Lab for the course on Process and Service Modeling and Analysis

LAB-05

Turning BPMN Process Models Executable

Lecturer: Andrea MARRELLA

Objectives of this lecture

- Understanding Business Process Automation.
- A methodology to turn business-oriented process models into executable ones.
  - Cf. also chapter 9 of the book:
    - M.Dumas, M.La Rosa, J.Mendling, H.A. Reijers ***Fundamentals of Business Process Management***. Springer (2013)
- Classroom exercises.

Process Automation

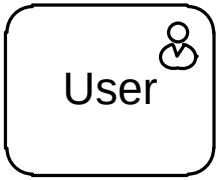
- **Process automation** refers to the intent to ***automate any conceivable part of procedural work that is contained within a business process***, from *simple* operations that are part of a *single* process activity up to the automated coordination of *entire*, complex processes.

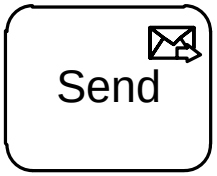
- In this context, we will refer to an ***automated business process*** (also known as ***workflow***), as a process that is automated *in whole* or *in part* by a **Business Process Management System** (BPMS).
  - The BPMS passes information from one participant to another for action, according to the temporal and logical dependencies set in the underlying process model.

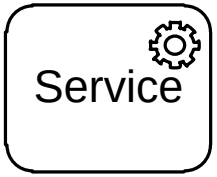
Turning Process Models Executable

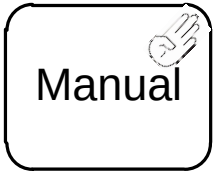
- Traditional ***business-oriented process models*** are not necessarily precise and may thus contain ambiguities.
- Conversely, ***executable process models*** must be precise specifications in order to be interpreted by a BPMS.
- We show a **six-step method** to incrementally transform a business-oriented process model into an executable one:
  1. Specify task markers
  2. Identify the automation boundaries
  3. Review manual tasks
  4. Complete the process model
  5. Bring the process model to an adequate granularity level
  6. Specify execution properties
- Through these steps, the business-oriented model will become less abstract and **more IT-oriented**.
- These steps should be carried out on a process model that is both **correct** in terms of ***structure*** and ***behavior***.

1. Specify Task Markers









- There are 6 specialized types of tasks (with different markers):
- **User** : A task where a human performer carries out the task with the assistance of a software application (e.g., the worklist handler of a BPMS).
- **Receive** : Waits for a message to arrive from an external participant (e.g., get shipping address). Once received, the task is complete.
- **Send** : Dispatches a message to an external participant (e.g., request raw material).
- **Service** : Links to some sort of service, which could be a web service or an external application (e.g., check stock availability).
- **Script** : Performs some code (the script) internally to the BPMS (e.g., selecting the best quote from a list of suppliers).
- **Manual** : A non-automated task that a human performer undertakes outside of the control of the BPMS (e.g., retrieve product from warehouse).

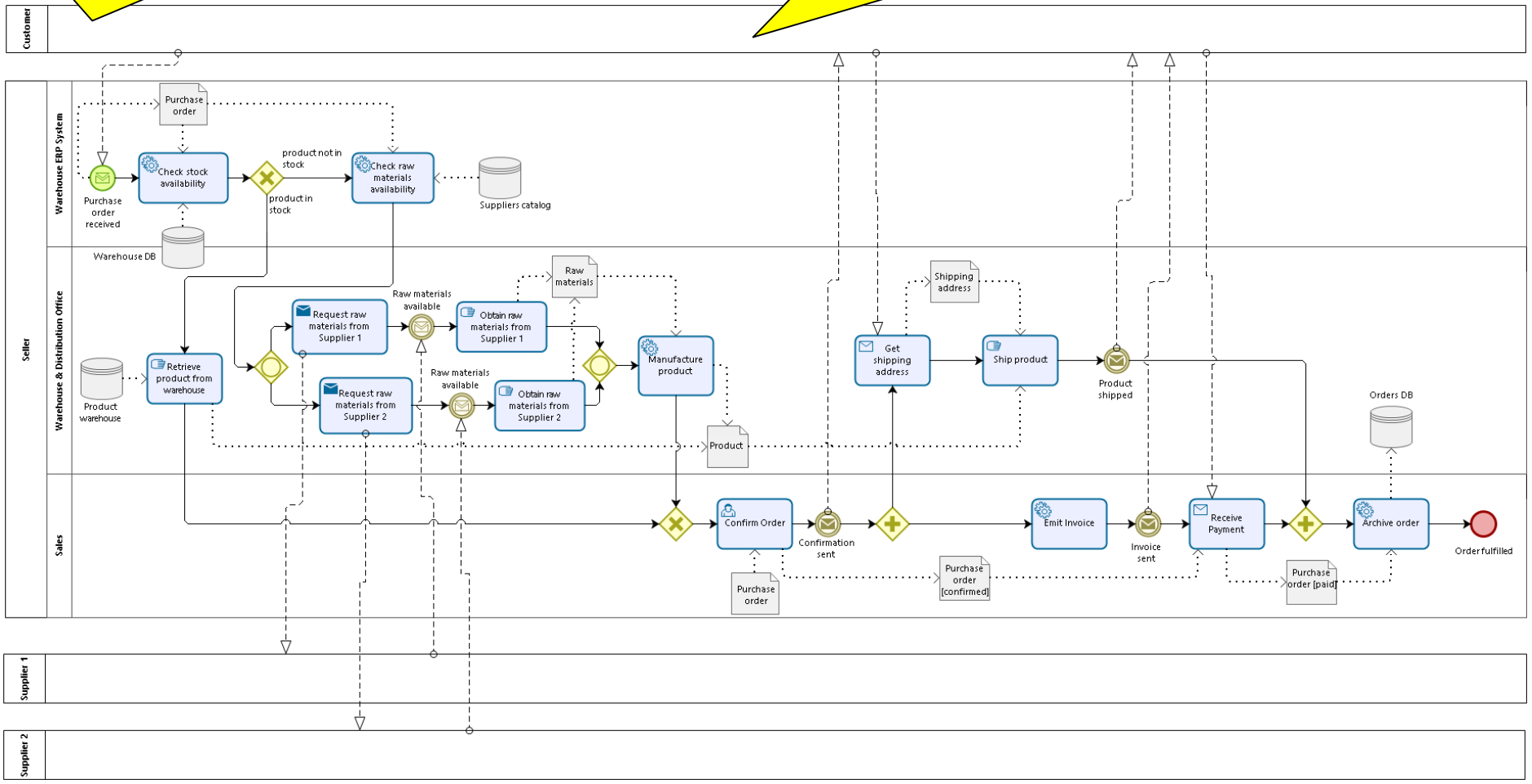
Running Example

- The order fulfillment process is carried out by a seller's organization which includes two departments: the sales department and the warehouse & distribution department.
- The purchase order received by warehouse & distribution is checked against the stock. This operation is carried out automatically by the ERP (Enterprise Resource Planning) system of warehouse & distribution, which queries the warehouse database.
- If the product is in stock, it is retrieved from the warehouse before sales confirm the order. Next sales emit an invoice and wait for the payment, while the product is shipped from within warehouse & distribution. The process completes with the order archival in the sales department.
- If the product is not in stock, the ERP system within warehouse & distribution checks the raw materials availability by accessing the catalog of two suppliers. Once the raw materials have been obtained the warehouse & distribution department takes care of manufacturing the product. The process completes with the purchase order being confirmed and archived by the sales department.

Solution

Four pools representing the Seller, the Customer and two Suppliers.

Three lanes are required: one for the warehouse & distribution department office, the second for the warehouse ERP system, the third for the sales department.



2. Identify the Automation Boundaries

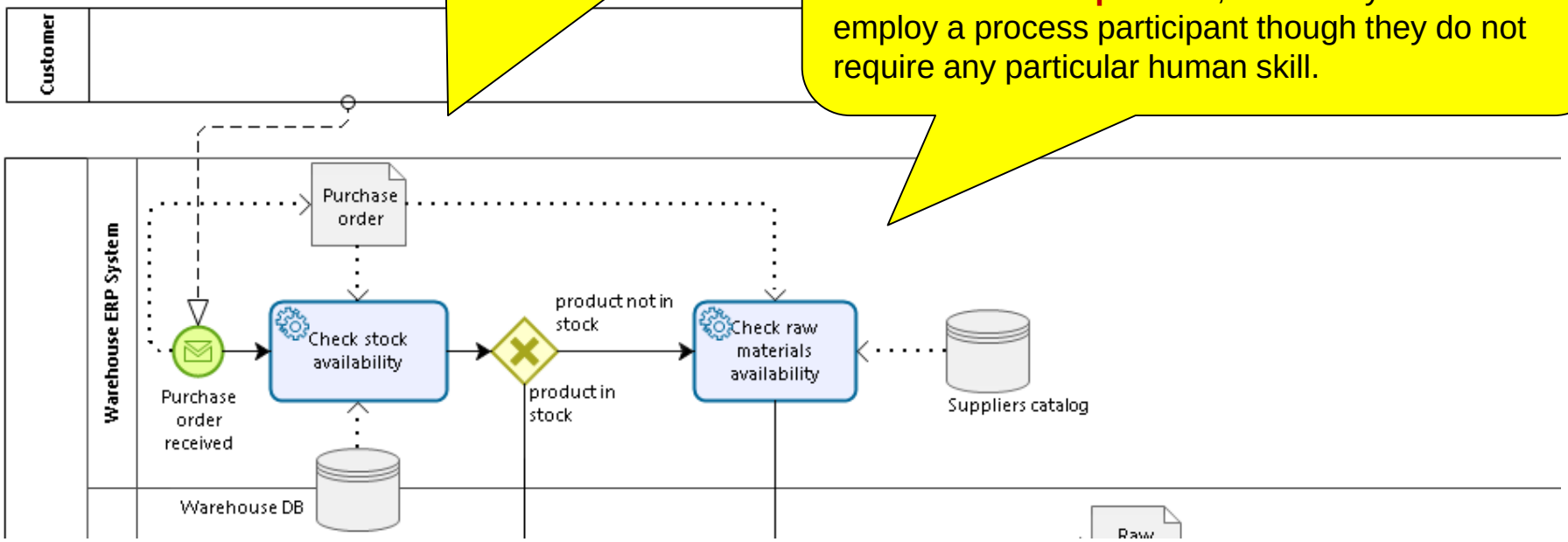
- We need to identify what parts of the process can be coordinated by the BPMS, and what parts cannot.
- In a process there are ***automated***, ***manual*** and ***user*** tasks.
  - **Automated tasks** are performed by the BPMS itself or by an external service. Script tasks, Receive tasks, Send tasks and Service Tasks are considered as automated tasks.
  - **Manual tasks** are performed by process participants without the aid of any software.
  - A **user task** sits in-between an automated and a manual task. It is a task performed by a participant with the assistance of the worklist handler of the BPMS.
- In this activity, we need to identify each type of task.

Notice that **markers apply to tasks only**. They cannot be used on sub-processes since a sub-process may contain tasks of different types.

Identify each type of task

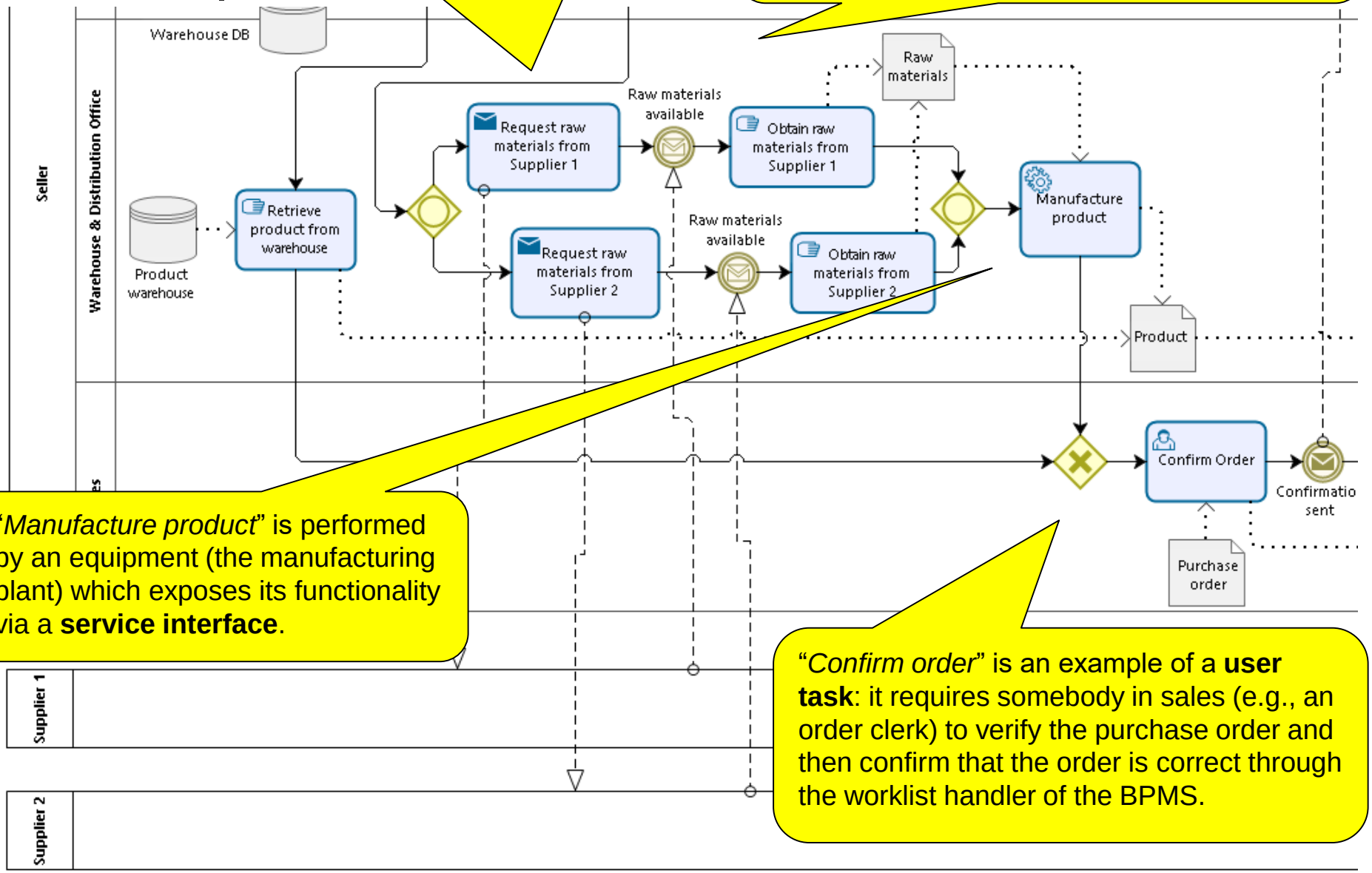
ERP systems provide modules to manage inventories which **automatically check the stock** levels of a product against a warehouse database.

These activities are highly repetitive as they are performed for each purchase order received. **Performing them manually** would be very **inefficient** and **expensive**, since they would employ a process participant though they do not require any particular human skill.



Some automated tasks are devoted to **sending** and **receiveing** messages. They can be implemented via **automatic e-mail exchange**.

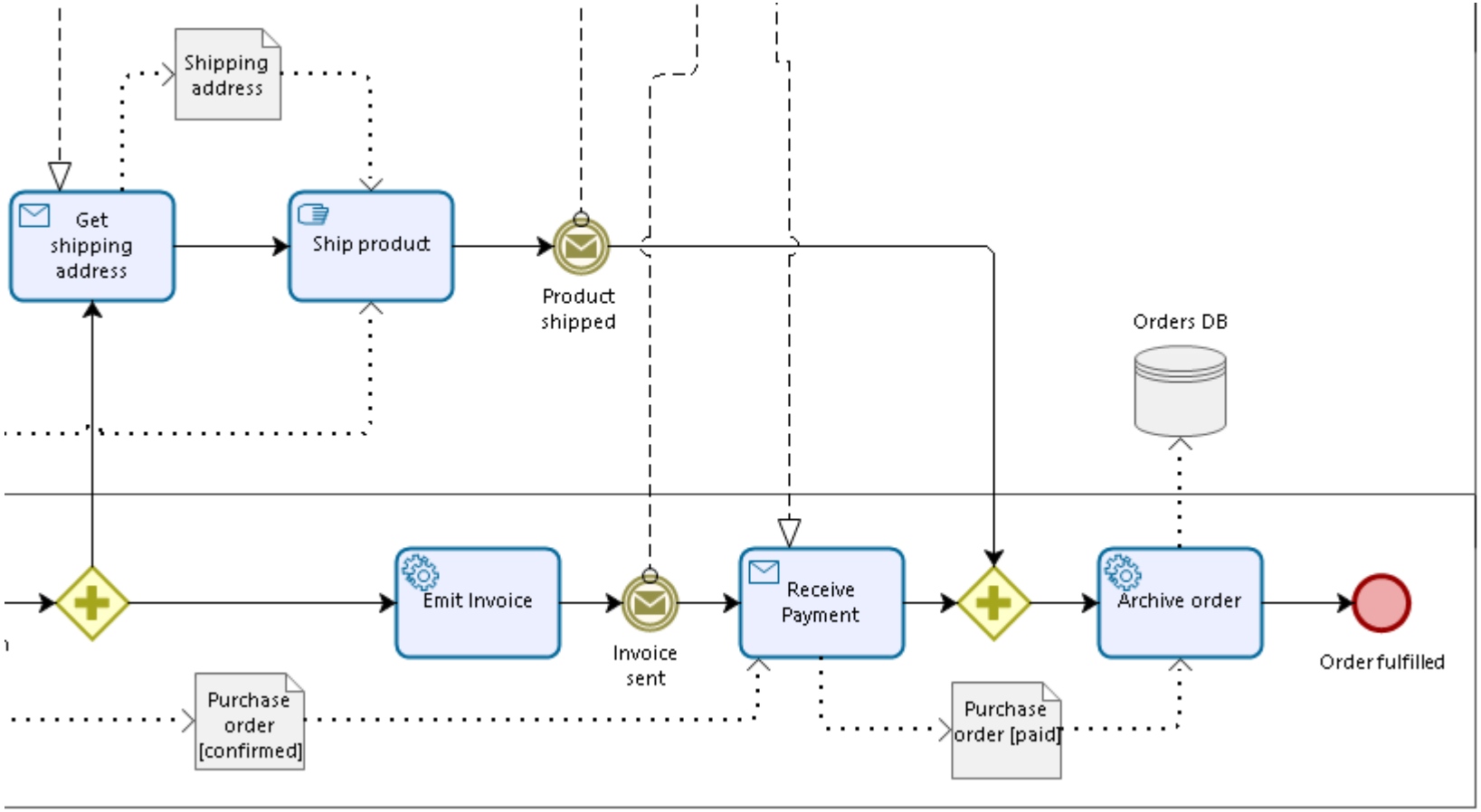
Other tasks are **manual**. For example, "*Retrieve product from warehouse*" requires a warehouse worker to physically pick up the product from the shelf for shipping.



"*Manufacture product*" is performed by an equipment (the manufacturing plant) which exposes its functionality via a **service interface**.

"*Confirm order*" is an example of a **user task**: it requires somebody in sales (e.g., an order clerk) to verify the purchase order and then confirm that the order is correct through the worklist handler of the BPMS.

Identify each type of task

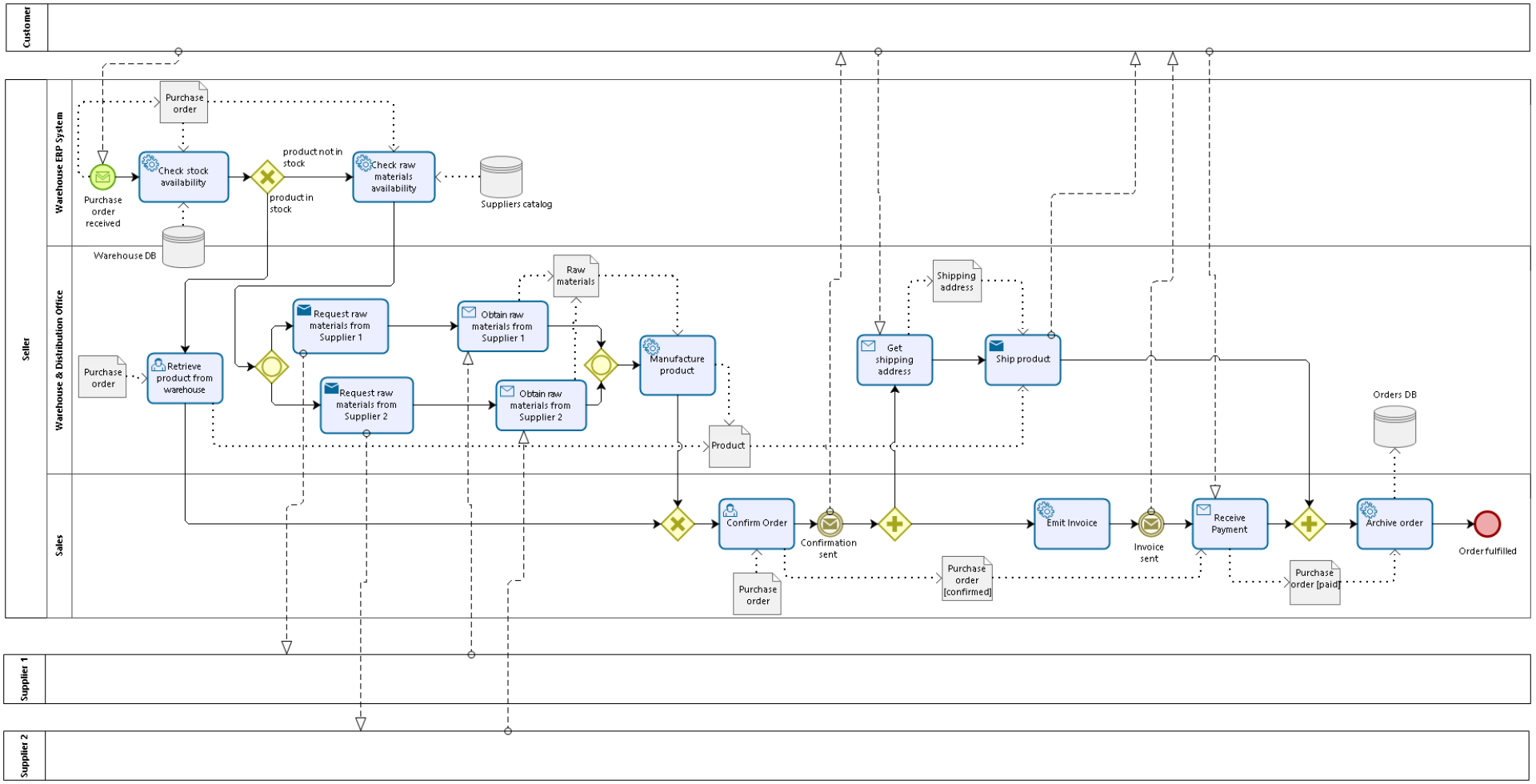


3. Review Manual Tasks

- The difference between automated, manual and user tasks is relevant: automated and user tasks can easily be coordinated by a BPMS, while **manual tasks cannot**.
- Once we have identified the type of each task, we need to check whether we can link the manual tasks to the BPMS

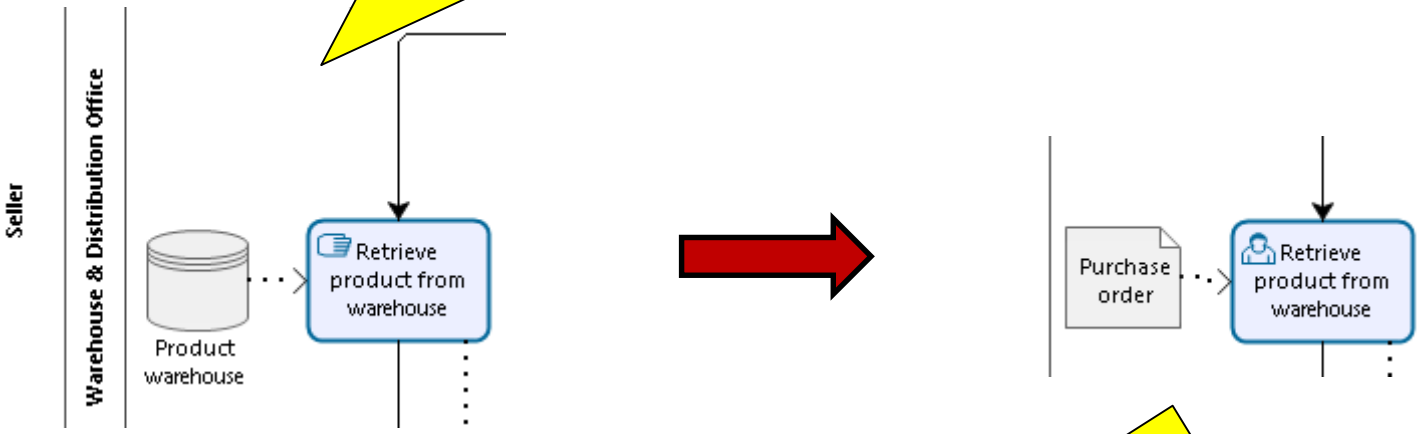
- In the presence of a manual task we have two options:
  - we **isolate the task** and focus on the automation of the process before and after it;
  - we find a way for the BPMS to be notified when the manual task has started or completed, i.e., we **implement it as a user task** or via an **automated task**.

Order fulfillment process without manual tasks



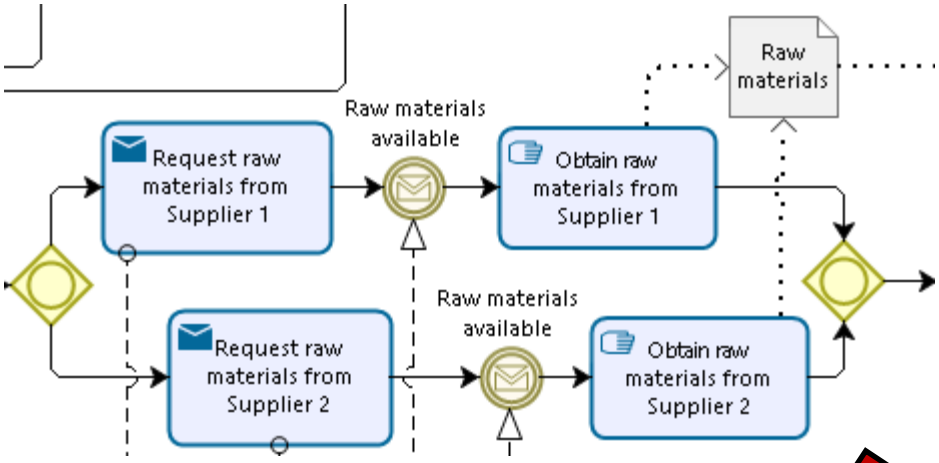
Review Manual Tasks

If the participant involved in the manual task can notify the BPMS of the task completion using the worklist handler of the BPMS, the manual task can be turned into a **user task**.



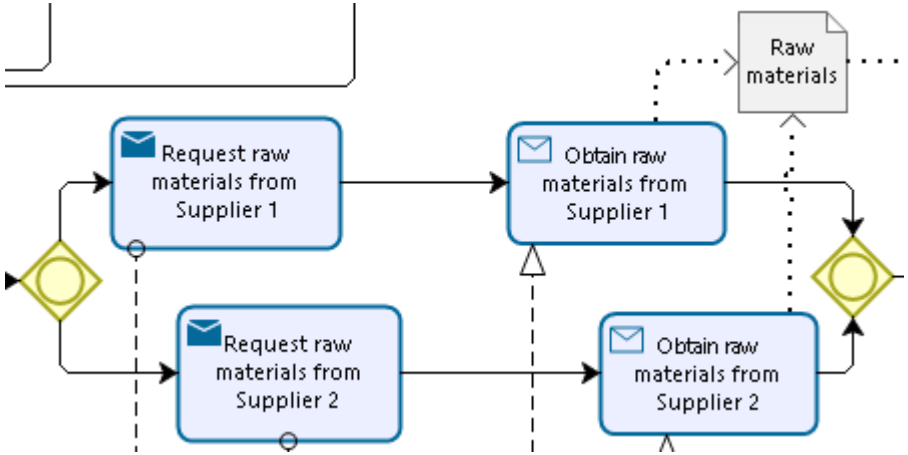
For example, the warehouse worker performing task "*Retrieve product from warehouse*" could **check-out** a work-item of this task from her worklist to indicate that she is about to perform the job, **manually retrieve** the product from the shelf, and then **check-in** the work-item to notify the BPMS engine that the job has been completed.

Review Manual Tasks

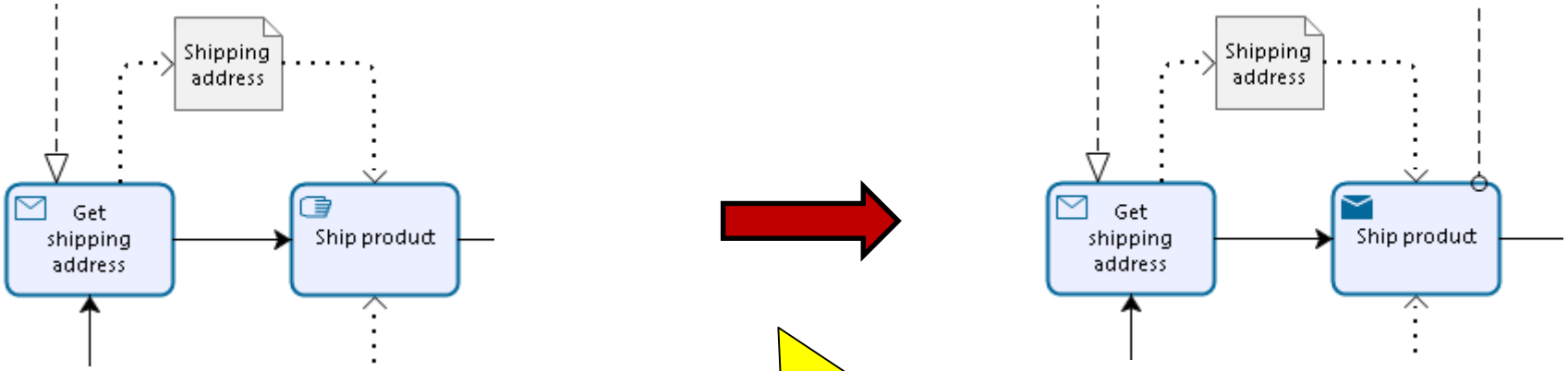


The warehouse worker could use a **barcode scanner** to scan the barcode of the raw materials being obtained. The scanner would be connected to the BPMS so scanning the barcode would automatically signal the completion of "*Obtain raw materials from Supplier 1(2)*".

The manual task can be implemented as a **receive task** awaiting the notification from the scanner. In this case, the BPMS will only be aware of the work item's completion



Review Manual Tasks



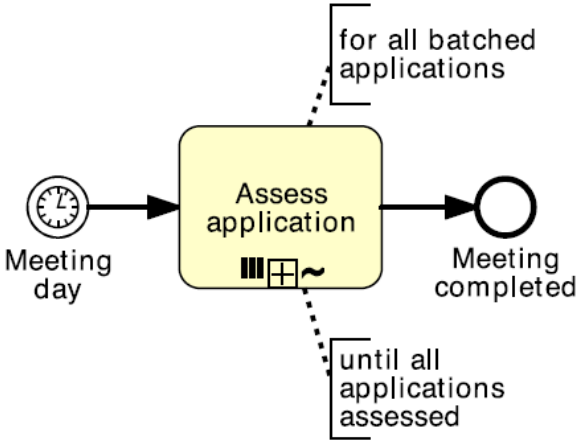
Since each manual task of our example can be linked with a BPMS, **this process can be automated as a whole**.

Isolating manual tasks

- There are cases in which **it is not convenient** to link manual tasks to a BPMS.
- For example, a *university admission process*, is usually organized in three steps:
  1. The admission procedures are collected via a BPMS.
  2. Once all the applications have been collected, the committee will meet and examine all of them at once.
  3. Eventually, the committee will draw a list of accepted candidates, transfer it to the admissions office, and a clerk at the admissions office will update the various student records.
- In this example **we cannot automate the whole process**.
- The tasks required for assessing applications **cannot be automated** because they involve various human participants who interact on an ad-hoc basis and it would not be convenient to synchronize them with the BPMS.

Isolating manual tasks

- We need to isolate activity "*Assess application*", an ad-hoc activity containing various manual tasks, and **automate the process before and after this activity**.
  - An option is to **split the model into three fragments** and only automate the first and the third fragment.



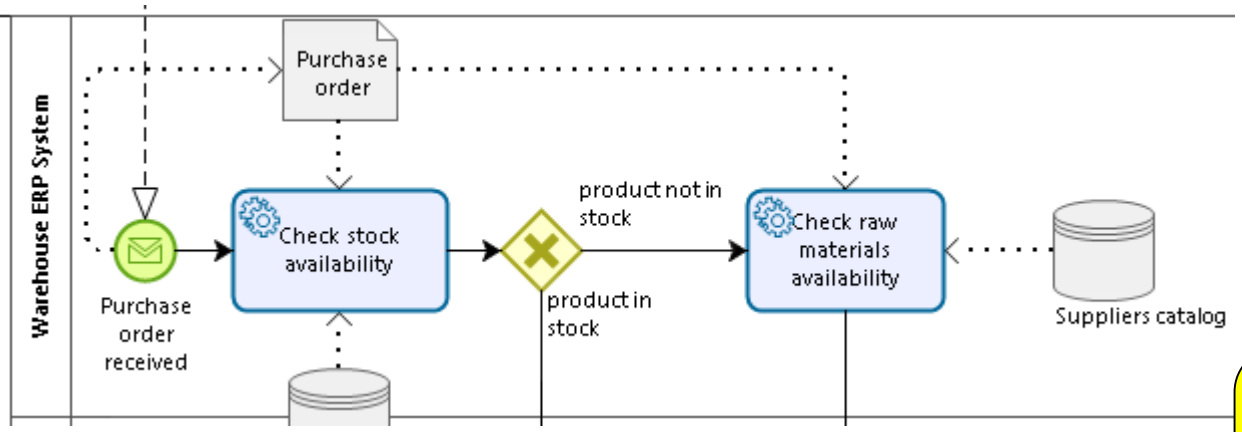
Tolerance of BPMSs to non-executable elements

- Some BPMSs are **tolerant** to the presence of manual tasks and ad-hoc activities in executable models, and **will discard them at deployment time**.
- There are other modeling elements, besides manual tasks, that are relevant at a conceptual level but **cannot be interpreted** by a BPMS.
  - For example, ***physical data objects*** and ***data stores***, ***messages bearing physical objects*** and ***text annotations***.
  - **Pools** and **lanes** are also used at a **conceptual level only**. In fact, they are used to capture *coarse-grained resource assignments*.
  - When it comes to execution, **we need to define resource assignments for each task.**
- Some BPMSs tolerate the presence of non-executable elements.
  - If this is the case, it is suggested to leave these elements in, as they will guide us in the specification of some execution properties.

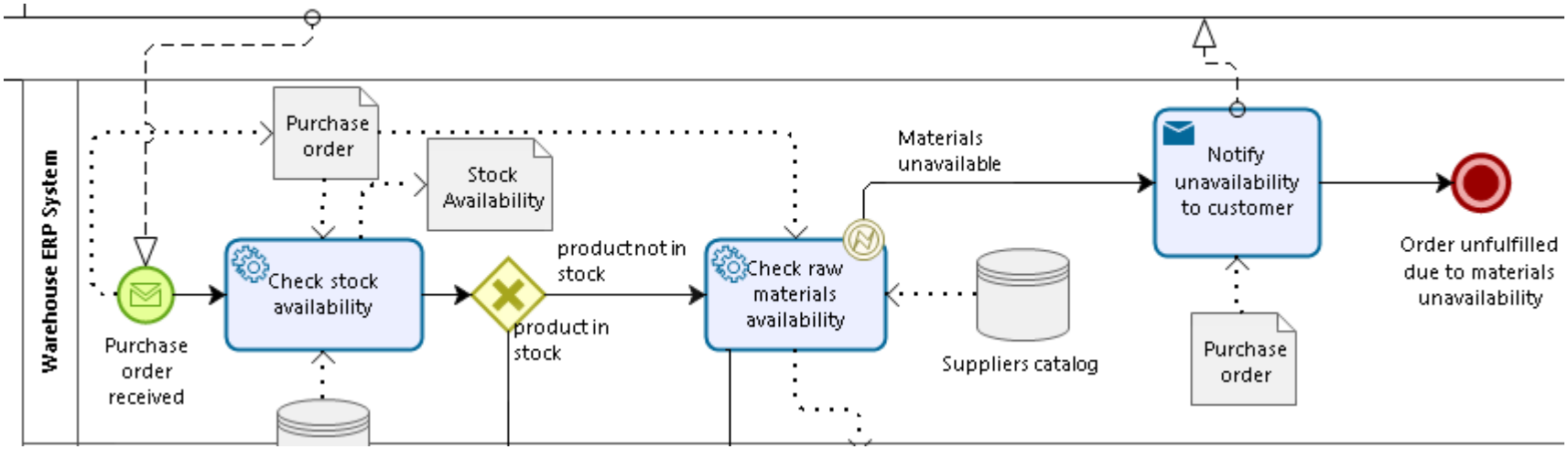
4. Complete the process model

- Often business-oriented process models **neglect certain information** because modelers deem it is not relevant for the specific modeling purpose.
- A typical example is when the process model **focuses** on the "***happy path scenario***" and **ignore all negative situations** (for example, ***exceptions***) that may arise during the execution of the process.
  - We need to make sure that **all exceptions are handled** using appropriate exception handlers.
- In this step, we also need to specify all ***electronic data objects*** that are required as input and output by the tasks of our process.
  - The principle is that every data object needed by the BPMS engine to **pass control between activities** and to **take decisions** must be modeled.

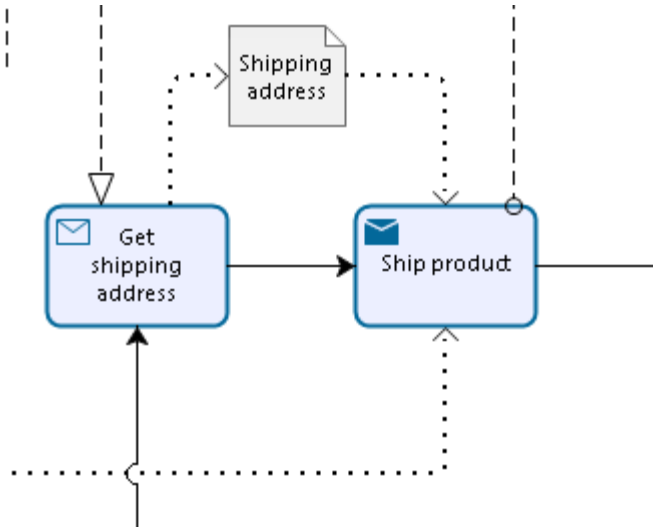
Inserting the exception handlers

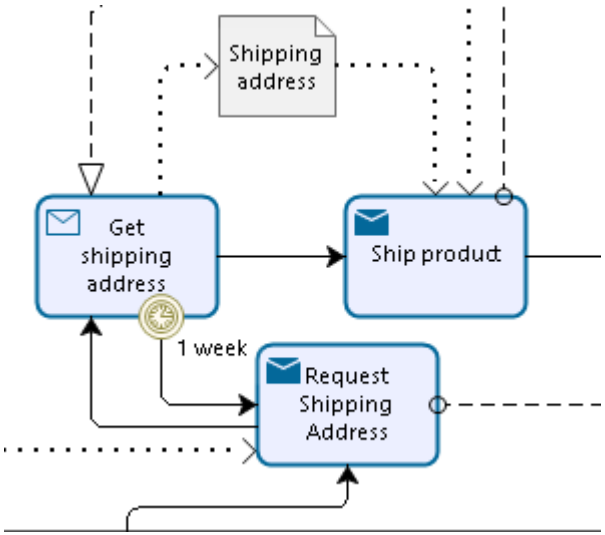


If the product is not available and can not be manufactured by the seller, the **process terminates** and the customer is notified of the **cancellation of the order**.

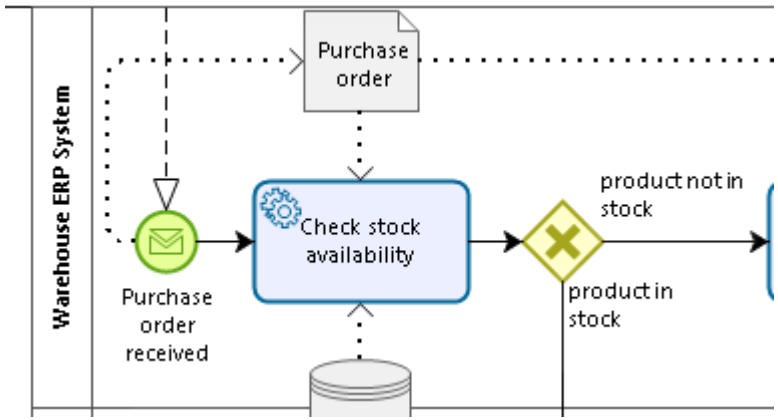


Inserting the exception handlers

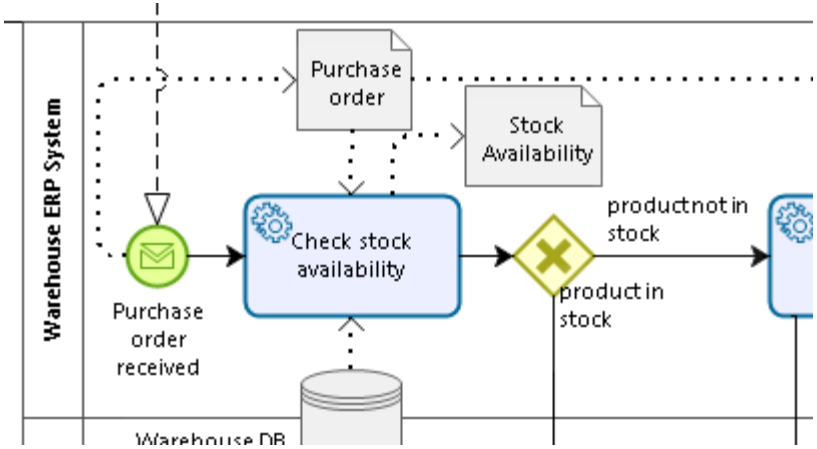




Inserting all required eletronic data objects

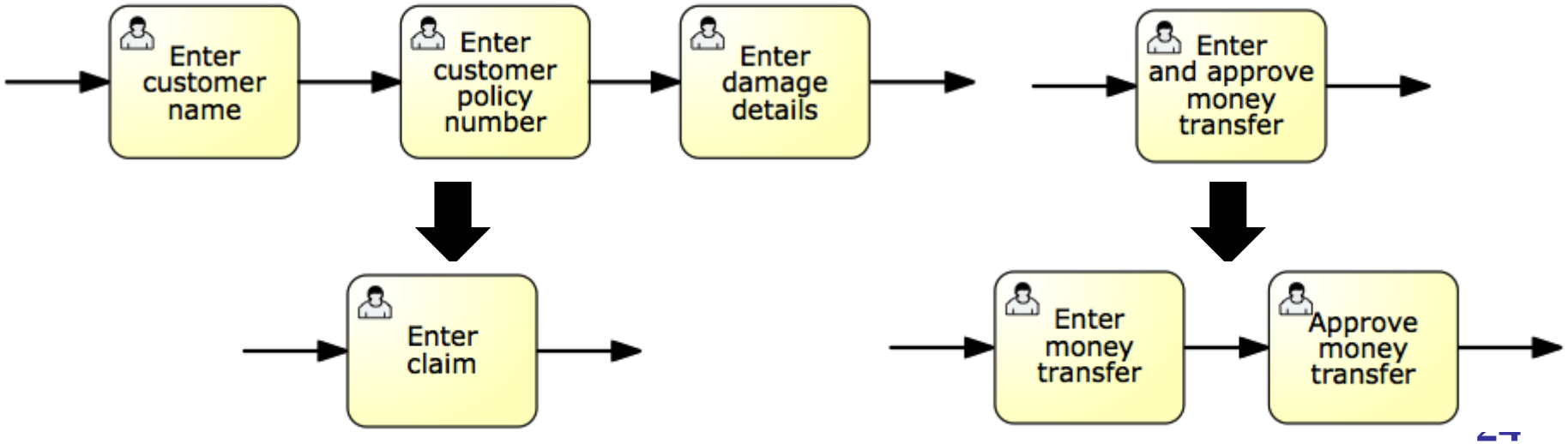


The task "*Check stock availability*" uses the "*Purchase order*" data object as input but **does not produce any output data** to store the results of the search. However, **without this information**, the subsequent **XOR-split cannot determine which branch to take**.



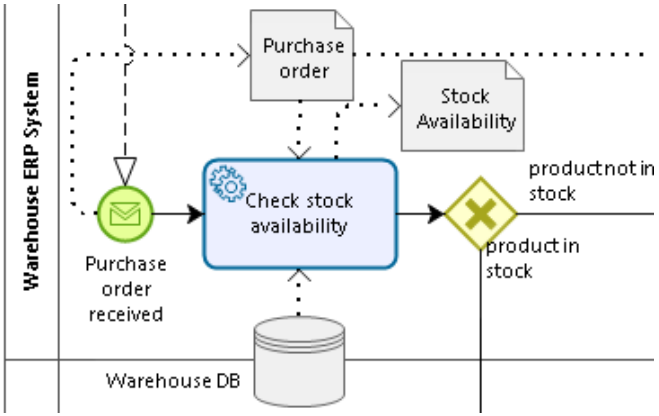
5. Bring the Process Model to an Adequate Granularity Level

- There is not necessarily a one-to-one mapping between the tasks in a business-oriented model and those in the corresponding executable model.
- BPMSs add value if they **coordinate handovers** of **work** between resources.
  - **Aggregate** two or more consecutive tasks assigned to the **same resource**.
  - **Split** tasks if they require **different resources**.



6. Specify Execution properties

- In the last step, we need to specify ***how*** each model element is effectively implemented by the BPMS.



  - For example, for "*Check stock availability*", we need to specify:
    - the **service** provided by the ERP system to check stock levels and its location in the network;
    - the **product information** in the purchase order that is required by this service (i.e. the format of the input object);
    - the **information produced by the service** (i.e. the format of the output object).

Specify Execution properties

- The implementation details of a business process are called ***execution properties***, and refer to:
  - Process variables, messages, signals, errors
  - Task and event variables and their mappings to process variables
  - Service details
  - Code snippets
  - Participant assignment rules and user interface structure
  - Task, event and sequence flow expressions
  - BPMS-specific: work queues, forms, connectors…
- These properties do not have a graphical representation in BPMN, but are stored in the **BPMN 2.0 interchange format**.
  - it is a textual representation of a BPMN model in XML format;
  - it supports the interchange of BPMN models between tools and also to serve as input to a BPMN execution engine.
- Often this step is performed directly with the help of the BPMS.