Contract No. H2020 – 636078

INFORMATION TECHNOLOGIES FOR SHIFT TO RAIL

D6.7 – Business Analytics Ontology document (FREL)

Due date of deliverable: 31/10/2017

Actual submission date: 20/11/2017

Leader of this Deliverable: Polimi

Reviewed: Y

Document status		
Revision	Date	Description
1	01/07/2017	Initial version
1.1	31/08/2017	Final version
2	02/09/2017	Final version after TMC review
3	20/11/2017	Final version after dissemination level's change

Project funded from the European Union's Horizon 2020 research and innovation programme		
Dissemination Level		
PU	Public	X
CO	Confidential, restricted under conditions set out in Model Grant Agreement	
CI	Classified, information as referred to in Commission Decision 2001/844/EC	

Start date of project: 01/05/2015 Duration: 36 months

EXECUTIVE SUMMARY

This document presents the part of the IT²RAIL ontology that concerns the Business Analytics module. After a brief introduction, the document presents the modeling alternatives that were explored in the ontology design, and motivates the decisions that were taken. Then, the ontology concepts are explained in detail.

This deliverable is an incremental modification of Deliverable D6.1. The main concepts have not changed with respect to that deliverable. On the other hand, to better facilitate the retrieval of Key Performance Indicators (KPIs) by clients, KPI concepts have been grouped in explicit categories.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

ACK	Acknowledge
BA	Business Analytics
BT	Booking & Ticketing
CDT	Context Dimension Tree
CRUD	Create, Read, Update, Delete
CW	Cloud Wallet
E-R	Entity Relationship
GUI	Graphical User Interface
KPI	Key Performance Indicator
NACK	Not Acknowledge
PA	Personal Application
TC	Travel Companion
TS	Travel Shopper
TT	Trip Tracker
UI	User Interface
UUID	Universally Unique IDentifier

1 INTRODUCTION

This document presents the fragment of the IT²RAIL ontology that concerns Business Analytics (BA) module. In more detail, the BA ontology introduces the concepts necessary to describe and semantically annotate the data that the BA module exports towards the other functional areas of the system and the external world.

The analytics results exported by the BA module are constituted by Key Performance Indicators (KPIs), i.e., indicators quantitatively describing the performance of the services provided by an organisation in the IT²RAIL scenario, the performance of transportation services. Therefore, the BA ontology deals with defining the KPIs, with their descriptions, their input parameters and the outputs of their evaluations.

With respect to the previous version of the BA ontology, this one makes more precise the notion of "category" of KPIs, to facilitate the retrieval of groups of KPIs, rather than single ones.

The document describes in Section 2 the alternatives identified in the ontology definition and the design choices. Section 3 provides a formal description of the concepts included in the ontology and their relationships.

2 METHODOLOGY

This section discusses the main alternatives that we identified to model the BA concepts, and explains our final design choices.

The BA module must deal with various kinds of information to derive the required indicators, like timetables, bookings, delays, etc. However, these concepts are inputs to the BA computations, while the aim of the BA ontology is to describe and annotate the concepts to be exported. As a consequence, the BA ontology must represent just the information concerning the indicators.

The BA fragment of the ontology needs thus to be built around the concept of KPI, and must contain a KPI class. This class represents a generic indicator, and must carry some descriptive attributes, like the description of the KPI and the algorithm used in the computation. Moreover, the KPI class is associated with eight subclasses representing the categories in which the KPIs are classified: KPIEconomics, KPIPassenger, KPIDisruption, KPIRecovery, KPIScoring, KPISocialNetwork, KPIService and KPITickets. In order to represent specific KPIs, e.g., *Total number of tickets*, and their evaluations, we explored two possible alternatives.

The first alternative envisages the representation of the specific KPIs as subclasses of the classes representing the KPI categories. Each subclass features the input parameters of the specific KPI as attributes. A KPI evaluation is then an instance of the class representing the KPI being evaluated. Adopting such a representation, the ontology would represent a KPI *Total number of tickets*, accepting as input parameters start and end dates, through a class TotalNumberOfTickets that has StartDate and EndDate as attributes and is a subclass of KPITickets. The evaluation of the total number of tickets between 1st and 30th April 2016 would be an instance of the class TotalNumberOfTickets, with “1st April 2016” and “30th April 2016” assigned as values to the attributes StartDate and EndDate.

The second possible solution, on the contrary, represents the specific KPIs as instances of the KPI subclass representing the appropriate category. The KPI evaluations are instances of another class, KPIEvaluation, and are associated with the instances describing the KPIs being evaluated through properties. Other properties associate the KPIs with their input parameters, and the KPI evaluations with values for the input parameters. In this scenario, TotalNumberOfTickets is an instance of the KPITickets class, and is connected through properties with the input parameters StartDate and EndDate. The evaluation of the total number of tickets between 1st and 30th April 2016 is an instance of KPIEvaluation, connected through properties both to TotalNumberOfTickets and to the values of start and end date.

The main advantage of the first solution relies on its simplicity: having a class for each KPI allows us to represent the input parameters as attributes of the class, without introducing further properties. However, to be rigorous, each instance of KPI, i.e., each evaluation, should instantiate the descriptive attributes of the KPI to be evaluated, like the algorithm and the description. This creates redundancy, since these attributes have equal values for each evaluation of the same KPI. Therefore, we decided to opt for the second solution, which is a little more complex but avoid these redundancies. In addition, having KPIs represented as instances fits better with the extendibility requirement of the KPI list.

3 DESCRIPTION OF THE CONCEPTS

The following Capella class diagram represents the concepts of the BA ontology and their relationships, formalised as properties.

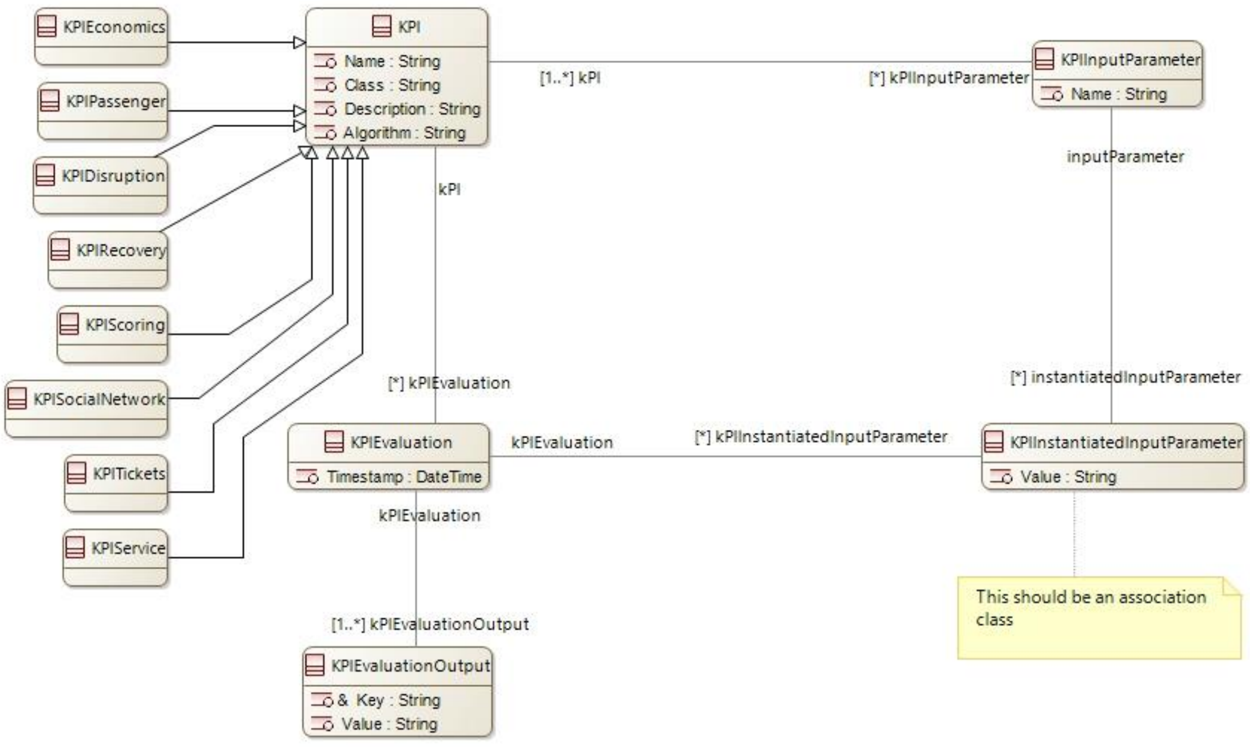

Figure 1 Capella class diagram representing the concepts of the BA ontology (Model Reference: [CDB]L BA KPI, SVN version 823)

A description of the employed concepts follows. The concepts are exemplified using the *Total number of tickets* indicator.

KPI

A KPI is an indicator measuring the performance of an organisation on a specific task. Besides the *total number of tickets*, other examples of KPIs of interest within IT²RAIL are *the ticket revenue per travel leg*, the *number of passengers per travel leg*, and the *total number of disruptions*. Each KPI features a set of attributes:

- Name: the name of the indicator, e.g., “Total number of tickets”.
- Description: a brief explanation of the indicator, e.g. “Total number of issued tickets in the time period”.

- Algorithm: the algorithm used to compute the KPI, e.g. “Collect all records from *Ticket* that are within the time interval specified by *StartTime – EndTime*; ∑ *Entries (tickets)* retrieved in the previous step”.

Each KPI is connected through properties to its input parameters. In the example, *Total number of tickets* is connected to two input parameters: *StartDate* and *EndDate*

The KPI class is characterised by eight subclasses representing the possible categories to which the indicators may belong:

- **KPIEconomics:** indicators denoting revenues related to the selling and purchase of tickets (e.g., *ticket revenue per travel leg*).
- **KPIPassenger**: indicators dealing with travelers/travels (e.g., *number of passengers per travel leg*).
- **KPIDisruption**: indicators denoting how particular events, like for instance strikes, affect travels (e.g., *total number of disruptions*).
- **KPIRecovery**: indicators denoting the needed time for recovery after a service disruption (e.g., *average service disruption recovery time*).
- **KPIScoring:** indicators dealing with the scores assigned to services by travelers (e.g., *travel leg service mean score*).
- **KPISocialNetwork**: indicators evaluating the feedbacks sent by users through social networks (e.g., *sentiment analysis score from social networks*).
- **KPIService**: indicators regarding the shopping carts (e.g., *total number of shopping carts*).
- **KPITickets:** indicators denoting tickets sold by Transport Service Providers (e.g., *total number of tickets*).

KPIInputParameter

It represents a parameter required for the computation of a KPI.

Each input parameter has just an attribute: its name.

In our example, as mentioned above, there are two input parameters: *StartDate* and *EndDate*.

KPIInstantiatedInputParameter

It represents the value associated with an input parameter in a specific evaluation of a KPI.

Each instantiated input parameter has just an attribute: the value that it assumes.

Each instantiated input parameter is connected through a property to the input parameter of which it is the instantiation.

In our example, we might have an instantiated input parameter *InstantiatedStartDate1* assigning the value "1st April 2016" to the input parameter *StartDate*. In addition, *InstantiatedEndDate1* assigns the value "30th April 2016" to the input parameter *EndDate*.

KPIEvaluationOutput

It represents an output produced by a KPI computation; the output of a KPI computation in general is constituted by a set of instances of KPIEvaluationOutput, that are (key, value) pairs.

Each KPI evaluation output has two attributes: the key and the value.

In our example, the output of an evaluation of *Total number of tickets* is represented by a unique instance of KPIEvaluationOutput. The possible output of a computation may be the instance *KPIEvaluationOutput1*, specifying a number of tickets as its value, e.g. 20000.

Note that our example KPI envisages just one output value, but other indicators may require a set of (key, value) pairs. A KPI *Total number of tickets per month*, for instance, might require a year as input parameter and return a set of (month, number of tickets) pairs as output.

KPIEvaluation

It is the specific computation of a KPI, executed in a given time instant. It is associated with the values assigned to the input parameters and with the output values.

Each KPI evaluation has just an attribute: the timestamp in which the computation was executed.

Each KPI evaluation is connected to the evaluated KPI, with the values of the input parameters, and with the output.

In our example, we might have an instance *KPIEvaluation1* representing the evaluation of the KPI *Total number of tickets* executed at time instant “2016-05-01 14:30.155” specifying as instantiated input parameters *InstantiatedStartDate1* and *InstantiatedEndDate1*, and producing as output *KPIEvaluationOutput1*.

End-of-document