

# **BPMN**

## **BUSINESS PROCESS MODEL AND NOTATION**

**OVERVIEW OF BUSINESS PROCESS  
MODEL AND NOTATION (BPMN) LANGUAGE FOR  
MODELING BUSINESS PROCESSES**

**PETER R. EGLI  
INDIGOO.COM**

## Contents

1. What is BPMN?
2. BPMN Main Parts
3. BPMN Scope
4. BPMN Core Concepts
5. BPMN versus BPEL
6. BPMN notation elements
7. BPMN 2.0 versus BPMN 1.2

## 1. What is BPMN?

BPMN is a **graphical modeling language and notation** for business processes (=graphical DSL – Domain Specific Language) with the following goals:

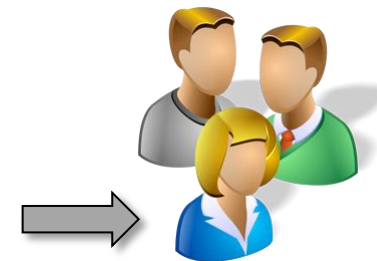
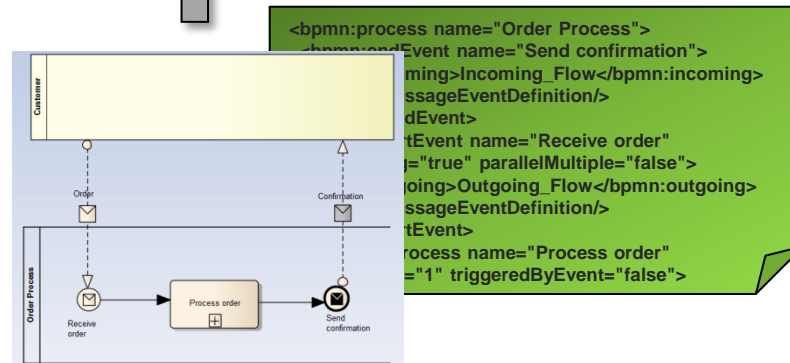
1. Common language understandable for different stakeholders.
2. Visualization of execution languages like WSBPEL.
3. Interchange format between tools for process description and diagrams.



**Business people  
use and monitor processes**



**Business analysts  
define processes**



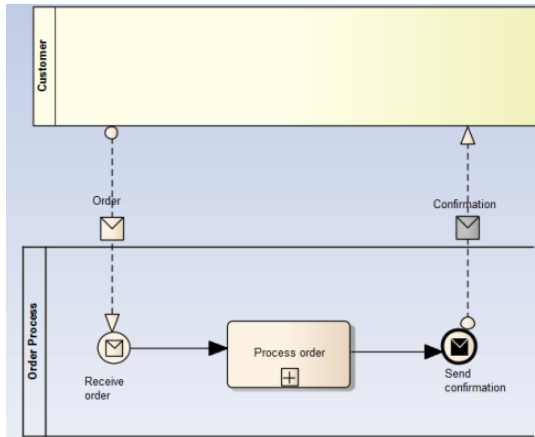
**Developers  
implement processes**

**BPMN = common language & notation**

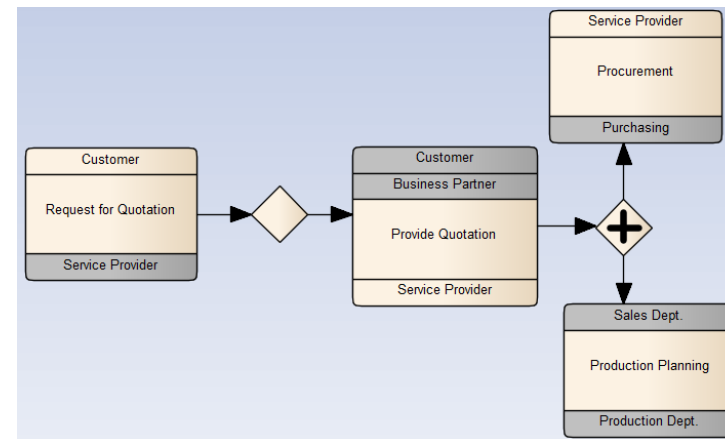
## 2. BPMN Main Parts (1/3)

### A. Diagram types:

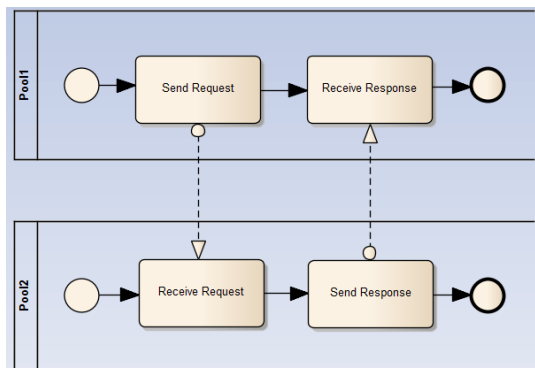
BPMN defines the following 4 diagram types.



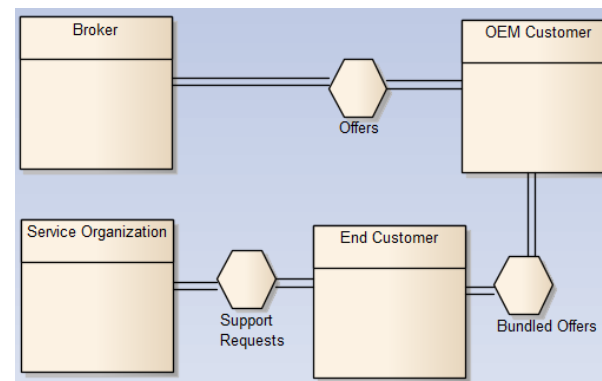
Orchestration/Process



Choreography



Collaboration

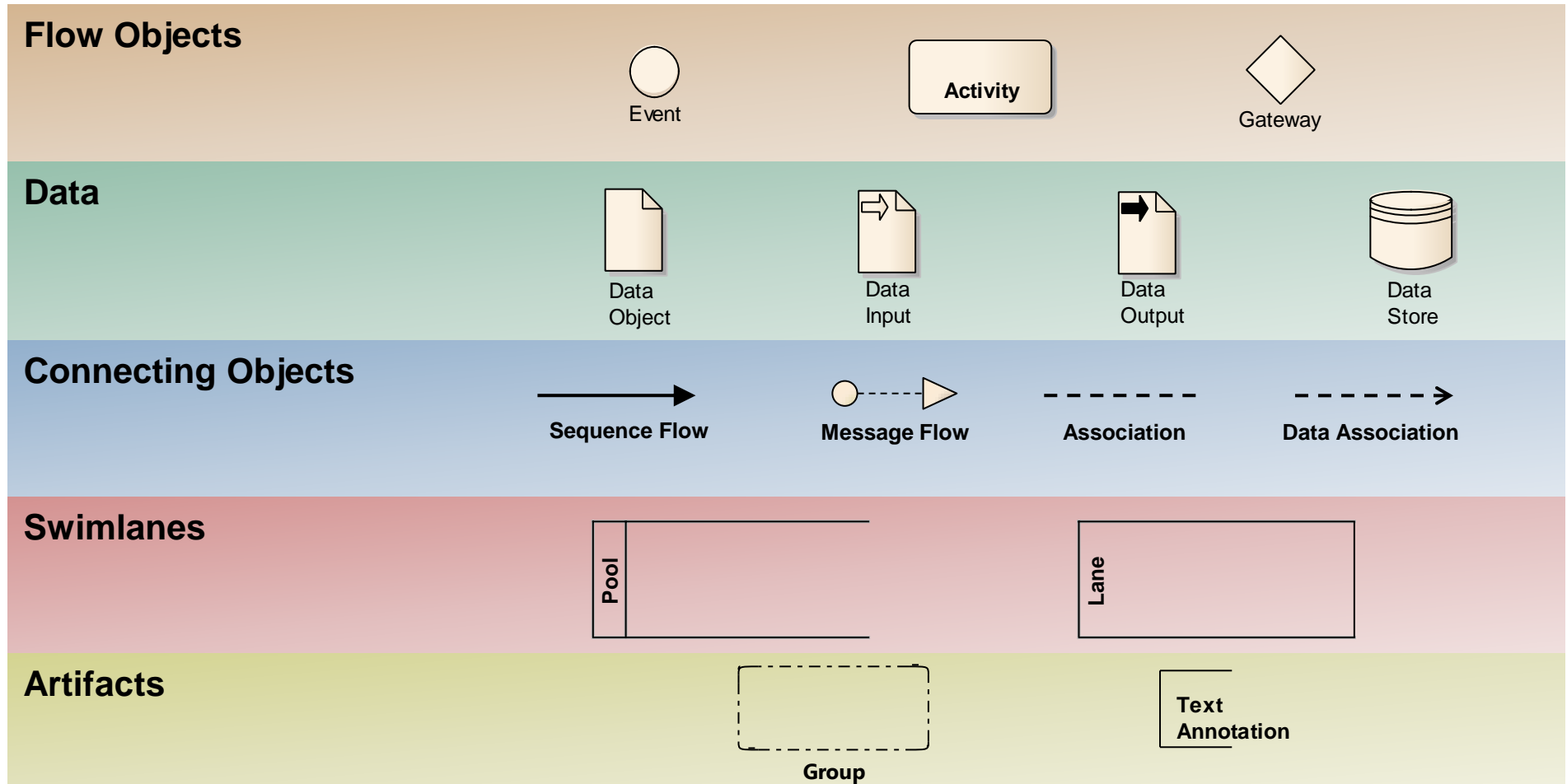


Conversation

## 2. BPMN Main Parts (2/3)

### B. Business process notation:

BPMN defines a common set of modeling elements to be used in BPMN diagrams.

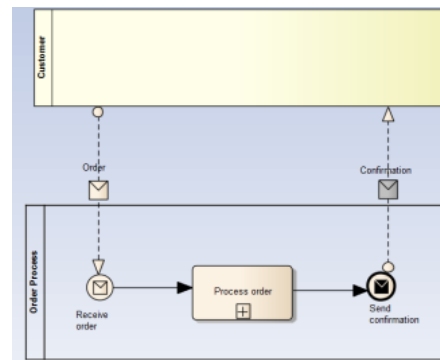


## 2. BPMN Main Parts (3/3)

### C. Mapping to execution language:

BPMN defines a mapping to the WSBPEL execution language.

BPMN roughly maps to WSBPEL, but could be mapped to another execution language as well.



**BPMN**  
**Business Process Model**



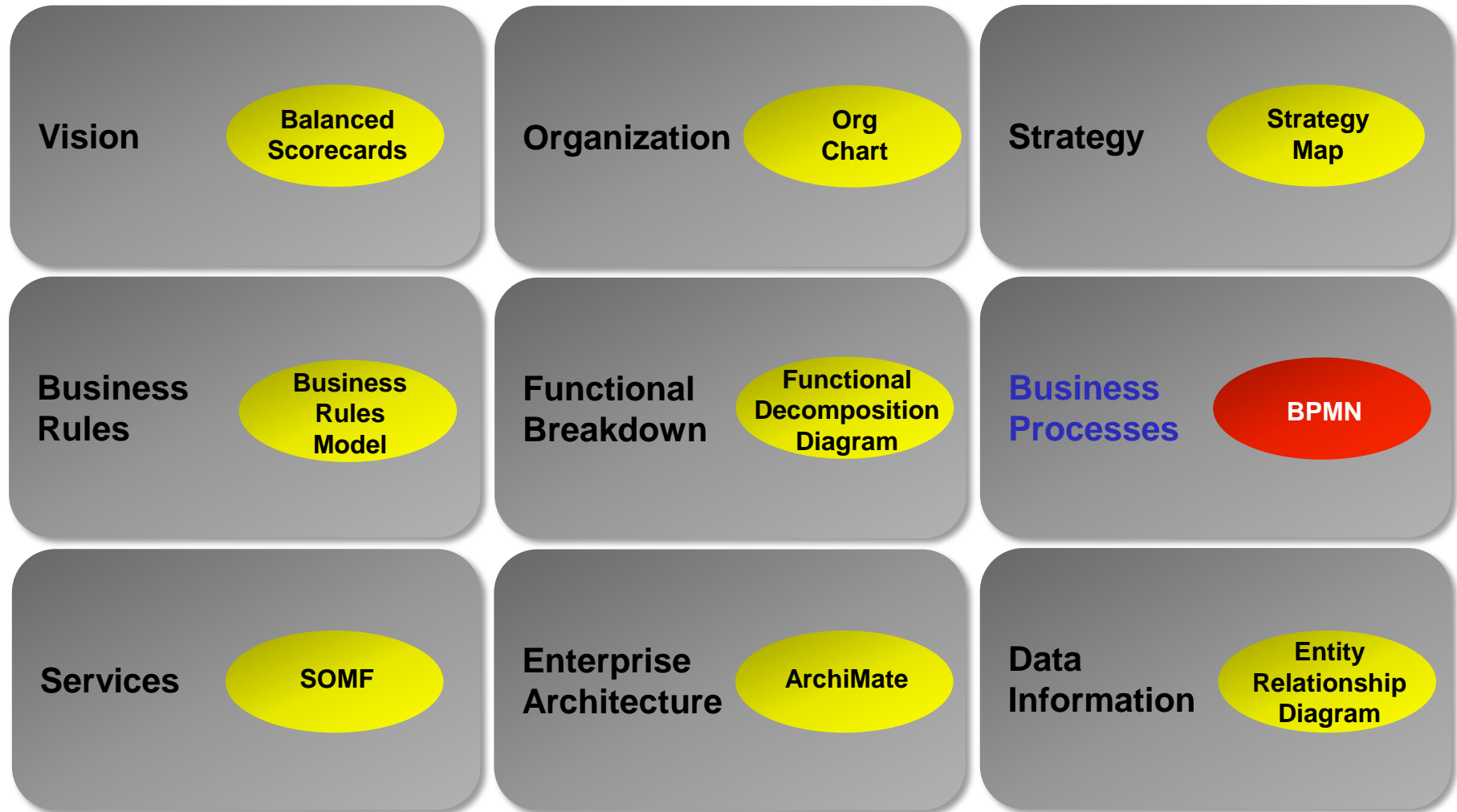
Mapping to

```
...  
<bpel:assign bpmn:label="invoke" name="invoke"  
  bpmn:id="_Oig9ANU5Edyqa7yB__NIQQ">  
  <bpel:copy>  
  
  <bpel:from>$thisStartRequestMsg.body/tns:city</bpel:from>  
  
  <bpel:to>$timeService1GetCityTimeRequestMsg.parameters/TimeService1:city</bpel:to>  
  <bpel:copy>  
  
...
```

**WSBPEL to run in BPEL-runtime**

## 3. BPMN Scope

Different model notations and languages are used for modeling business aspects. BPMN is a notation to model business processes.

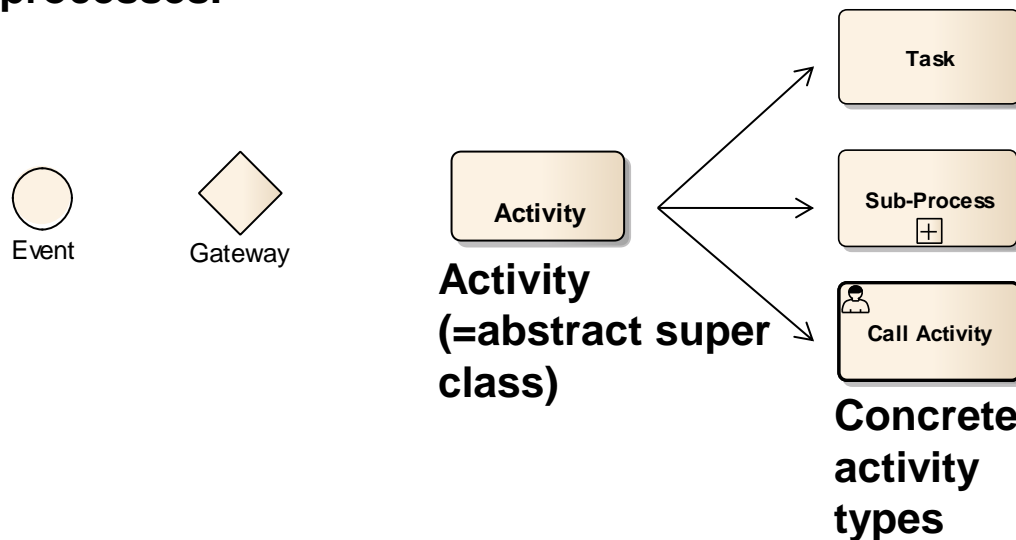


## 4. BPMN Core Concepts (1/5)

### A. Orchestration / processes (1/2):

A **process** describes a **sequence or flow of activities** as part of work to be carried out (=workflow).

Processes contain the BPMN flow elements (events, activities, gateways) and callable processes.



### Process types:

- Private non-executable process (process for documentary purposes only)*
- Private executable process (contains enough detail to be executable)*
- Public process*

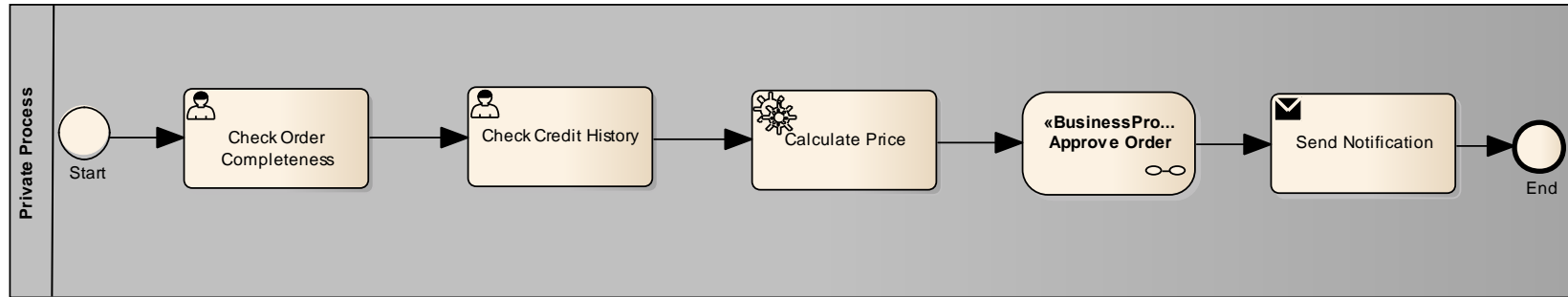


## 4. BPMN Core Concepts (2/5)

### A. Orchestration / processes (2/2):

#### a. Private process:

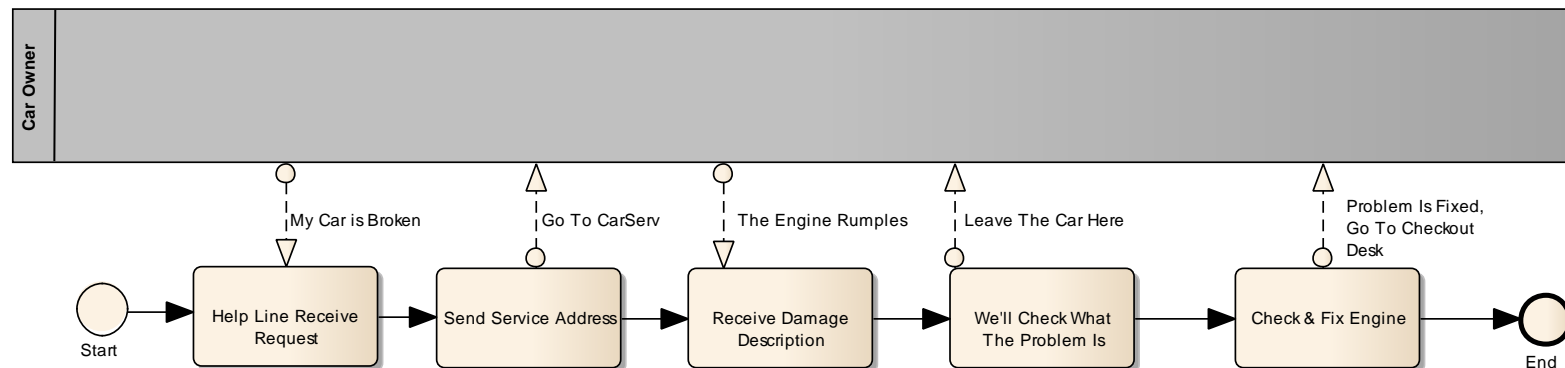
A private process is specific to an organization. There is no interaction with another swimlane (pool = participant).



#### b. Public process:

Public processes show interactions between a private process and another process or participant (pool).

Internals of public process are not shown, only the interactions with another process.



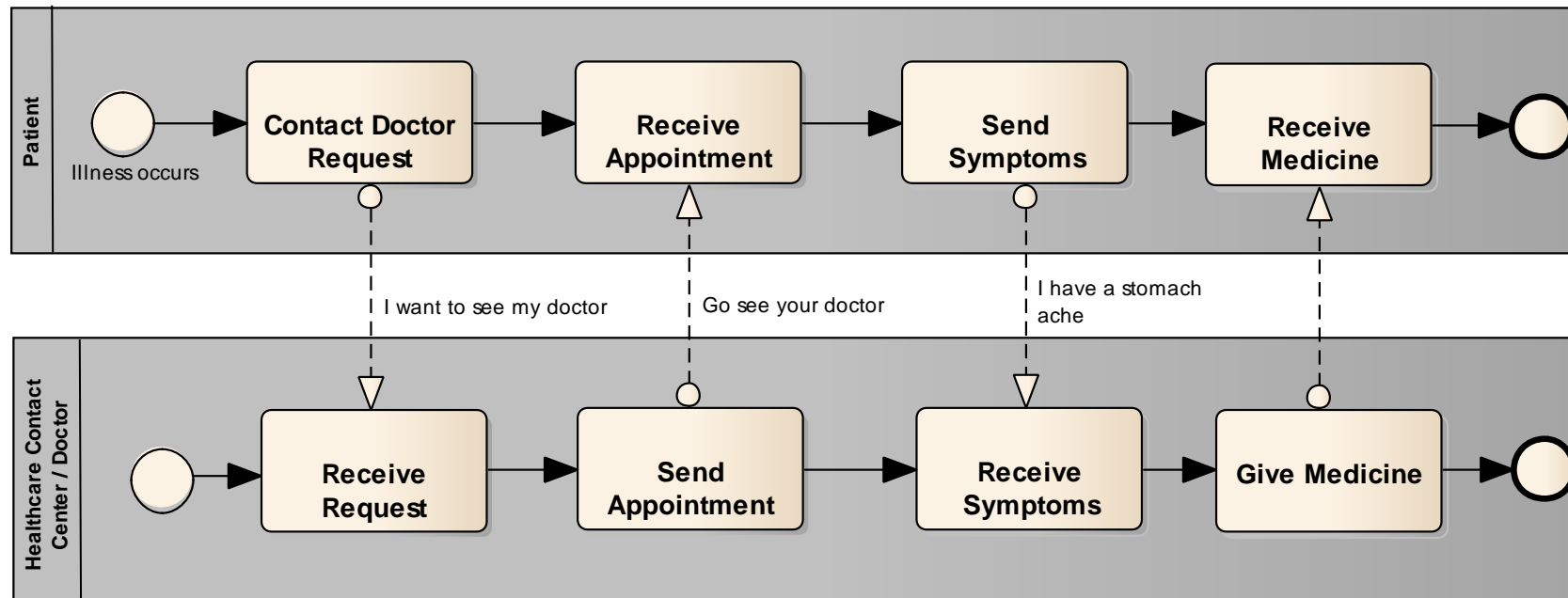
## 4. BPMN Core Concepts (3/5)

### B. Collaboration:

A collaboration shows the **interaction between participants** modeled as swimlanes (pool, lane).

A collaboration diagram typically contains 2 or more pools / lanes.

**Message flows cross the pool boundaries while sequence flows connect activities within pools.**



## 4. BPMN Core Concepts (4/5)

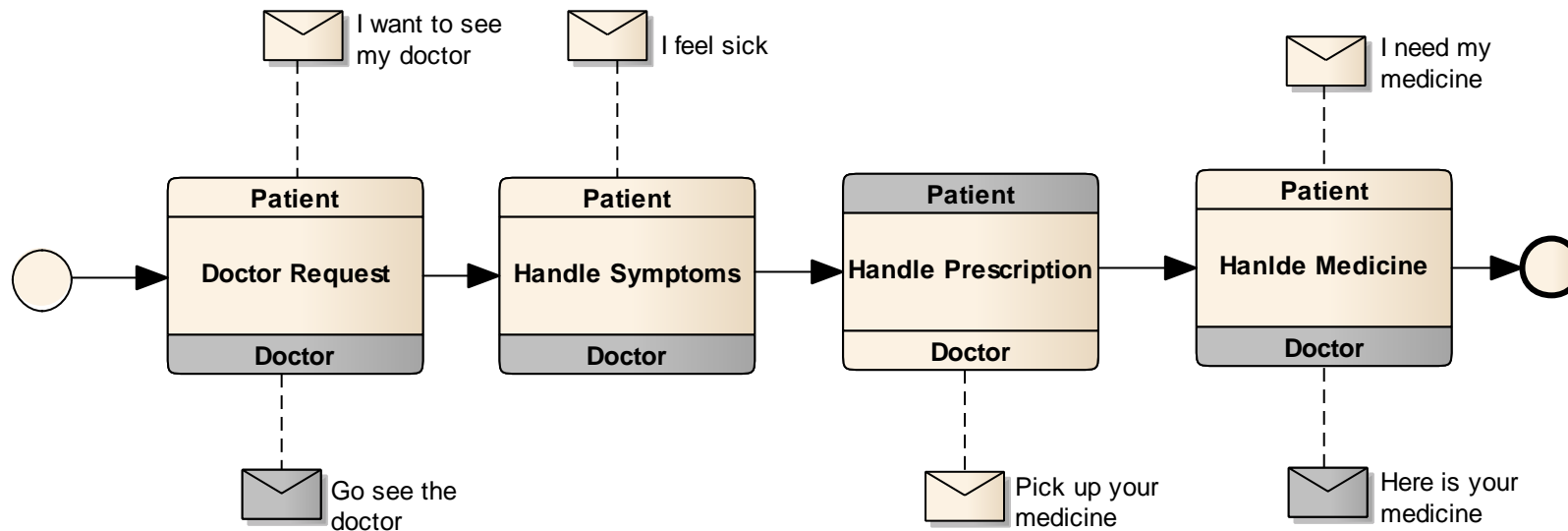
### C. Choreography:

Choreography shows the interactions between participants modeled as pools.

Choreographies are defined outside of pools and exist between pools.

The focus of choreographies is on the exchange of information between participants.

In choreographies, there is no central control, responsible entity or observer.

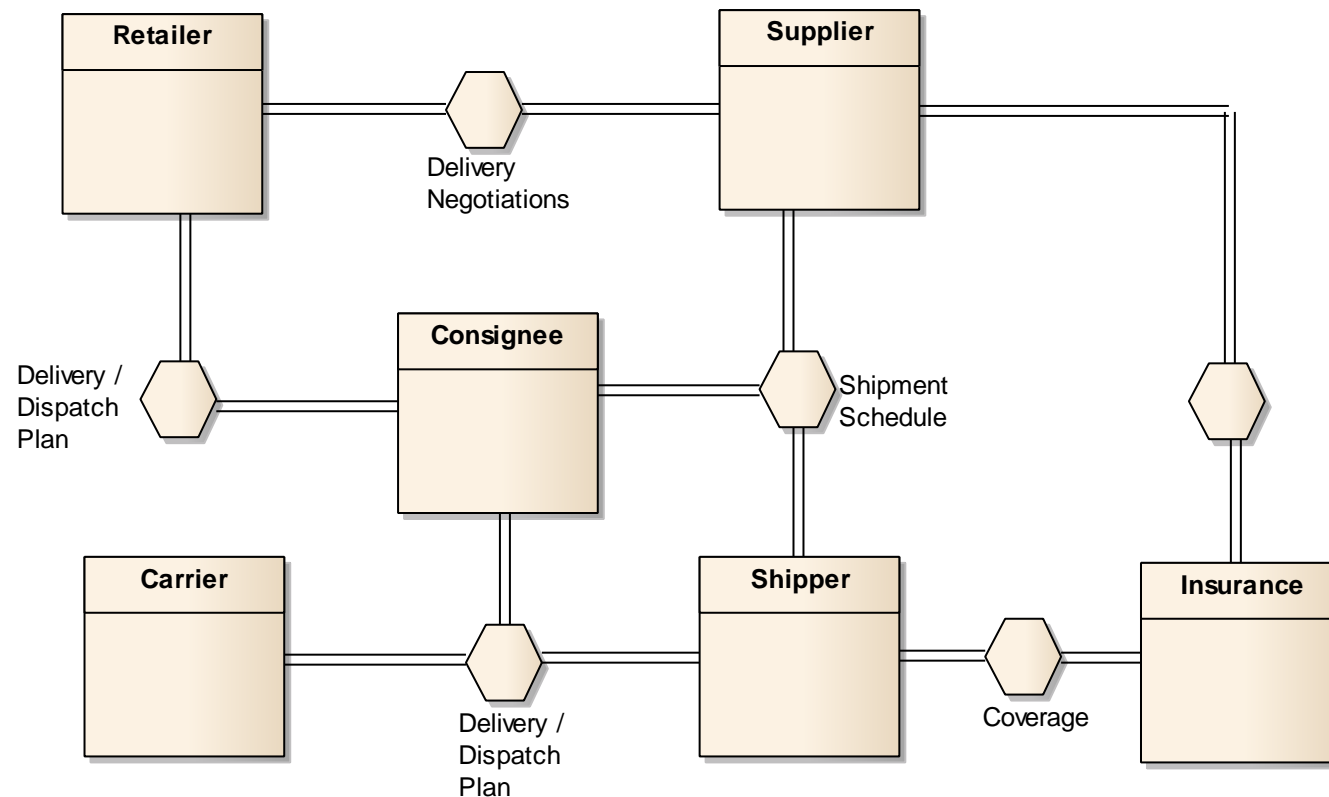


## 4. BPMN Core Concepts (5/5)

### D. Conversation:

Conversation diagrams show the logical message exchanges between participants.

Unlike process, collaboration and choreography diagrams, conversation diagrams show a "birds eye view" of the different conversations (exchange of information) between participants.



## 5. BPMN versus BPEL

BPMN and BPEL share common concepts, but are not tools for the same purpose.

BPMN is mainly a notation that defines how business processes can be modeled graphically.

WS-BPEL defines a machine-processable and –executable format for business processes.

BPMN 1.2 is typically translated to BPEL 2.0 which in turn is executed on a BPMN runtime.

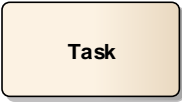
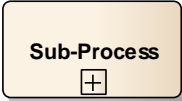
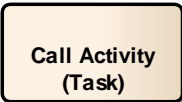
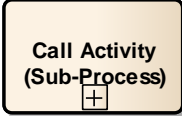
BPMN 2.0 can be directly executed on a BPMN runtime without translation to another format.

| Description Language | Translator    | Execution Language | Runtime                   |
|----------------------|---------------|--------------------|---------------------------|
| BPMN 1.2             | BPMN Mapper   | WS-BPEL (BPEL 2.0) | BPMN runtime like intalio |
| BPMN 2.0             | -             | BPMN 2.0           | BPMN runtime              |
| Java                 | Java compiler | Byte code          | Java Virtual Machine      |
| C / C++              | Compiler      | Machine code       | CPU                       |

## 6. BPMN notation elements (1/10)

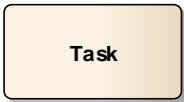

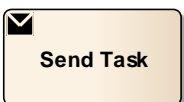
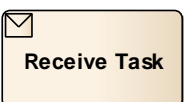
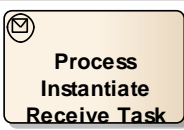
### Activity:

Activities (tasks, sub-processes) are executable elements of a BPMN process.

| Activity type | Symbol  | Description   |
|---------------|---|---|
| Task          |    | <p>Atomic activity within a process that cannot be broken down further.</p> <p>Typically a task is executed by an end-user or application.</p>  |
| Sub-process   |    | <p>Activity with an internal structure containing activities, gateways, events and sequence flows.</p>  |
| Call activity | <br> | <p>Defines a point where a global task or process is called.</p> <p>A call activity differs from a sub-process in that it is a reference to a process while a sub-process is a process itself.</p> <p>This means that a call activity calls a reusable process or task. The called sub-process or task can be called multiple times by different call activities.</p> |

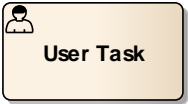
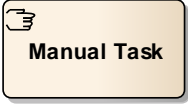
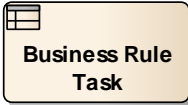
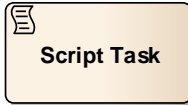
## 6. BPMN notation elements (2/10)

### Task types (1/2):

| Task type                  | Symbol   | Description   |
|----------------------------|--|---|
| Abstract task              |   | Task without any specialization.  |
| Service task               |   | Task that uses some sort of service like a web service.   |
| Send task                  |   | Task for sending a message to an external participant. After the message is sent, the task is completed.                    |
| Receive task               |   | Task that waits for a message to arrive from an external participant. After the message is received, the task is completed. |
| Instantiating receive task |  | Same as receive task, but instantiates a process ("creates process token").   |

## 6. BPMN notation elements (3/10)

### Task types (2/2):


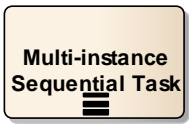

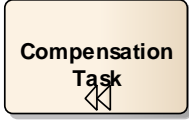


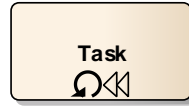
| Task type          | Symbol   | Description  |
|--------------------|--|--|
| User task          |  User Task          | Task executed by a human with the assistance of a software application.  |
| Manual task        |  Manual Task        | Task executed by a human without the assistance of any process execution engine or application.  |
| Business rule task |  Business Rule Task | Provides input to a business rule engine.<br>Receives output from a business rule engine.<br>Business rule tasks connect a process or sub-process to a business rule engine. |
| Script task        |  Script Task       | A script task is executed by a business process engine.<br>When the script is finished, the task is finished as well.  |



## 6. BPMN notation elements (4/10)

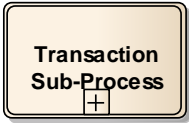

### Task markers:

Additional markers add more semantics to tasks.

| Task marker                 | Symbol   | Description  |
|-----------------------------|--|--|
| Loop                        |   | A looping task repeats its action as long as a loop flag is set.   |
| Multi-instance sequential   |   | The task is executed in multiple instances in sequential order.  |
| Multi-instance parallel     |   | Multiple instances of the task are executed in parallel.   |
| Compensation                |    | Compensation is used in case of errors (exceptions) to undo steps that are already executed (call of compensation handler for a "rollback"). |
| Allowed marker combinations |  Task<br>Multi-instance sequential & compensation |  Task<br>Multi-instance parallel & compensation         |
|                             |  Task<br>Loop & compensation                      |  |

## 6. BPMN notation elements (5/10)

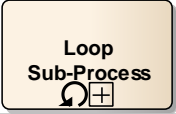
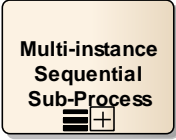


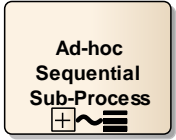
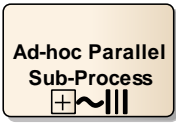
### Sub-process types:

| Sub-process type        | Symbol  | Description  |
|-------------------------|---|--|
| Transaction sub-process |  | <p>A transactional sub-process executes all internal activities either successfully or none (exception case).</p> <p>Typically, a transaction sub-process is combined with a cancel event to call a transaction cancel (rollback) handler.</p>   |
| Event sub-process       |  | <p>Execution of an event sub-process is triggered by an event. The execution is independent of the parent process flow, thus there are no incoming and outgoing sequence flows. However, an event sub-process is only executed when the parent process (or sub-process) is active.</p> |

## 6. BPMN notation elements (6/10)




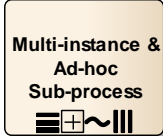

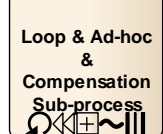
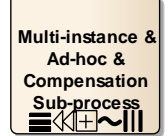
### Sub-process markers (1/2):

Sub-processes may have the same markers as tasks. Additionally, the ad-hoc marker may be used on sub-processes to express a less strict execution ordering.

| Sub-process marker        | Symbol  | Description   |
|---------------------------|---|---|
| Loop                      |    | A looping sub-process repeats its internal activities as long as a loop flag is set.  |
| Multi-instance sequential |    | The sub-process is executed in multiple instances in sequential order.  |
| Multi-instance parallel   |    | Multiple instances of the sub-process are executed in parallel.   |
| Compensation              |   | Compensation is used in case of errors (exceptions) to undo steps that are already executed (call of compensation handler to "rollback").   |
| Ad-hoc sequential         |  | Ad-hoc sub-processes have a less strict temporal ordering of activities. Internal activities may or may not have sequence flows. Ad-hoc sequential means that contained activities are executed sequentially. |
| Ad-hoc parallel           |  | In a parallel ad-hoc sub-process, contained activities are executed in parallel.  |

## 6. BPMN notation elements (7/10)

### Sub-process markers (2/2):

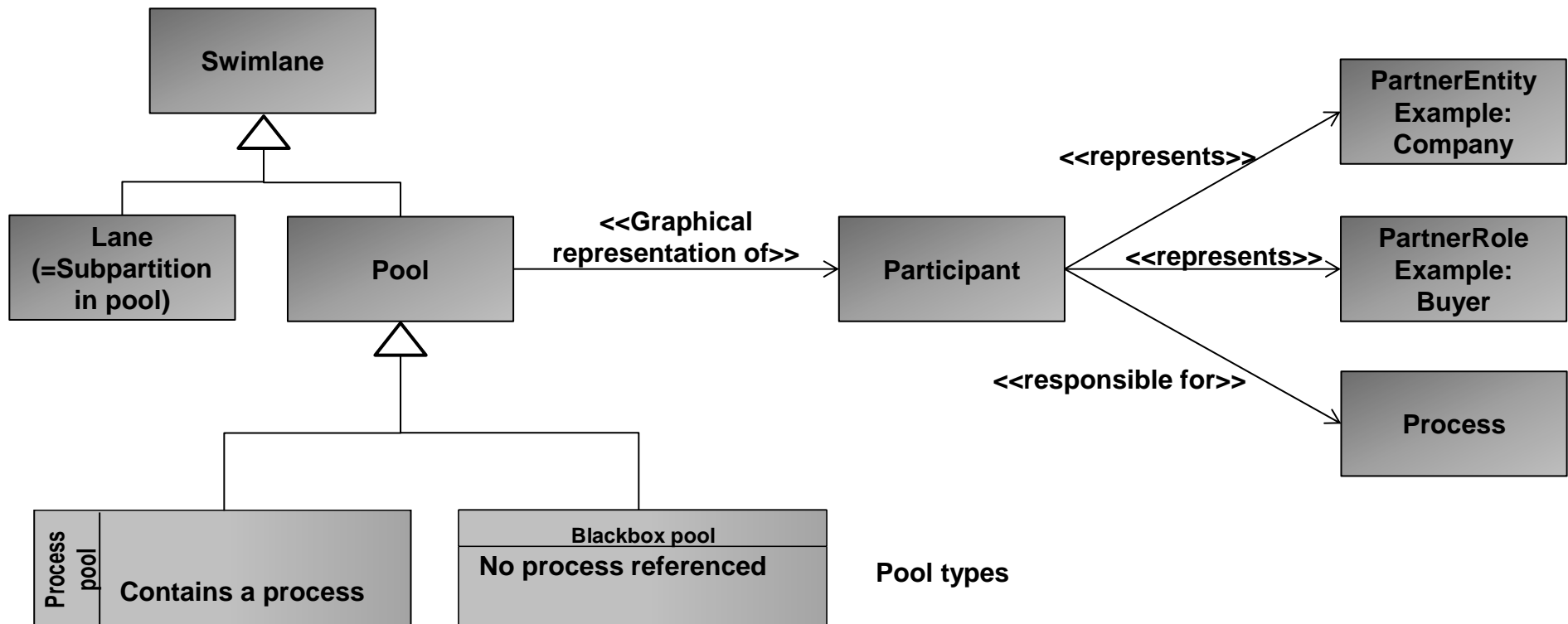
| Task marker                 | Symbol & Description   |  |   |  |
|-----------------------------|--|--|---|--|
| Allowed marker combinations |  <p>Loop &amp; Compensation Sub-Process</p>                         | <p><b>Loop &amp; compensation</b></p>  |  <p>Loop &amp; Ad-hoc Sub-Process</p>                    | <p><b>Loop &amp; ad-hoc (parallel and sequential)</b></p>                    |
|                             |  <p>Multi-instance &amp; Compensation Sub-process</p>               | <p><b>Multi-instance &amp; compensation (parallel and sequential)</b></p>              |  <p>Multi-instance &amp; Ad-hoc Sub-process</p>          | <p><b>Multi-instance &amp; ad-hoc (parallel and sequential)</b></p>          |
|                             |  <p>Ad-hoc &amp; Compensation Sub-process</p>                       | <p><b>Ad-hoc &amp; compensation (parallel and sequential)</b></p>                      |  <p>Loop &amp; Ad-hoc &amp; Compensation Sub-process</p> | <p><b>Loop &amp; compensation &amp; ad-hoc (parallel and sequential)</b></p> |
|                             |  <p>Multi-instance &amp; Ad-hoc &amp; Compensation Sub-process</p> | <p><b>Multi-instance &amp; compensation &amp; ad-hoc (parallel and sequential)</b></p> |   |  |

## 6. BPMN notation elements (8/10)

### Swimlanes (1/2):


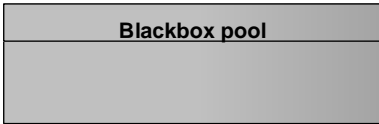

Swimlanes are either pools (white-box or black-box) or lanes (=partitions in pools). Pools represent participants which are partner entities like a supplier. Participants are often responsible for a process execution in a pool.

### Simplified meta-model:

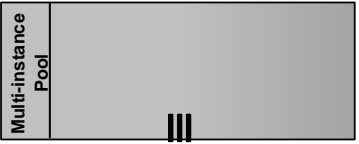


## 6. BPMN notation elements (9/10)

### Swimlanes (2/2):

| Swimlane type              | Symbol  | Description   |
|----------------------------|---|---|
| Process ("White-box") pool |  | Pool that contains a process.   |
| Black-box pool             |  | Pool that does not contain a process.   |
| Lane                       |  | Lanes are sub-partitions in a pool or process. Lanes can be nested. Lanes do not have semantics in BPMN, i.e. lanes are a simple grouping and partitioning concept. |







### Swimlanes markers:

| Swimlane marker | Symbol  | Description  |
|-----------------|---|--|
| Multi-instance  |  | Multi-instance pools represent multi-instance participants. Example: Multiple suppliers. |

## 6. BPMN notation elements (10/10)

### Gateways:

Gateways are used to produce and consume process tokens (process instances).

| Gateway type      | Symbol   | Description   |
|-------------------|--|---|
| Exclusive gateway |  or  | Exclusive gateways are like decisions. The process flow takes either of the outgoing directions (only 1 or none).   |
| Inclusive gateway |   | The inclusive gateway evaluates all outgoing conditions and creates a process token for each condition that evaluates to true.  |
| Parallel gateway  |   | Parallel gateways join and fork flows. Thus parallel gateways are used to create (fork) and synchronize (join) parallel flows.<br>No condition checking is involved with parallel gateways.   |
| Complex gateway   |    | Complex gateways are used to model complex synchronization behavior.<br>Example: 3 out of 5 incoming flow tokens must be present to create an outgoing process token (depending on the conditions of the outgoing flows).<br>Outgoing and incoming token semantics are the same as in inclusive gateways. |
| Event gateway     |   | Event gateways represent branching points based on events rather than conditions.<br>Example: Reception of different messages is dispatched into different process paths to handle the messages.  |

## 7. BPMN 2.0 versus 1.2

BPMN 2.0 brought a number of changes :

- Formalization and clarification of the execution semantics of all BPMN elements
- Extensibility mechanism for graphical elements
- Refinement of event composition and correlation
- Definition of choreography model (choreography, choreography diagram)
- Redefinition and clarification of various BPMN elements (see table below)

| <b>BPMN 1.2 Feature</b>   | <b>BPMN 2.0 Feature</b>  |
|---|--|
| Reusable sub-process  | Call activity  |
| Embedded sub-process  | Sub-process  |
| Abstract process  | Public process   |
| Directional association to show how data objects are inputs and outputs to activities | Data association connector to show inputs and outputs  |
| Message is a BPMN element   | Message is only a graphical decorator  |
| None task   | Abstract task  |
| Directional associations to show data flows to and from activities                    | Data association connector to show inputs and outputs of activities                            |
| Performer role for describing people in activities                                    | In addition to Performer role, HumanPerformer role allows to identify humans performing a task |
| Intermediate events possible without incoming sequence flows                          | Intermediate events must have an incoming sequence flow  |