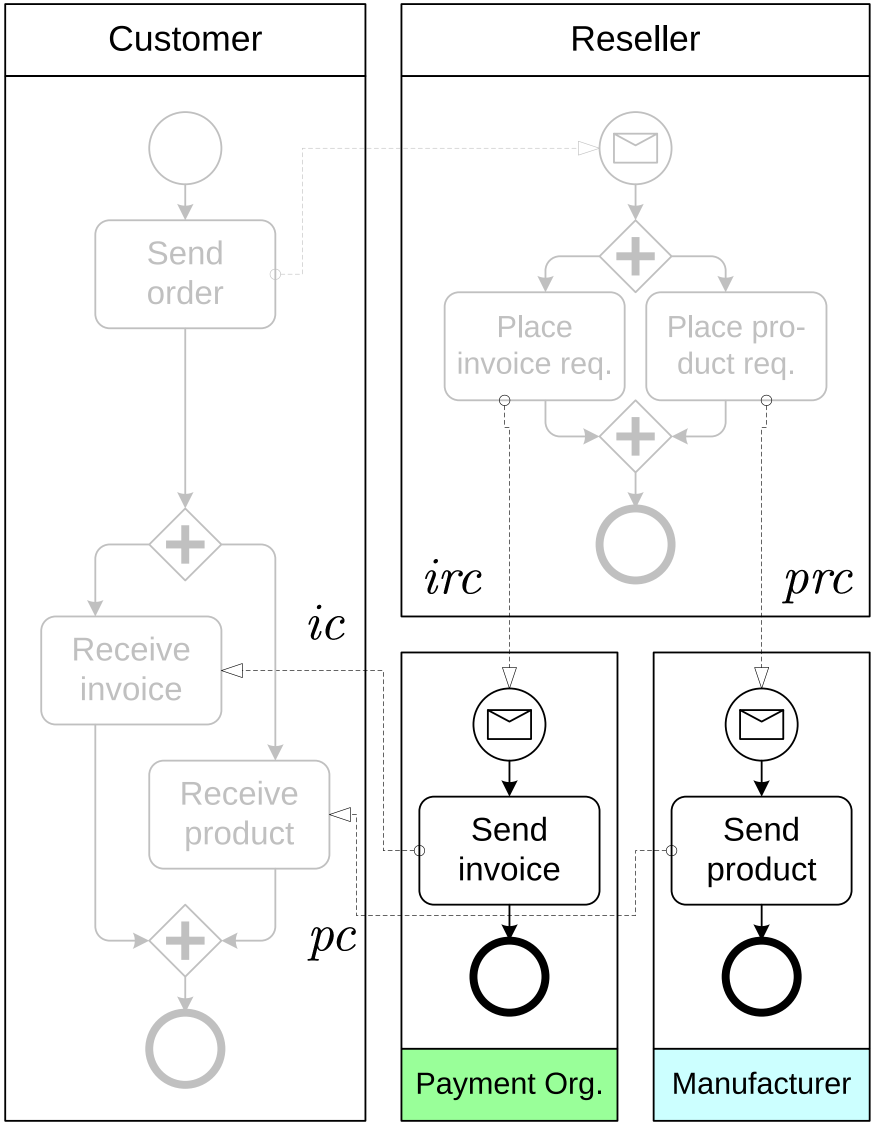
The Reseller



- Reseller sends customer's callback-channels
- Uses public invoice and payment request channels

$$
\begin{aligned}
R \overset{def}{=} \; & (\nu invoiceReq, productReq) \\
& oc(order, ic, pc). \\
& (\overline{irc}\langle invoiceReq, ic\rangle \mid \\
& \; \overline{prc}\langle productReq, pc\rangle \;)
\end{aligned}
$$

The Manufacturer and the Payment Organization



- Receive the callback-channel via public channel
- Create invoice/product
- Use the callback-channel to send invoice/product

$$P \overset{def}{=} (\nu invoice)\, irc(invoiceReq, ic).\, \overline{ic}\langle invoice\rangle$$

$$M \overset{def}{=} (\nu product)\, prc(productReq, pc).\, \overline{pc}\langle product\rangle$$

Summary

- BPEL4Chor as interconnection model
- Link passing mobility as first-class citizen
- Pi-Calculus as formalism

- Ongoing work
  - Complete mapping: BPEL4Chor to pi-calculus
  - Interaction models vs. interconnection models
  - Conformance