Technical Overview

Challenge #1:

JSON definition:

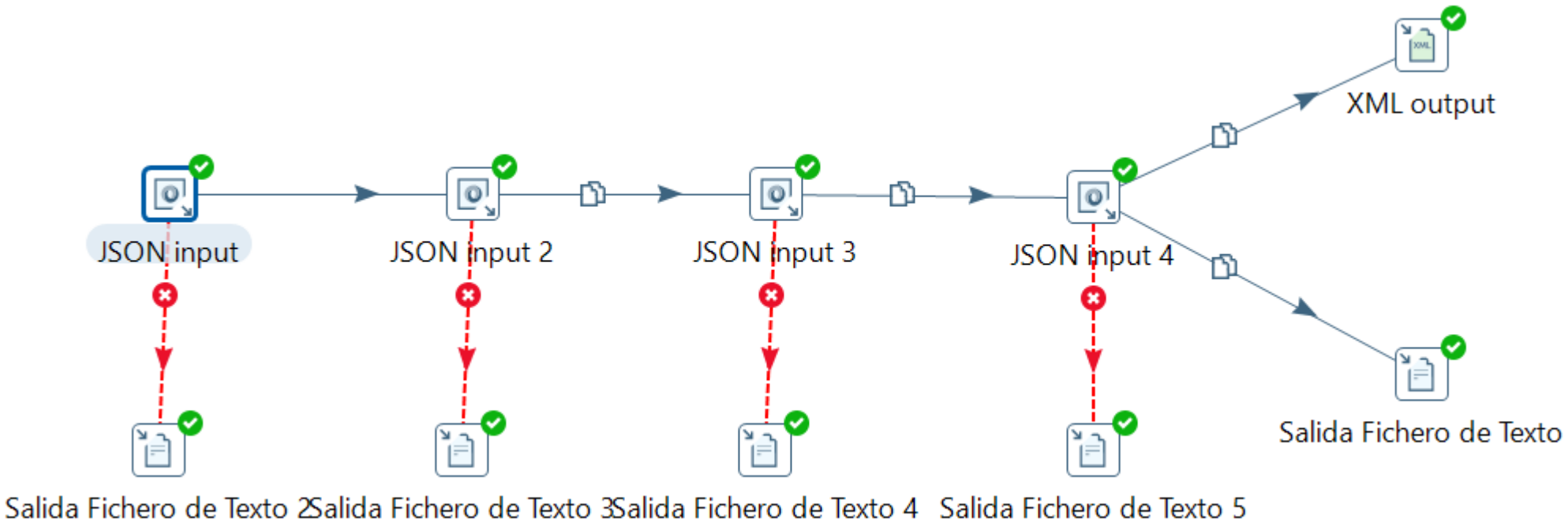
We have created the json structure adding new array fields to enrich the reporting part.

```
{
   "userid": "5ccefaba7231cea552605682",
   "user_name": "taylorkline@kidstock.com",
   "event": "product_view",
   "product": [
    {
     "category": "Tennis",
     "product_model": 753,
     "product_sex": "Women"
    }
   ],
   "timestamp": "2018-06-01 03:01:42",
   "device_info": [
    {
     "vendor": "Other",
     "device_model": 312
    }
   ],
   "gps": [
    {
     "latitude": -67.525498,
     "longitude": 92.252578
    }
   ],
   "applicattion": "Adidas Fitness"
  }
```

Challenge #1:

Normalization

Now, we have to normalize the json and generate de xml file. We Will generate data automatically using json-generator



Challenge #1:

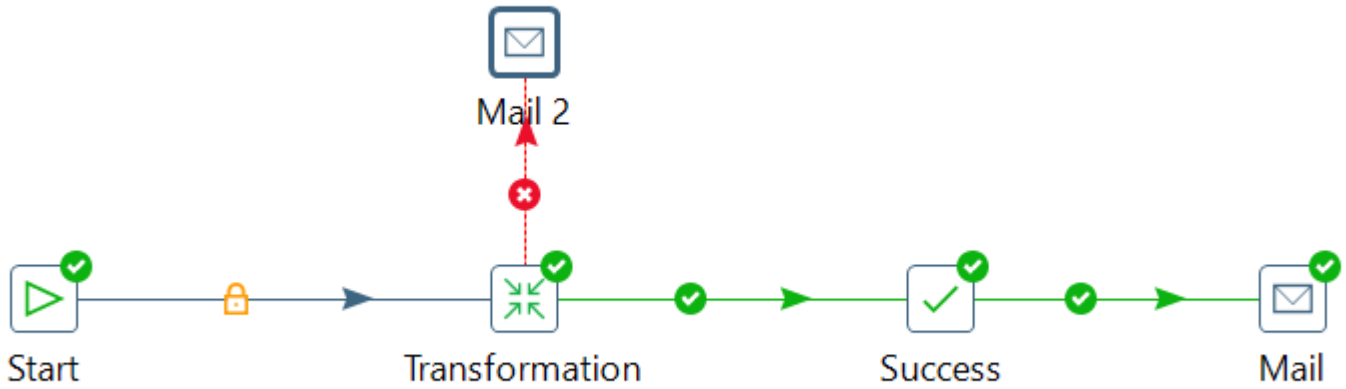
Batch Prcoess:

This transformation has been created for testing purposal.

For production environment you will have to implement a batch process thinking in incremental data. You have to define the time you will execute your process and how add this information to the model. For example we can define a time directory folder per days, and execute the process from the day before and add this information as incremental data to the csv.

RollBack:

For rollingback, you have all the errors saved into the errors folder. So you can try to re-apply the files in the error folders.

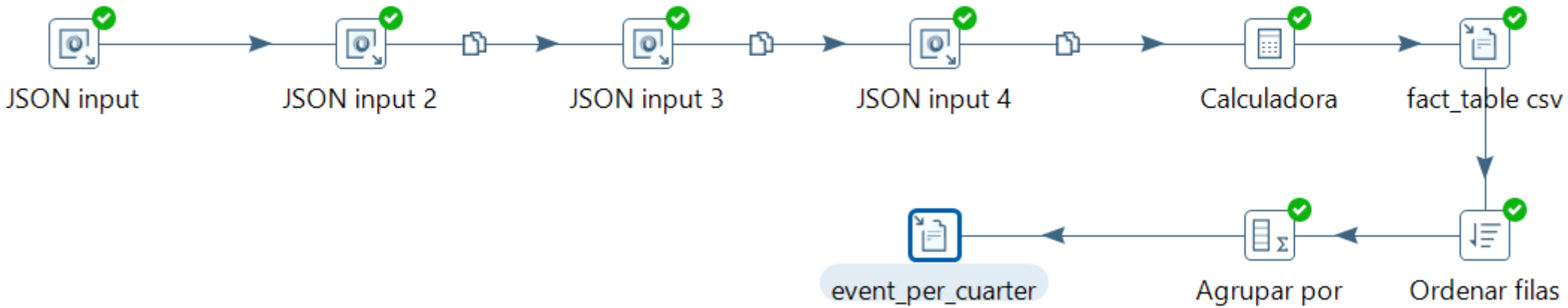


Technical Overview

Challenge #1:

Reporting

Now we have to use normalized data in a datawarehouse csv files.
First we will Split our csv timestamp in YEAR/MONTH/DAY/Quarter in our fact table definition.
With this information we can analyze our business in time line.
This data Will be our fact_table.
We can define new kpis to ease our analysis. For example we Will define a KPI like events per cuarter.



Technical Overview

Challenge #1:

Reporting

I think the best option is to generate de csv with the fact_table and use a database for generating the kpis. You Will have best performance. If you decide to use this option you have to define a PK in your table. You can use de id (userid) and the timestamp. Using this columns you will prevent to load duplicate rows. Related with this, it would be a good idea to create indexes and partitions by range in your reporting table schema.

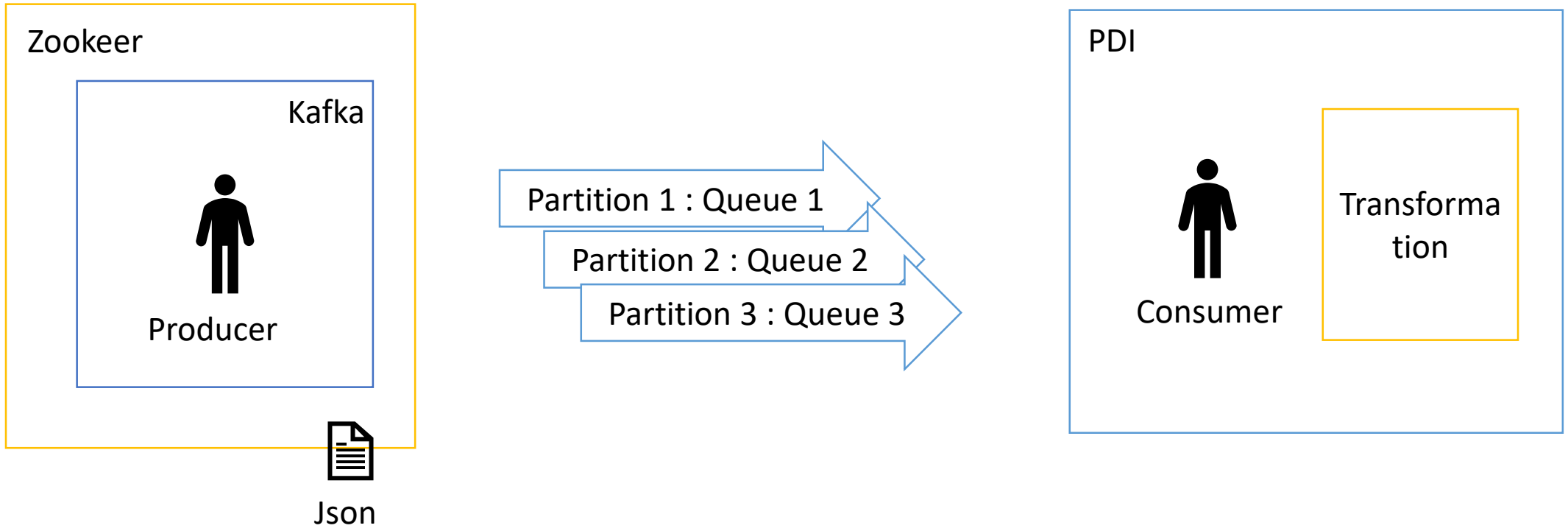
Technical Overview

Challenge #2:

API

We Will use Apache Kafka for creating and streaming API.

We need to install Zookeeper and Kafka to create the producer. Then when the producer are working we can start our consumers from PDI and receiving the data for transforming.



Challenge #2:

API

We Will use Apache Kafka for creating and streaming API.

We need to install Zookeeper and Kafka to create the producer. Then when the producer are working we can start our consumers from PDI and receiving the data for transforming.

Kafka consumer

records from stream Salida Fichero de Texto

Challenge #2:

API

You have to start the producer and then the consumer for catching the message.

In the producer you can define the redundancy – replication_factor and the queueing – partitions in the topic creation

kafka-topics.bat --create --zookeeper localhost:2181 --replication-factor 1 --partitions 3 --topic test

1.

kafka-console-producer.bat --broker-list localhost:9092 --topic test < C:\Users\cgonza\Desktop\data-integration\files\json\json_real_time.json

2.