DELIVERY SWAB

Matteo Saglimbeni - Fabio Rossanigo

20/07/2022

OUR COMPANY

Delivery Swab is a company that handles the **order**, the **home deliver** and the **test** of swabs for customers who are **unable to move** from their houses. The customer selects the details of his order, a medical operator will be sent to perform the test and after some time the swab result will be received by a notification.

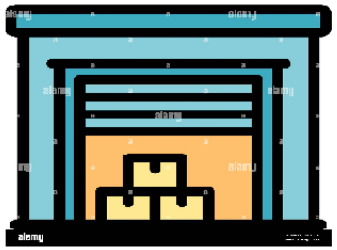
OVERVIEW

CUSTOMER

BANK

DELIVERY SWAB

HOSPITAL

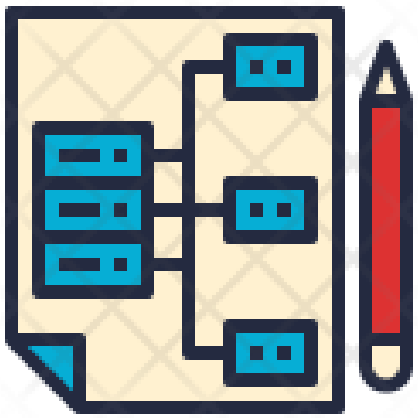


SUPPLIER

WHAT HAVE WE DONE

ArchiMate modelling	Business Process Choreography & Orchestration	Soundness checks	Design and implementation of 3rd party API's	Executable Process
ArchiMate is a language for enterprise architecture modelling used to the describe the various aspects and layers	Choreography diagram defines the sequence of interaction among the participants. Each task is a step of interaction between two participants. Orchestration diagram instead focuses on the lifecycle of the process.	The Soundness check is a validation process that tells us if our process is in presence of behavioural anomalies, This is done by first converting the BPMN diagram in a petri net diagram.	API's needed to manage the interaction with the external 3rd party services.	Executable simulation of the Delivery Swab process thanks to Camunda, Swagger tools and REST API.

ARCHIMATE MODEL



ArchiMate is a language for **enterprise architecture modelling** to support the description analysis and visualization of the architecture within and across business domain.

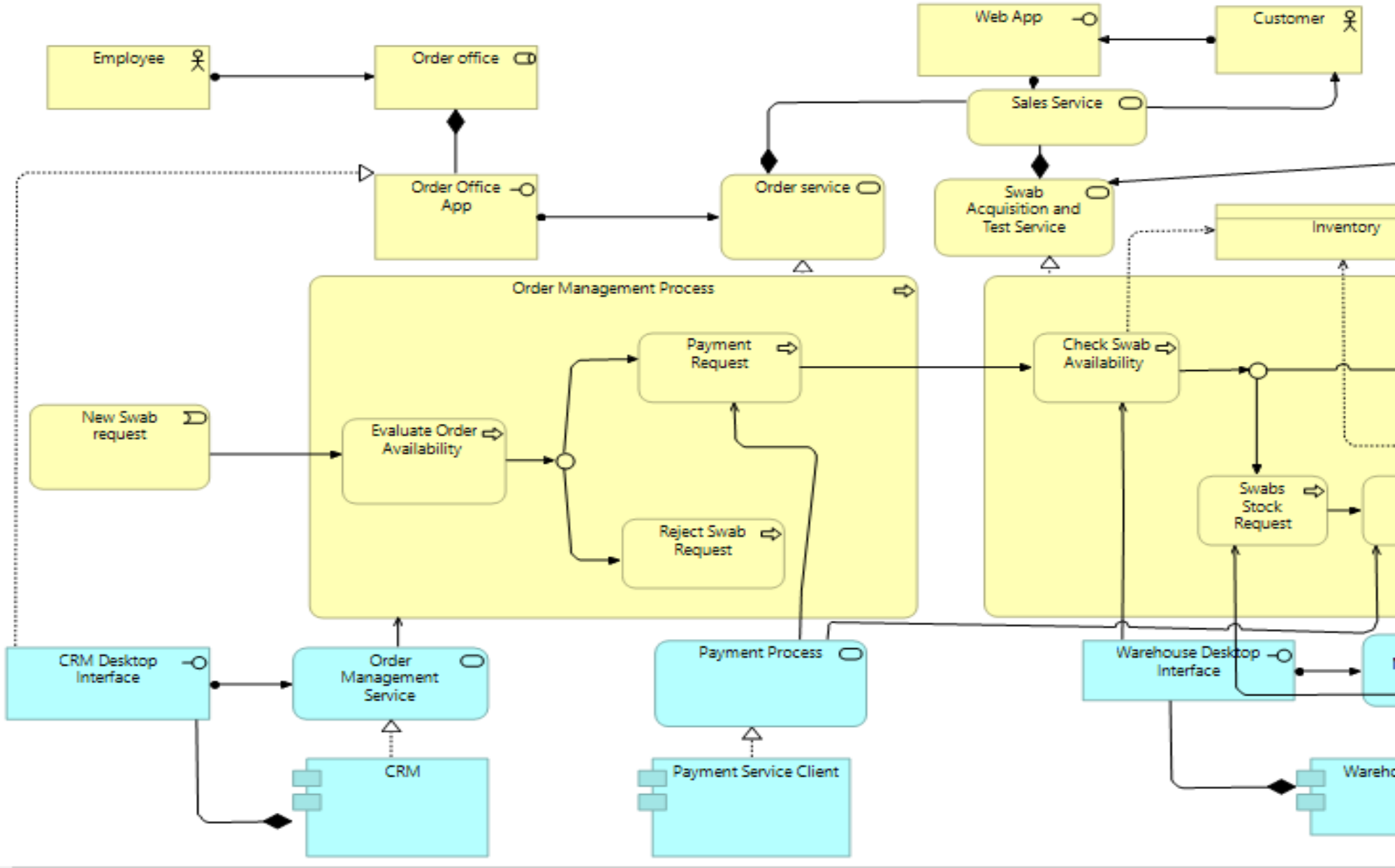
The goal of the ArchiMate model is to define the **process at high level of granularity**, how the consumer is connected to the enterprise and how the processes are related

ARCHIMATE MODEL(PART I)

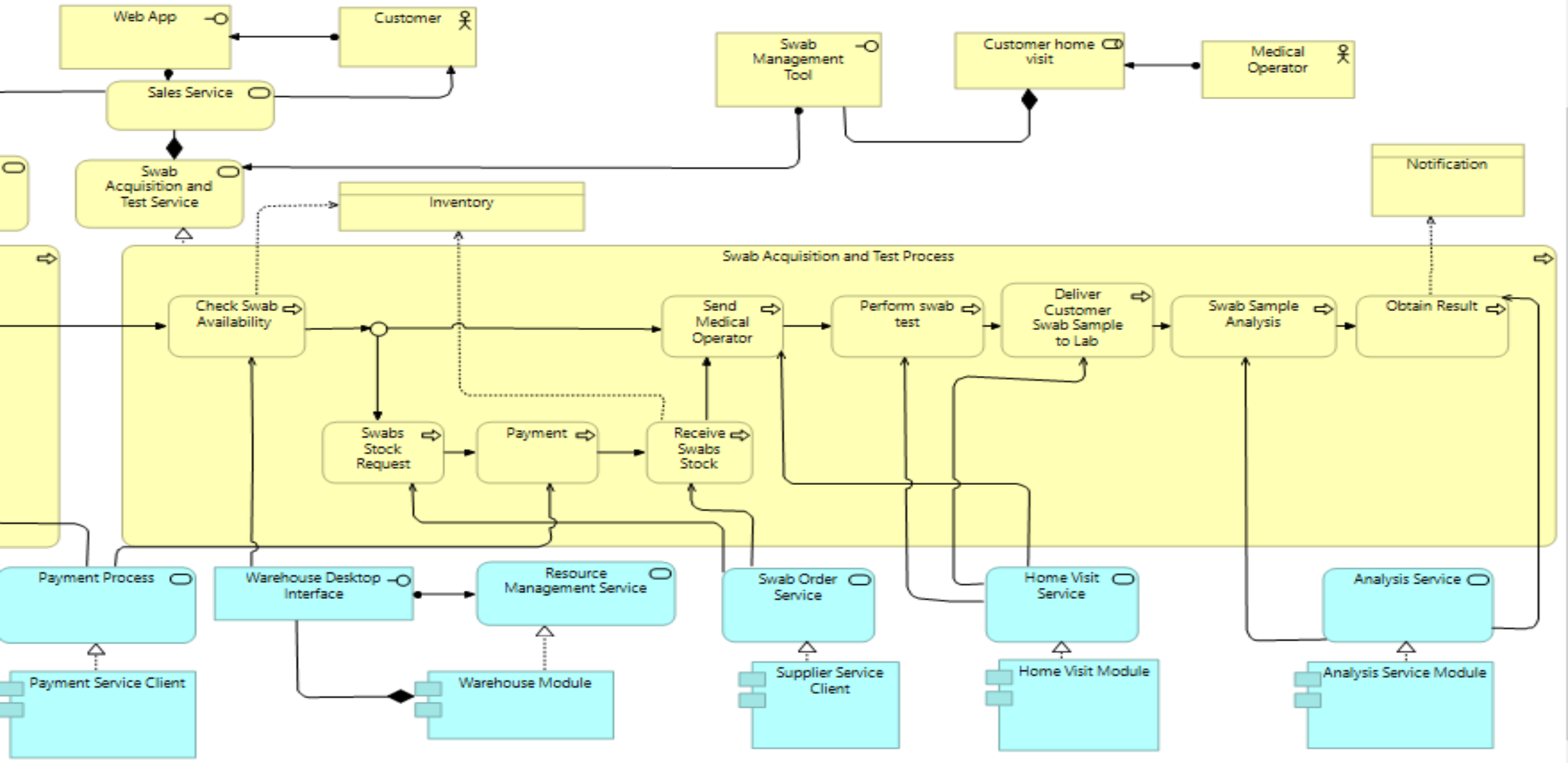
Our model consist of two main processes:

- **Order Management:** which realizes the *Order Service*
- **Swab Acquisition and Test:** realize the *Swab Acquisition and Test service*

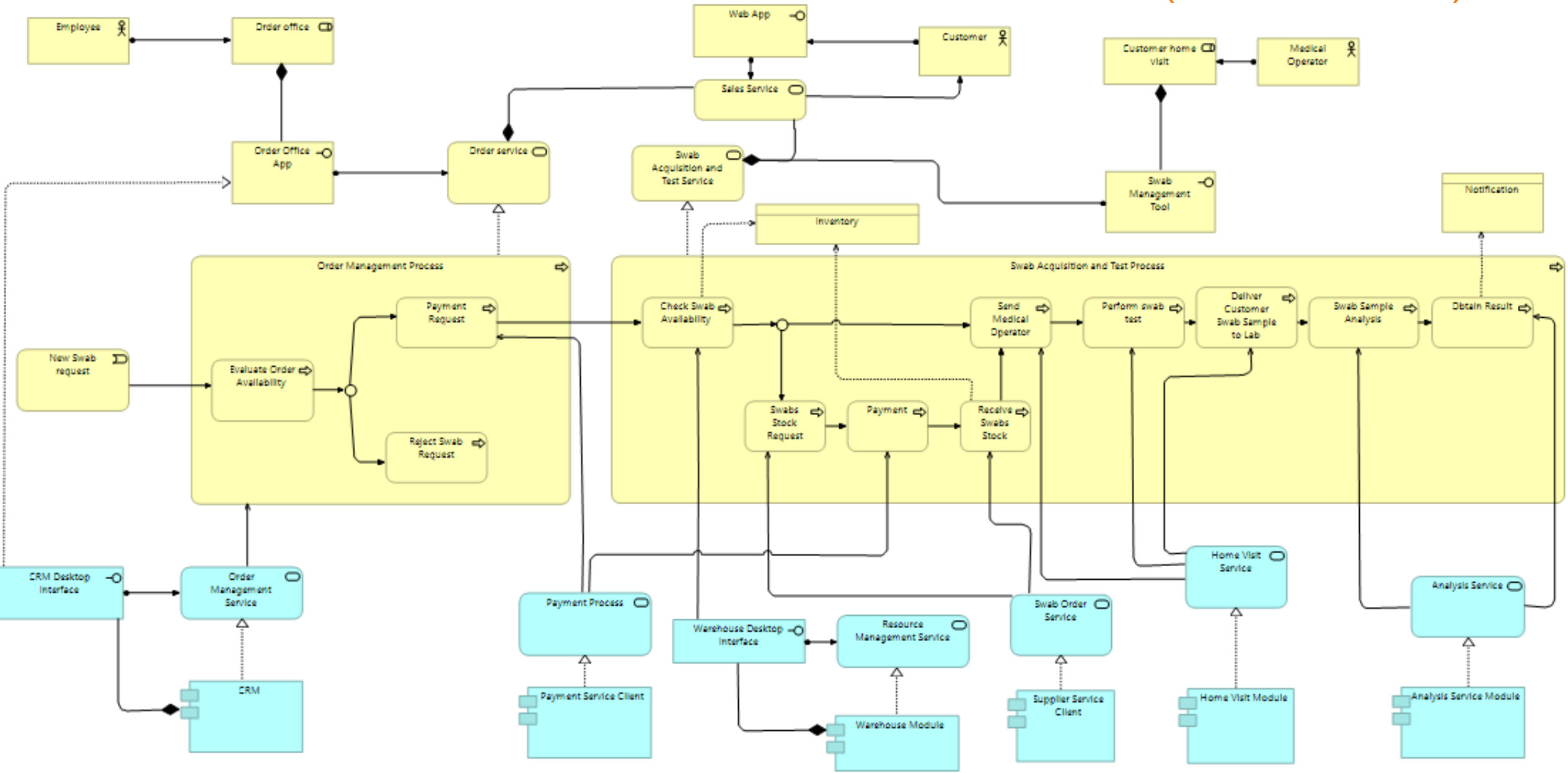
The two services contribute to the composition of the **Sales Service** which il the main service of Delivery Swab.



ARCHIMATE MODEL(PART II)



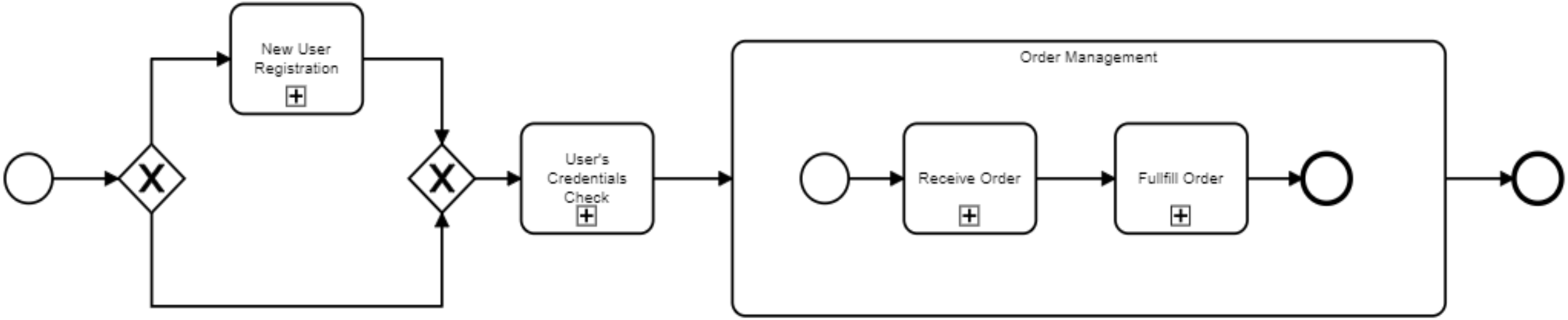
ARCHIMATE MODEL(FULL VIEW)



HIGH LEVEL CHOREOGRAPHY & ORCHESTRATION

The main subprocesses of our company:

- **User Registration**
- **User Credentials Check**
- **Receive Order**
- **Fulfill Order**

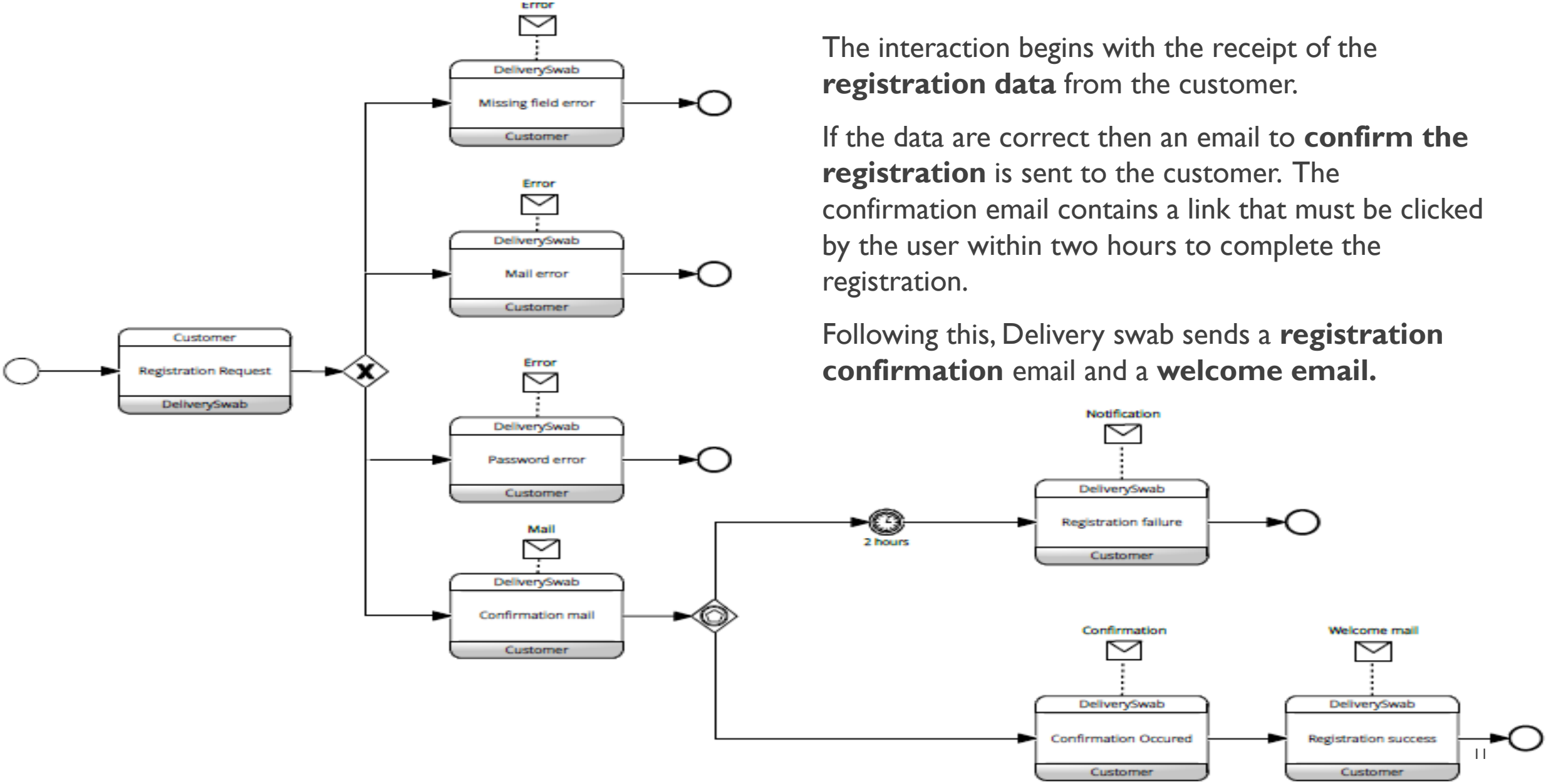


CHOREOGRAPHY

The choreography diagram provides a **high-level perspective** focused **only on the relationships among the organization**, but without specifying the internal processes.

It does not take the standpoint of any of the participants and typically is the base from which the collaboration diagram is derived.

CHOREOGRAPHY - USER REGISTRATION

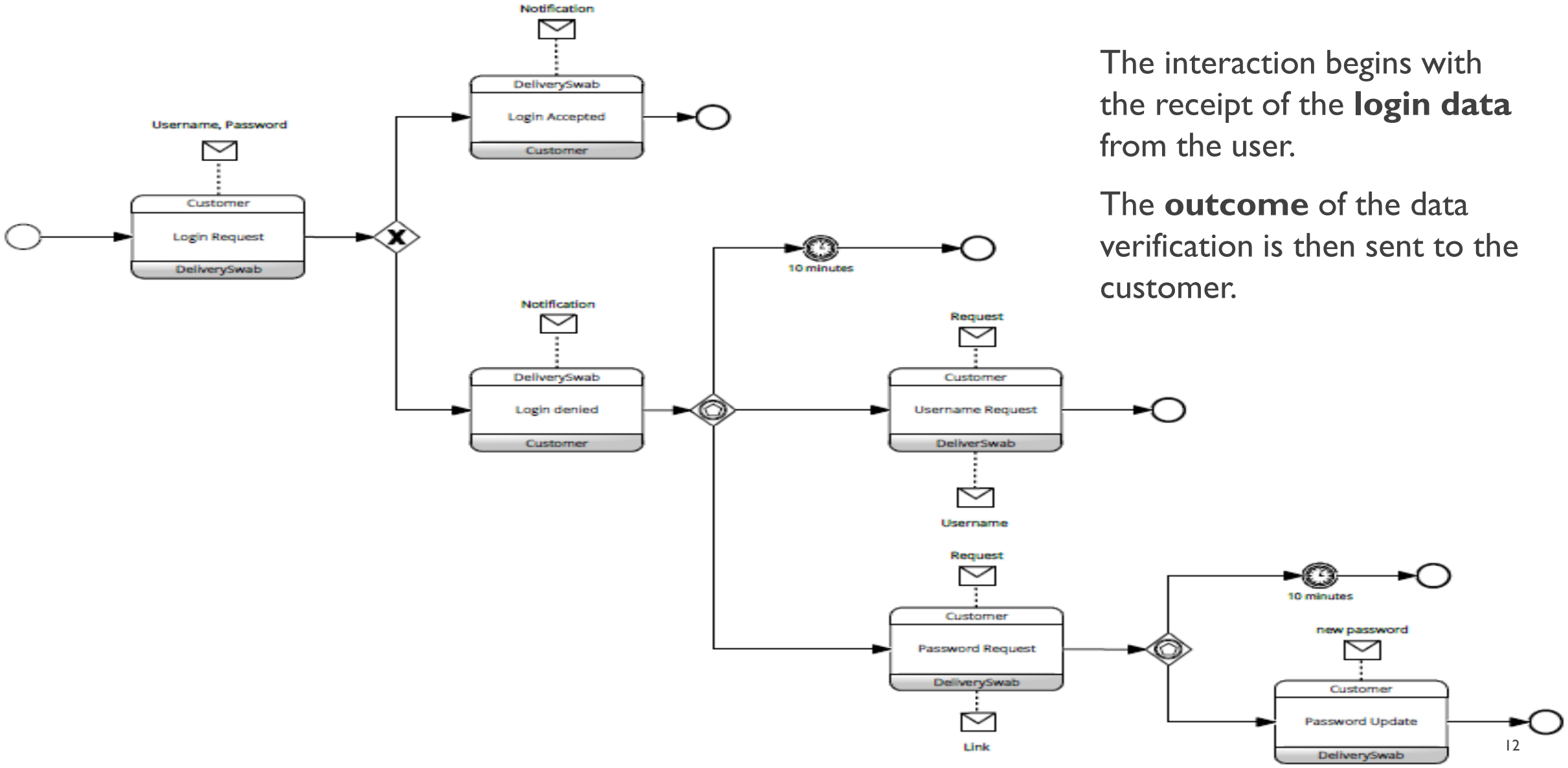


The interaction begins with the receipt of the **registration data** from the customer.

If the data are correct then an email to **confirm the registration** is sent to the customer. The confirmation email contains a link that must be clicked by the user within two hours to complete the registration.

Following this, Delivery swab sends a **registration confirmation** email and a **welcome email.**

CHOREOGRAPHY - USER LOGIN

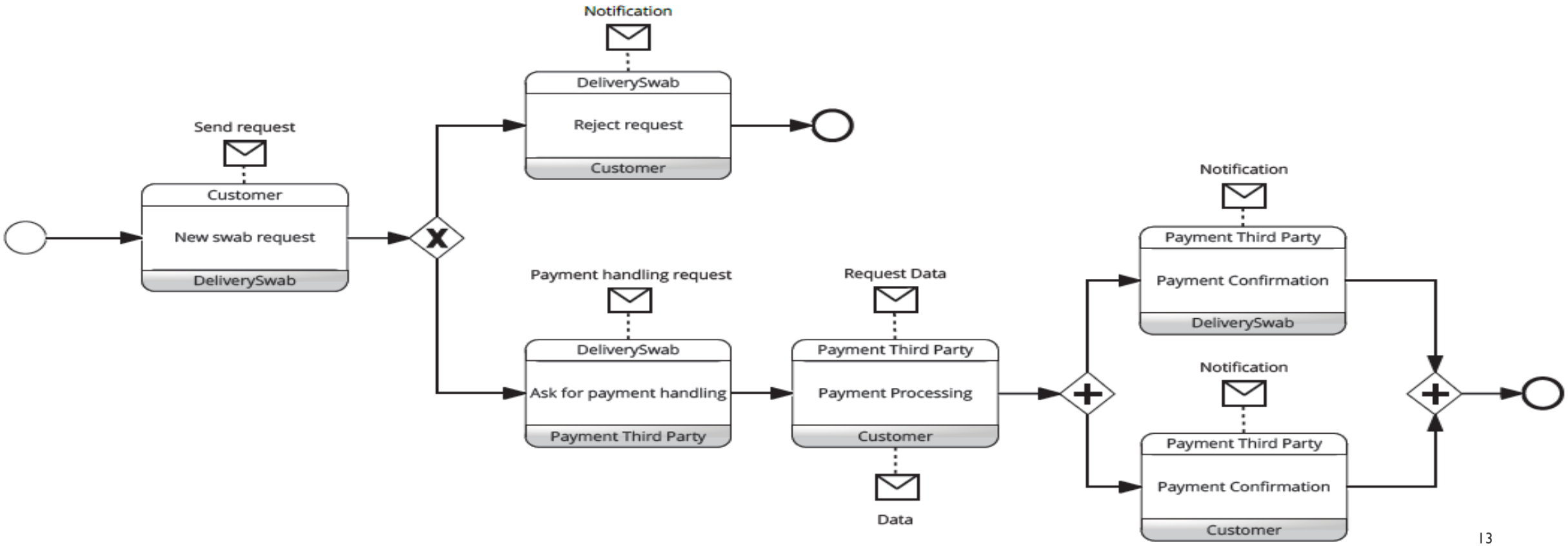


The interaction begins with the receipt of the **login data** from the user.

The **outcome** of the data verification is then sent to the customer.

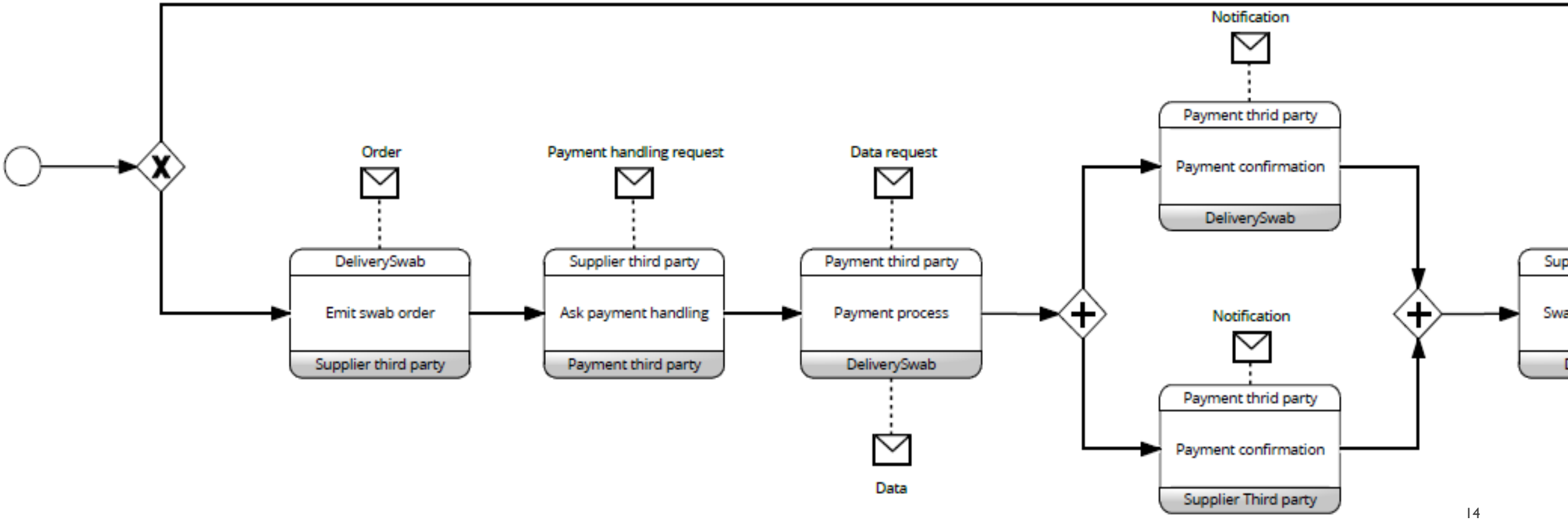
CHOREOGRAPHY – RECEIVE ORDER

The interaction begins with the receipt of the **order data**. Once received, the **payment** process takes place through the exchange of messages between the company and the 3rd-party and between the 3rd-party and the user.

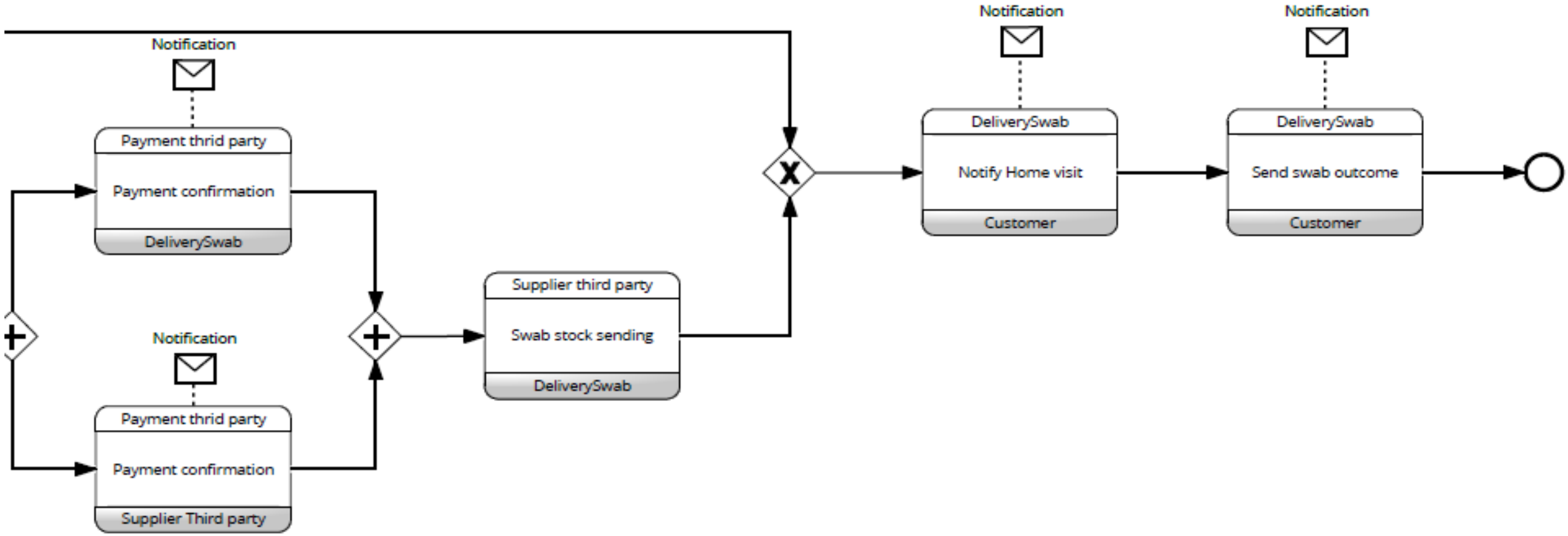


CHOREOGRAPHY – FULFILL ORDER(PART I)

The **order** is emitted. Then follows the actual **payment of the Deliver Swab**.

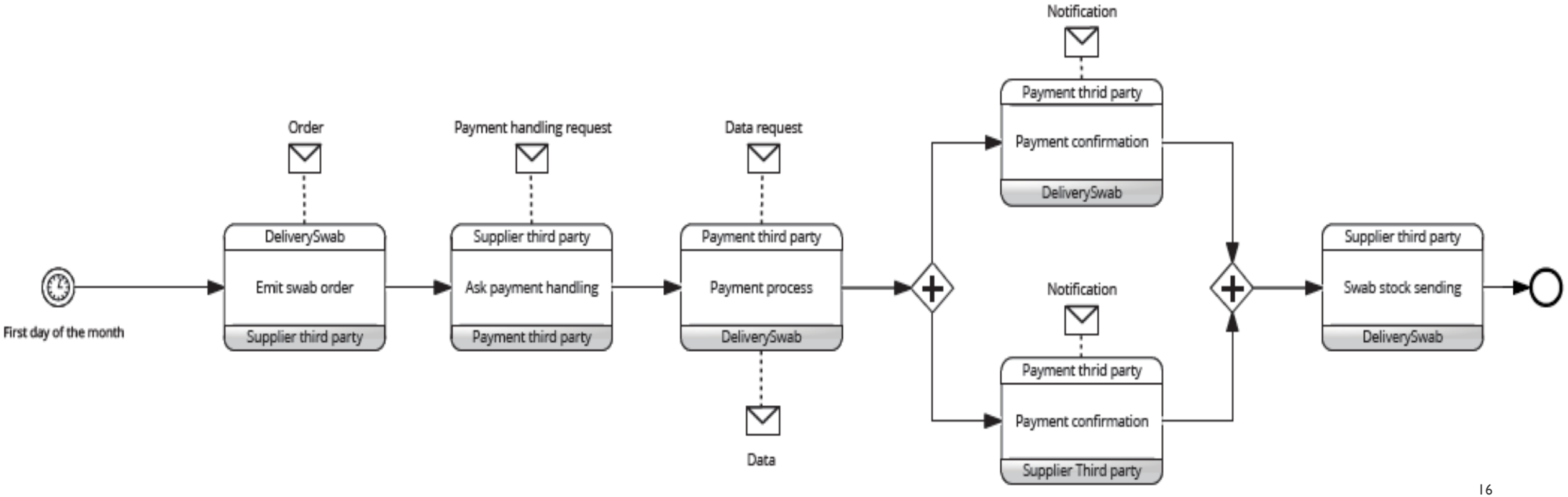


CHOREOGRAPHY – FULFILL ORDER(PART II)

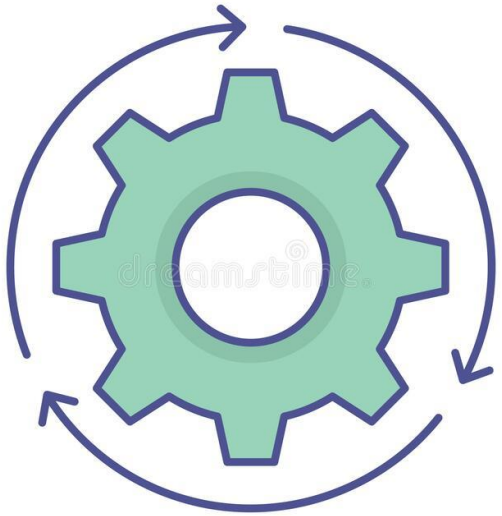


CHOREOGRAPHY – FULFILL ORDER(PART III)

The payment to the delivery partner takes place every month and, also in this case, it is managed by the payment 3rd-party.



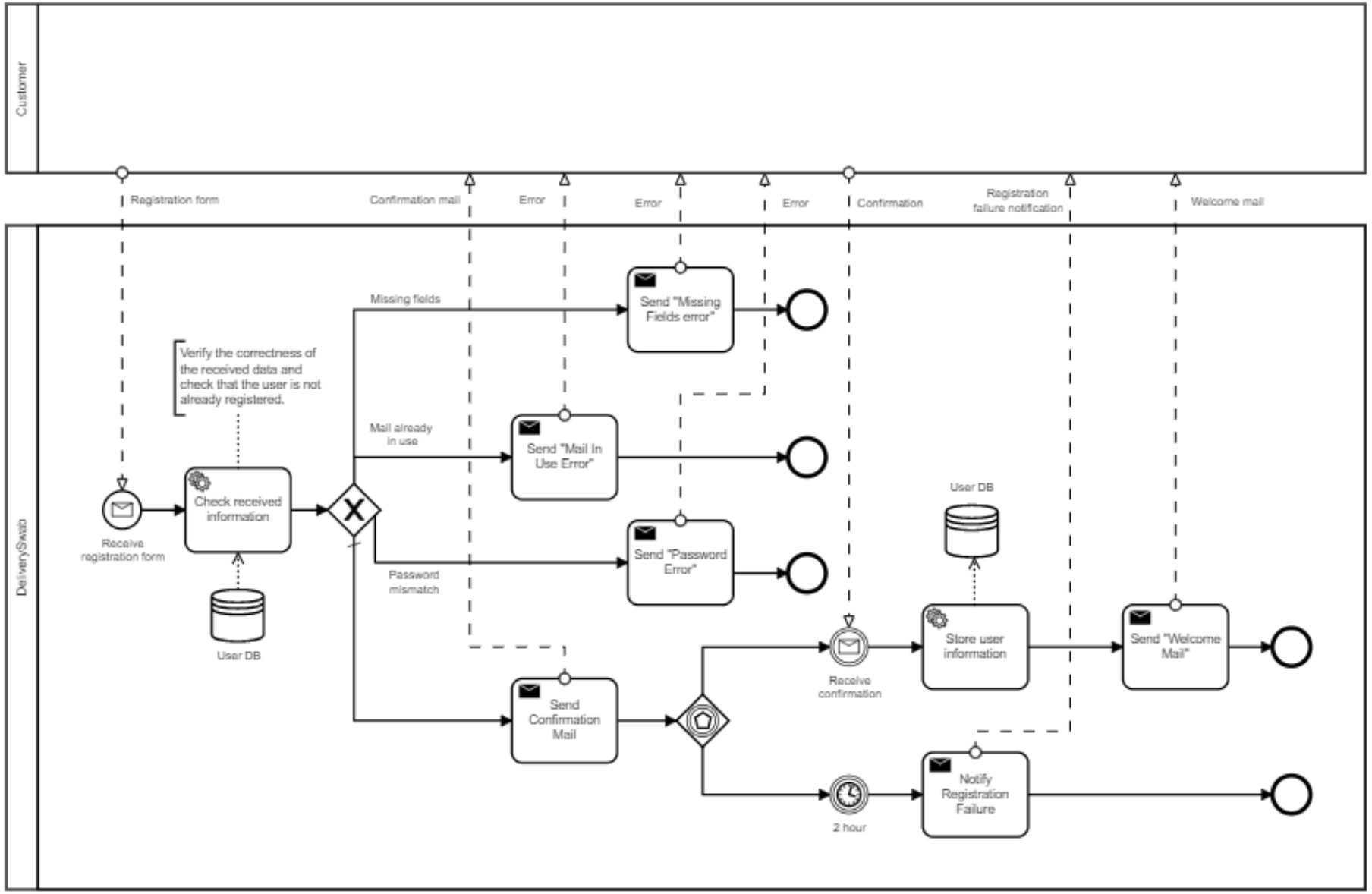
ORCHESTRATION



Orchestration concerns the management of **activities executed under a common controlling element** (the orchestrator). Usually, the boundary of an orchestrator is an organization.

The orchestrator controls the execution of the business process according to a control-flow model. The control-flow is usually modelled based on a set of patterns.

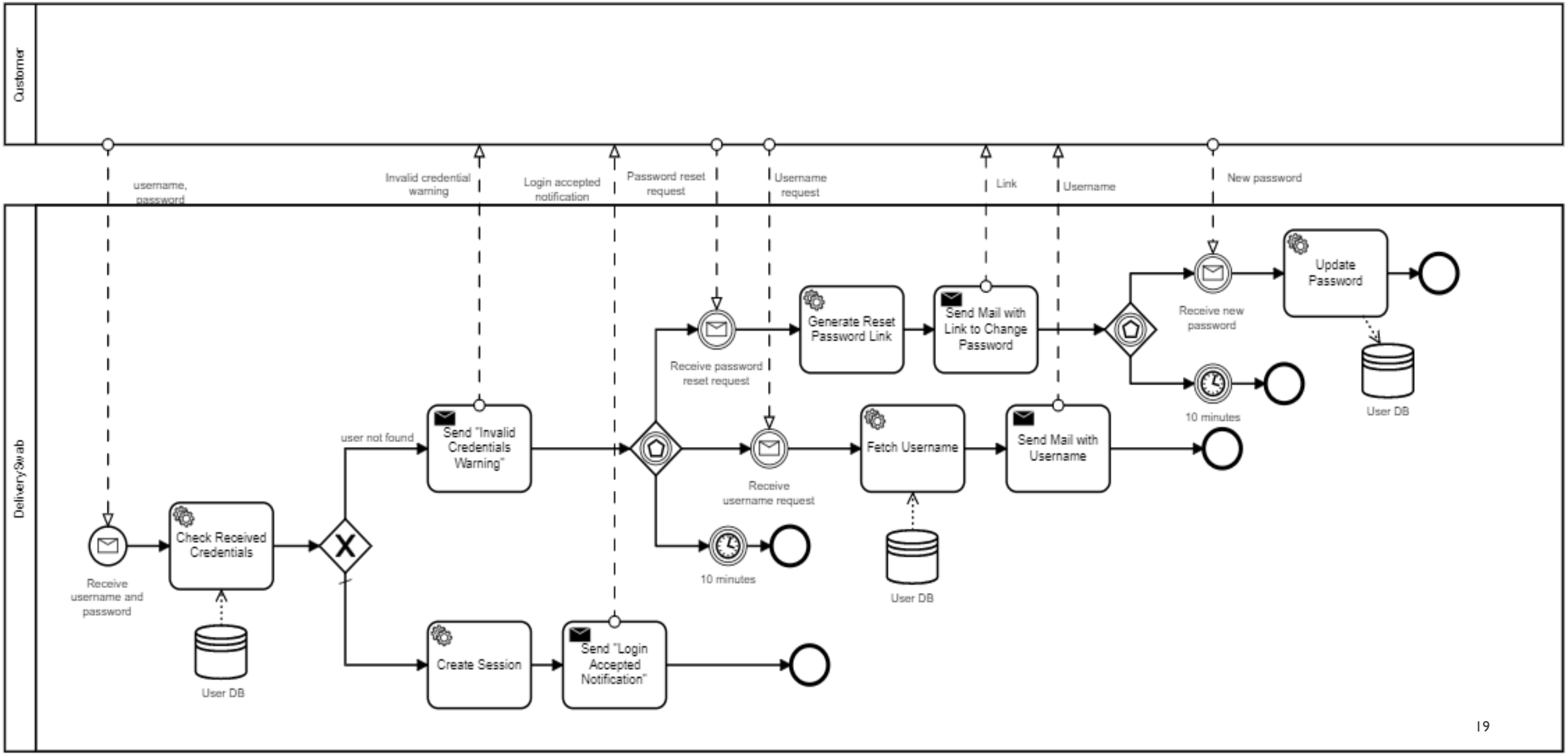
ORCHESTRATION - USER REGISTRATION



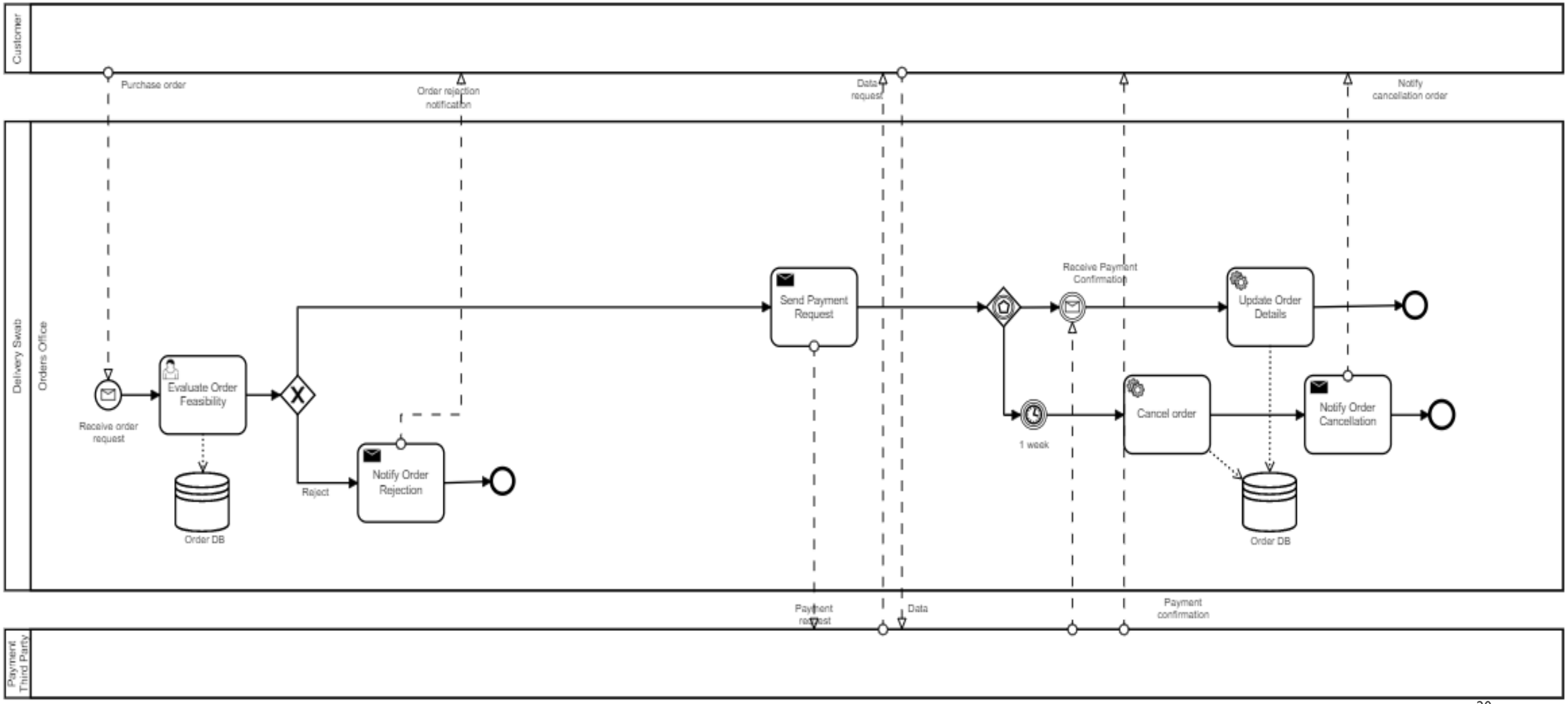
During the registration we control of the **correctness** of the data.

The **database** is used both to verify the customer data and to memorize them when the registration is confirmed.

ORCHESTRATION - USER LOGIN

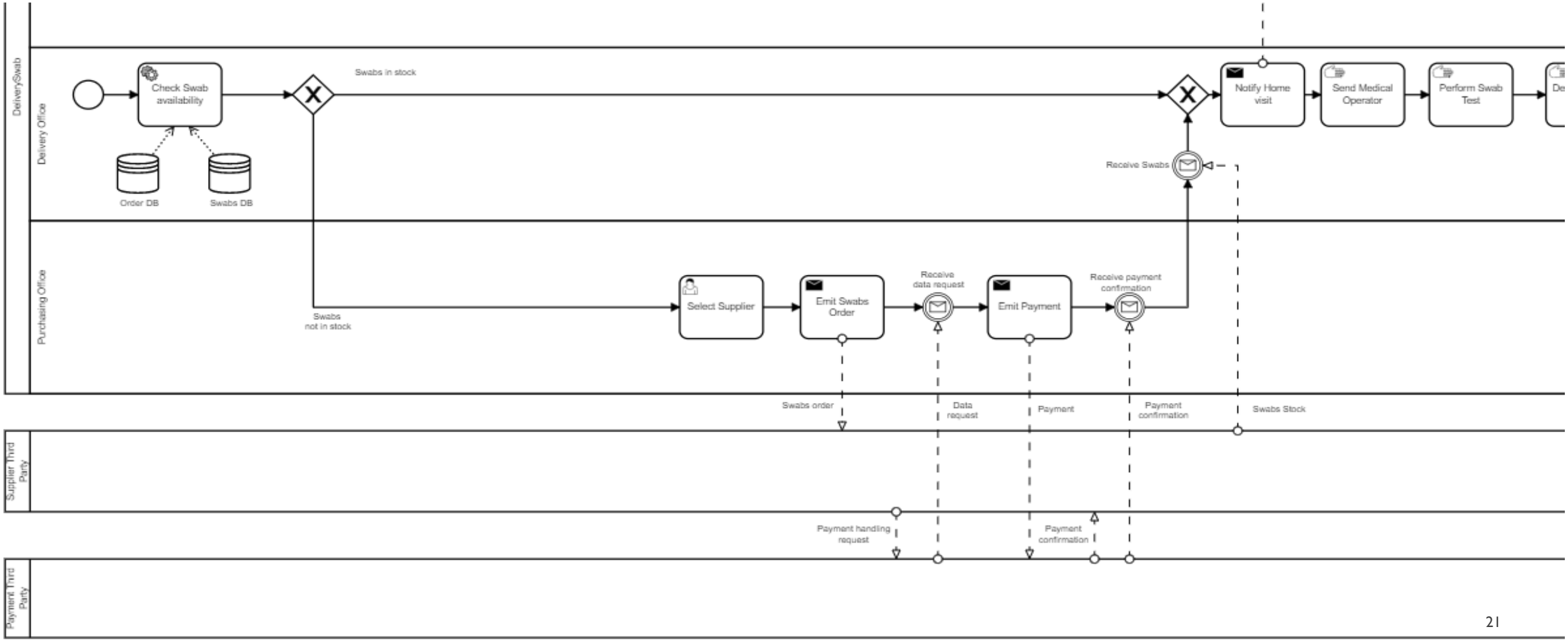


ORCHESTRATION – RECEIVE ORDER



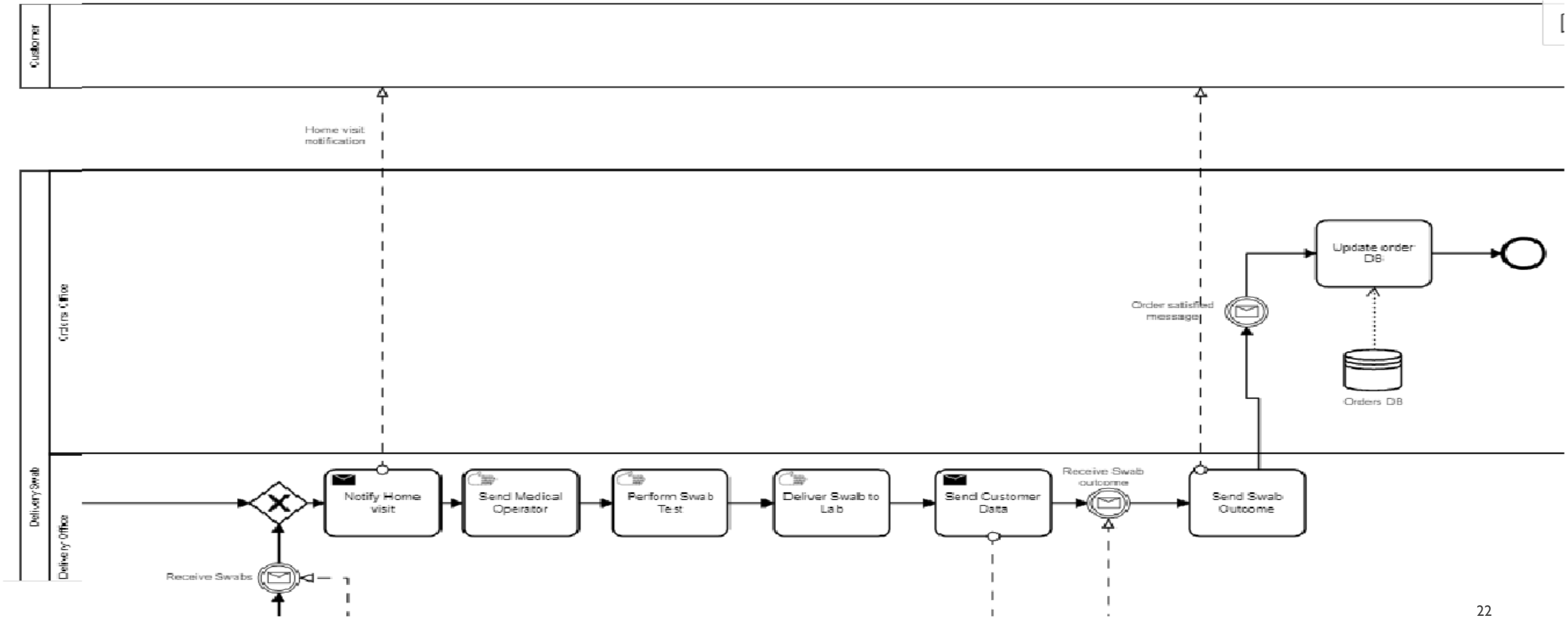
ORCHESTRATION – FULFILL ORDER (PART I)

A service task verifies the **availability of swabs**. If the resources are not in stock, the company has to **purchase swabs stocks** before proceeding with the home visit.
Once the company has received the swabs, or if the swabs are already in stock, a set of user tasks, concerning the **sending of the medical operator and swab test**, are executed.



ORCHESTRATION – FULFILL ORDER (PART II)

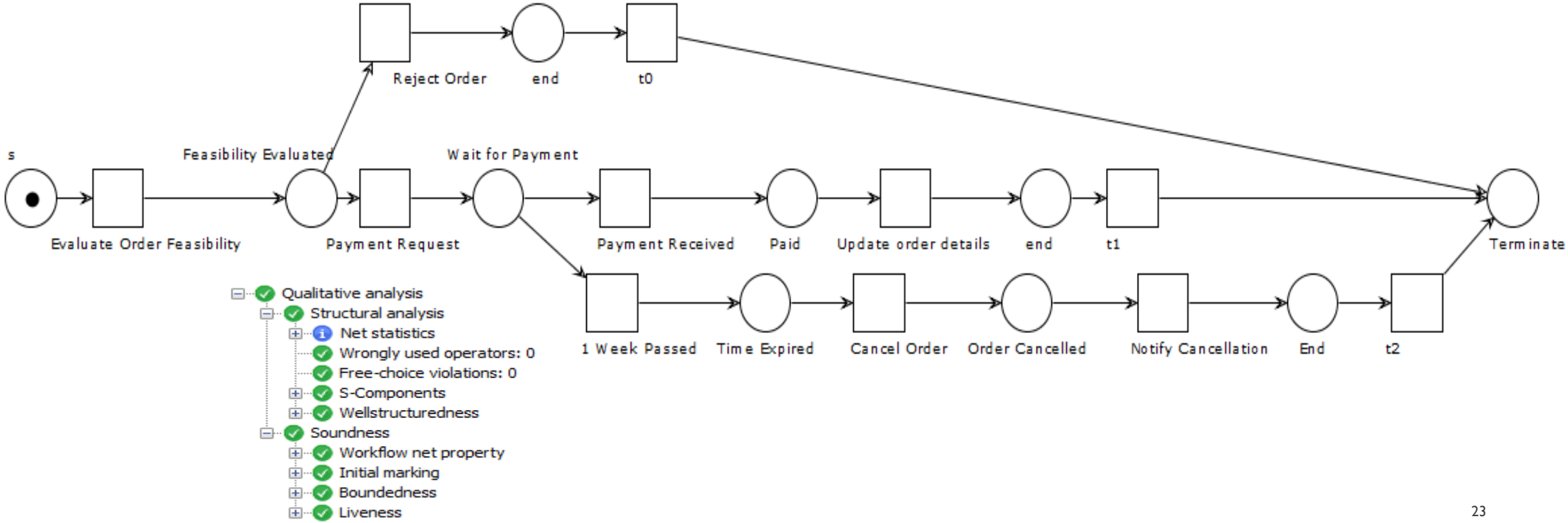
After the swabs are finally available, the company send an home visit notification and a medical operator goes to customer house ready **for performing a swab test**. After the swab sample has been collected, the operator delivers it to a 3rd-party hospital that handles the analysis itself. When the swab outcome is ready, a notification is forwarded to the customer.



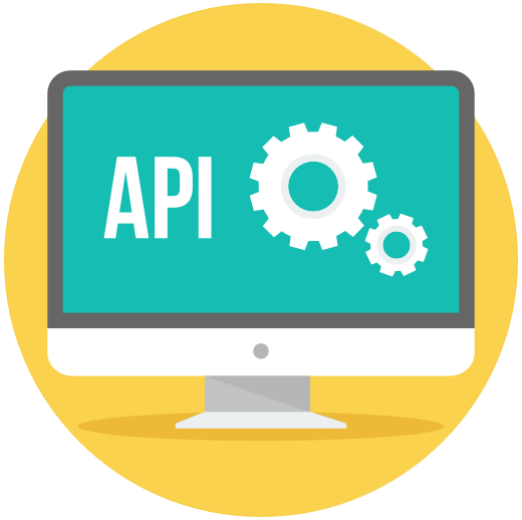
SOUNDNESS

The **soundness** is a property that allow to identify correct WF-nets and so business processes.

A WF-Net (corresponding to a BP) is sound if and only if: for any case, the procedure will **terminate** eventually, and at the moment the procedure terminates there is **a token** in place 'o' (the final place) and all the other places are **empty**. The soundness is checked on the *Receive Order* phase.



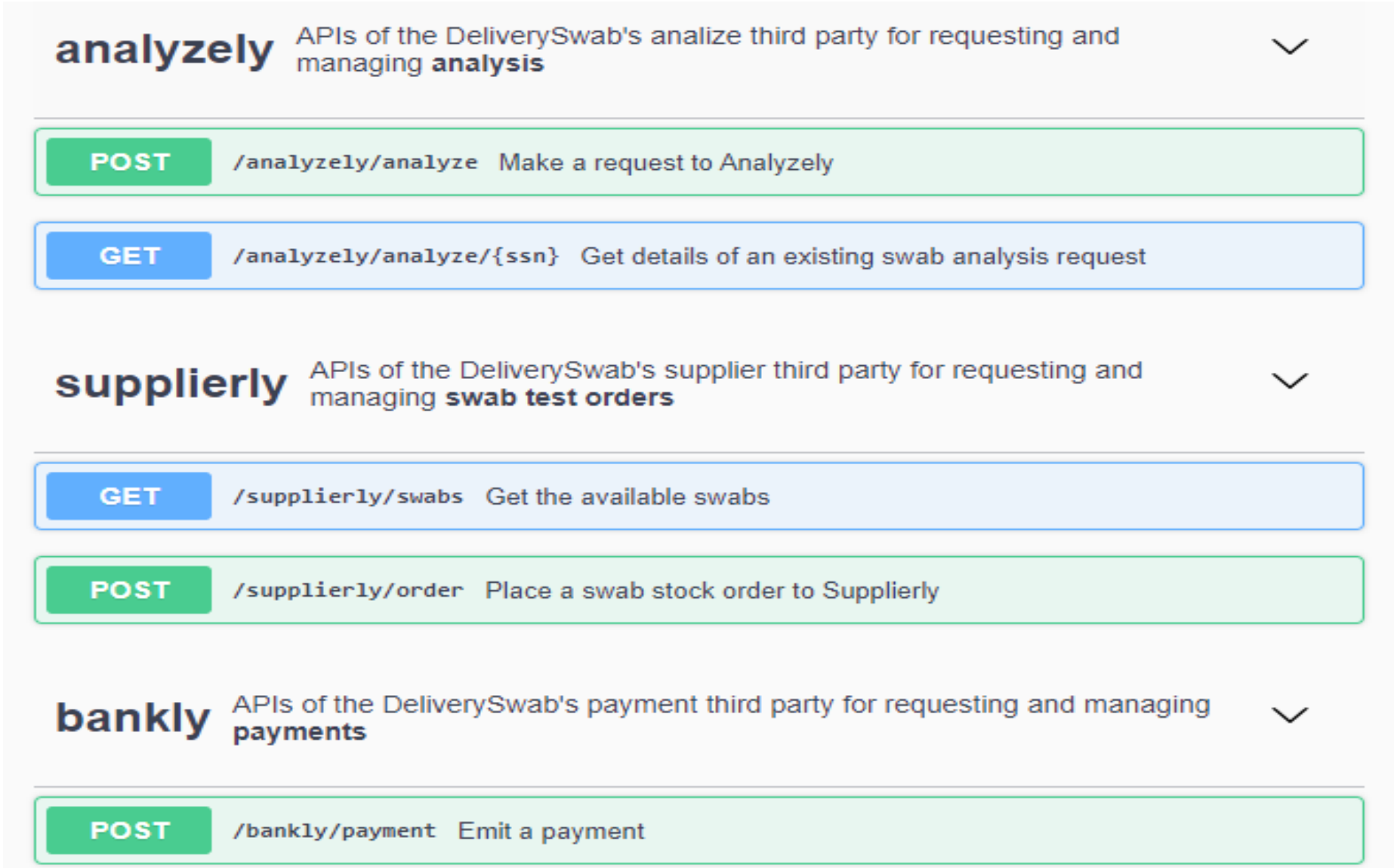
REST API



An **API** is a set of definitions and protocols for building and integrating application software.

A **REST API** (also known as RESTful API) is an architectural style for an application program interface (API) that uses HTTP requests to access and use data. That data can be used to GET, PUT, POST and DELETE data types, which refers to the reading, updating, creating and deleting of operations concerning resources.

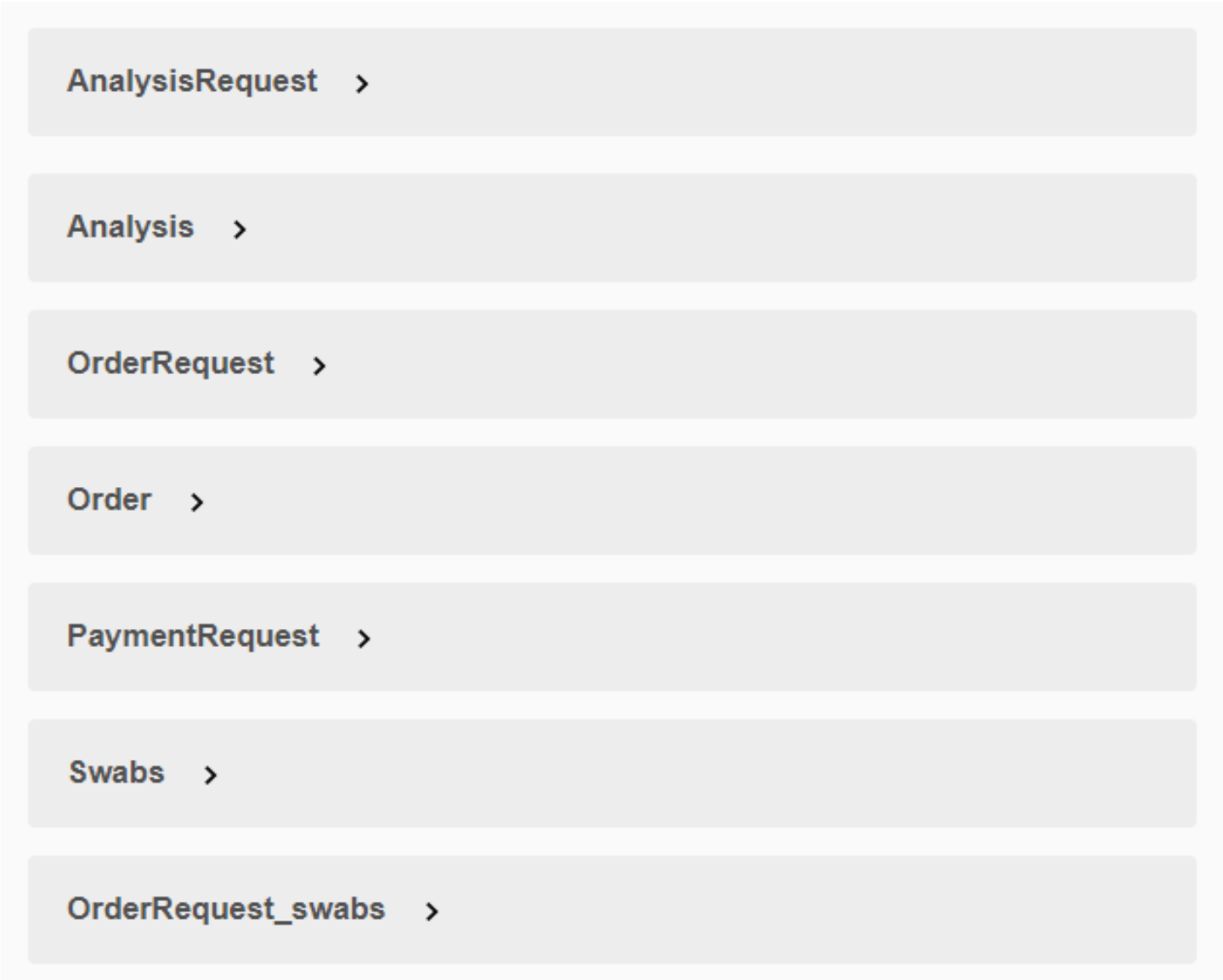
REST API



We have three **external services**:

- **Analyzely(the Hospital)**: requesting and executing the swab analysis.
- **Supplierly(a swab supplier)**: requesting and managing swab orders.
- **Bankly(a bank)**: requesting and managing payments.

REST API SCHEMAS



The majority of the exchanged messages are structured according to some schemas that we have defined.

POST - ANALYZELY

POST `/analyzely/analyze` Make a request to Analyzely

This API allows to submit a request for a new delivery

Parameters

No parameters

Request body required

Details of the customer that made the swab test

Examples:

Example Value | Schema

```
{
  "name": "Mario",
  "surname": "Rossi",
  "mail": "mario.rossi@gmail.it",
  "ssn": "MRARSS97H28C933E",
  "date": "28/6/2022"
}
```

Responses

Server response

Code	Details
200	**Response body** `{ "name": "Mario", "surname": "Rossi", "mail": "mario.rossi@gmail.it", "ssn": "MRARSS97H28C933E", "date": "28/6/2022", "status": "processing", "swabResult": "not analyzed" }`
400	Bad request, the address is invalid or not served by Deliverly Example Value | Schema `(no example available)`

GET `/analyzely/analyze/{ssn}` Get details of an existing swab analysis request

This API allows to get details about a analysis request

Parameters

Name	Description
ssn * required string *(path)*	Analysis identifier MRARSS97H28C933E

Responses

GET - ANALYZELY

Code	Description
200	Analysis details Media type application/json Controls Accept header. Example Value | Schema

```
{
  "name": "Mario",
  "surname": "Rossi",
  "mail": "mario.rossi@gmail.com",
  "date": "28/6/2022",
  "status": "terminated",
  "swabResult": "negative"
}
```

404	A problem occurred

POST - SUPPLIERLY

POST `/supplierly/order` Place a swab stock order to Supplierly

This API allows to submit a swab stock order

Parameters

No parameters

Request body required

Details of the order to be placed

Examples:

Example Value | Schema

```
{
  "swabs": {
    "id": 0,
    "quantity": 4
  }
}
```

Responses

Code	Description

200	Order details
	
	Media type: application/json
	Controls Accept header.
	
	Example Value | Schema
	
	{
	  "deliveryDate": "01/09/2022",
	  "cost": 40,
	  "iban": "IT78-F569-3411-1000-0000-0145-123",
	  "swabs": [
	  {
	  "id": 0,
	  "quantity": 4
	  }
	  ]
	}

400	Bad request, required materials do not exist or invalid order Example Value | Schema (no example available)

GET `/supplierly/swabs` Get the available swabs

This API allows to get the available swabs

Parameters

No parameters

Responses

GET - SUPPLIERLY

Code	Description
200	Swabs

Media type

application/json

Controls Accept header.

Example Value | Schema


```
{
  "id": 0,
  "price": 10,
  "availability": 100
}
```

POST - BANKLY

POST `/bankly/payment` Emit a payment

This API allows to emit a payment

Parameters

No parameters

Request body required

Details of the payment to be placed

Examples:

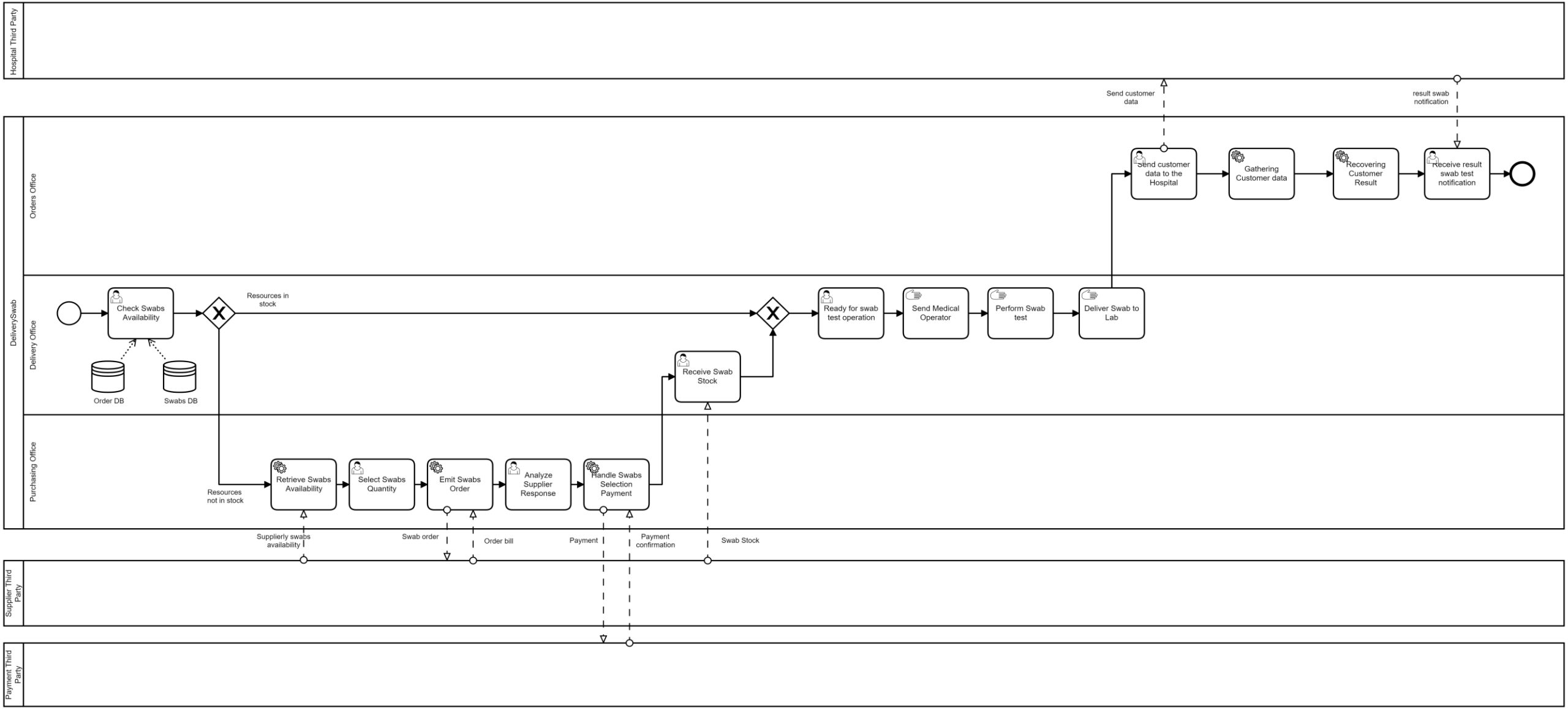
Example Value | Schema

```
{
  "iban": "IT78-F569-3411-1000-0000-0145-123",
  "amount": 1000
}
```

Responses

Code	Description
200	Payment successfully processed Example Value | Schema (no example available)
400	Insufficient credit to execute the transaction Example Value | Schema (no example available)
404	Invalid IBAN Example Value | Schema (no example available)

IMPLEMENTATION



Thanks for your attention!

by

DELIVERY SWAB