Automating the process of building flexible Web Warehouses with BPM Systems

Andrea Delgado, Adriana Marotta
Instituto de Computación, Facultad de Ingeniería
Universidad de la República
Montevideo, Uruguay
{adelgado, amarotta}@fing.edu.uy

***Abstract*—The process of building Data Warehouses (DW) is well known with well defined stages but at the same time, mostly carried out manually by IT people in conjunction with business people. Web Warehouses (WW) are DW whose data sources are taken from the web. We define a flexible WW, which can be configured accordingly to different domains, through the selection of the web sources and the definition of data processing characteristics. A Business Process Management (BPM) System allows modeling and executing Business Processes (BPs) providing support for the automation of processes. To support the process of building flexible WW we propose a two BPs level: a configuration process to support the selection of web sources and the definition of schemas and mappings, and a feeding process which takes the defined configuration and loads the data into the WW. In this paper we present a proof of concept of both processes, with focus on the configuration process and the defined data.**

Keywords—Data Warehouses (DW), Web Warehouses (WW), building process, Business Process Management Systems (BPMS)

I. INTRODUCTION

In last years the amount of data generated on the web has grown considerably, being one of the current key challenges to be able to effectively collect and analyze it, in order to gain usable information. In this context, Web Warehouses (WW) are Data Warehouses (DW) whose data sources are taken from the web. They are a valuable tool for analysis and decision making based for example, on open government data [1], in many different areas. We define a flexible WW, which can be configured accordingly to different domains, through the selection of the corresponding web sources and the definition of data processing characteristics. This WW is implemented through a general platform that is configured for each domain, or particular case it will be used. The general platform is built according to an architecture that is suitable for different WWs [2]. Although the process of building DW is well known with well defined stages, it is still mostly carried out manually by IT people in conjunction with business people.

Business Process Management Systems (BPMS) [3] provide tool support for the Business Process Management (BPM) [3][4][5] vision in organizations. This kind of software integrates different tools to support the complete BPs lifecycle [3], mainly modeling BPs and executing them in a compatible process engine. One of the many benefits associated with this vision is that BP models provide an explicitly view on which activities are performed to reach the organization goals, how things are done by whom, when and using and generating which artifacts. Nowadays, one of the most used notations for modeling and executing BPs is the Business Process Model and Notation (BPMN 2.0) [6] standard from OMG, which provides not only a notation for modeling BPs but also a defined semantic for its elements, making it also executable. Also, it is easily understandable by business people and it has been embraced both at the academic and industry level as a way to communicate between business and IT people [7].

Based on the previous analysis, we propose a two Business Process (BPs) level vision to help define and automate the process of building flexible WW: at the first level a *configuration process* to support the selection of web sources and the definition of schemas and mappings, which is mostly carried out manually, and at the second level, a *feeding process* which takes the defined configuration and loads the data into the WW, which is performed mostly automatically. Both the configuration and the feeding processes are modeled as BPs in the BPMN 2.0 notation and executed in the BPMS Activiti [8]. We have selected it mainly because it implements the BPMN 2.0 standard and it is open source. In previous work [9] we have presented the initial idea of building a Quality-aware WW with BPs in BPMN 2.0, and since then we have split the proposal into two projects mainly corresponding to the two process defined: the first one to define, model and execute the complete configuration process, which we have worked on last year, and the second one to define, model and execute the feeding process based on configuration data, adding quality data to both processes, in which we are working now. Data quality is taken into account when building the WW, to obtain information about the quality of data provided to the user, and to improve quality of data throughout the WW process.

In this paper we present the complete configuration process along with the analysis and definitions we have taken, to provide automated support to the configuration BP and data that will be used by the feeding process in order to effectively load the WW. We also present a proof of concept of the Feeding process to validate the configuration BP. The rest of the document is organized as follows: in section II we present some concepts regarding WW, BPs and BPMS, in section III we describe in detail the BPs we have defined to support the WW building process, in section IV we present a case study with open data to illustrate the application of our proposal, in section V we discuss related work and finally in section VI we present some conclusions and current and future work.

This work has been partially funded by Comisión Sectorial de Investigación Científica (CSIC) from Universidad de la República, Montevideo, Uruguay.

II. WW and Business Processes

In this section we introduce concepts regarding WW and BPs that are used throughout the proposal.

A. *Processes to build flexible WWs*

The WW we define is a system that consists of several components and is service oriented. Fig. 1 shows its architecture. The system components are in charge of the main tasks that are necessary to adequately process web data and provide them to the user for analysis. These tasks are always present in this kind of systems, but there are many aspects of them that should be solved and implemented in different ways, according to the needs of the particular case. An effective mechanism for encapsulating the solution to the different aspects, is through the utilization of specialized services. The utilization of services facilitates the configuration of the system, contributing to the flexibility of its building.

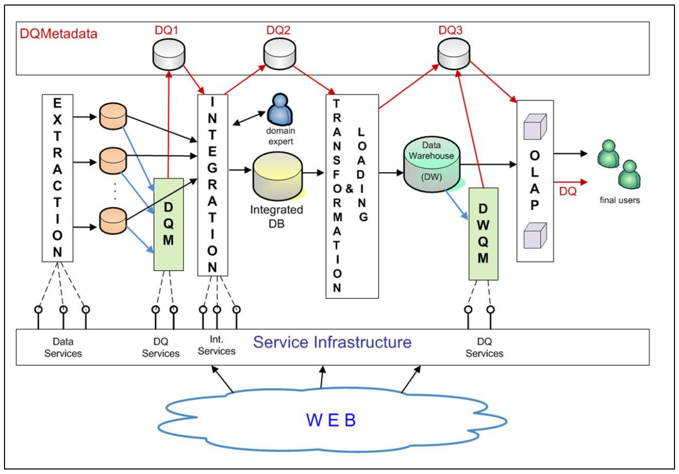


Fig. 1. Quality-Aware WW General Architecture

The system components are: Extraction, Integration, Transformation & Loading, OLAP (On-Line Analytical Processing), Data Quality Measurement (DQM) and DW Quality Measurement (DWQM).

The Service Infrastructure layer offers the different available specialized services, which include data services, data quality services and integration services.

The Extraction component must solve the problem of format heterogeneity, since data can be found in a variety of formats, such as *csv*, *html*, *xml*, *rdf*, etc. It must be able to extract from each source a pre-defined set of items and store them into a database, assuring that each data item goes to the right place. The Integration component must perform the integration of data coming from the different web sources, solving the problems of entity resolution and data conflicts, with the presence of semantic heterogeneity. The Transformation & Loading component transforms data to the multidimensional model, preparing them to OLAP manipulation. The OLAP component manages the data cubes allowing the multidimensional data analysis through front-end utilities.

DQM component is in charge of measuring data quality of data that have been just extracted from the web, while DWQM component is in charge of measuring data quality of the DW. These components register data quality information in the databases DQ1, DQ2, and DQ3, which contain quality information about the WW data (DQMetadata).

The described components base their solutions on the invocation of the services offered by the Service Infrastructure layer.

DQMetadata is read by the WW processes in order to take quality issues into account when treating data. On the other hand, it is written by them for enabling the propagation of the quality metadata throughout the different stages of the data transformation. This is necessary in order to maintain the correspondence between quality information and WW data, when WW data is combined and transformed.

As previously explained, the flexible WW we propose consists of a configurable platform, such that, in order to build a specific WW, two phases must be carried out: the configuration phase and the feeding phase.

In the configuration phase different activities are applied in order to configure each WW component. The main configuration activities for the Extraction component are: (1) selecting domain and web sources, (2) defining the sources schemas, (3) defining data mappings, and (4) selecting data services for sources data extraction. For the Integration component, they are: (1) integrated schema definition, (2) schema mappings specifications, and (3) selecting integration services. The Transformation & Loading component configuration consists in the implementation of the transformation process, while OLAP component configuration is done by defining the OLAP cubes. Finally, the main configuration activities for DQM and DWQM components are: (1) defining data quality model for extracted data, (2) defining data quality model for DW data, (3) defining data quality metadata, and (4) selecting data quality services.

In the feeding phase the WW configured components will automatically load and build the specific WW, executing the corresponding extraction, integration and transformation processes, managing data quality throughout the building process.

B. *BPs, BPMN 2.0 and BPM Systems*

BPs provide an horizontal vision of the organization, showing explicitly the sequence of activities and different options of execution that are performed to provide value to the business in an organizational and technical environment [3]. BPM [3][4][5] provides the means to support this horizontal vision based on the BPs lifecycle [3] and different tools integrated in BPMS, to help carrying out the activities that are defined within each phase. The core of these BPMS is a process engine, which allows executing the BP model driving the execution of the system, in which are known as Process Aware Information Systems (PAIS) [10]. Since the definition of workflows in the nineties decade and web services in early two thousands, the definition and execution of processes has been carried out in languages such as XPDL[11] and WS-BPEL[12], among other proposals. BPMN 2.0 allows not only modeling BPs with a notation that is understood by business people, but also executing the same model, helping to minimize errors from requirements to development, and helping business and IT people to work together. BPMN 2.0 defines elements that are needed to model BPs, including activities (sub-processes and tasks), gateways (AND, XOR, OR), events (time, message, etc.), swimlanes (pools and lanes), data, connectors, among others.

Regarding BPs execution in BPMS, the process engine that executes BP models can be seen as an intermediate layer between the user interface (mostly a Web Portal) and the implementation of the BP (mostly in Java with elements from the Java world). The process engine then controls, at any given time, which activity is done by whom, presenting the corresponding elements to the user (managing the so called working list) and/or performing the invocations to classes, following the defined BP's control flow. In the case of user tasks, the execution of the process engine will send it and present it to the inbox associated with the role defined (or user depending on the assignment e.g. algorithms for load balancing or other criteria). In the case of automated tasks the execution of the process engine will perform the invocations to the classes, external systems, Web Services (WS) or other defined components, such as a business rules engine. In Fig. 2 we present examples for different execution scenarios in a process engine: user task, service task and business rule task.

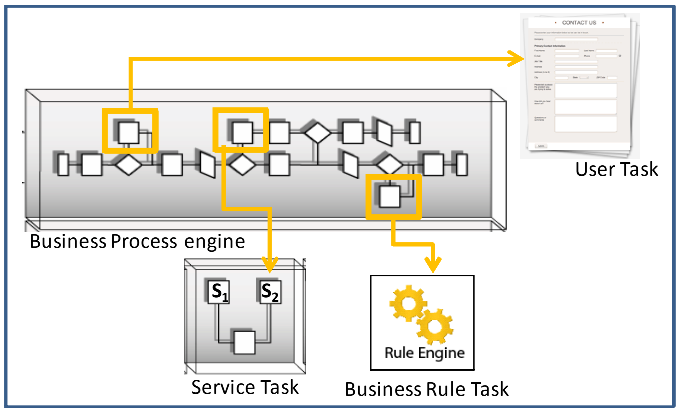


Fig. 2 Different types of execution in a BPMS

III. AUTOMATED SUPPORT FOR THE WW CONFIGURATION

The configuration BP is a key step in building a flexible WW, in which the user defines all the data that will be used in the feeding process to actually generate the WW. The configuration data is registered by the system in our own data model in which we have defined key elements we need to further generate the WW.

A. *WW configuration process*

The aim of the configuration process of the WW is to gather from the user and register in the configuration data base the data that will allow to carry out the ETL for the DW in the feeding process. Then, defining the WW configuration involves things such as selecting the web sources from which to gather the data, defining the schemas that will be used (expected schema, integrated schema and data warehouse schema), as well as the mappings between their elements that are needed to guide the extraction, selection and integration of data in each step of the feeding process. The WW configuration process we have defined is mainly focused on supporting the definition of schemas and correspondences between their elements as shown in Fig. 3.

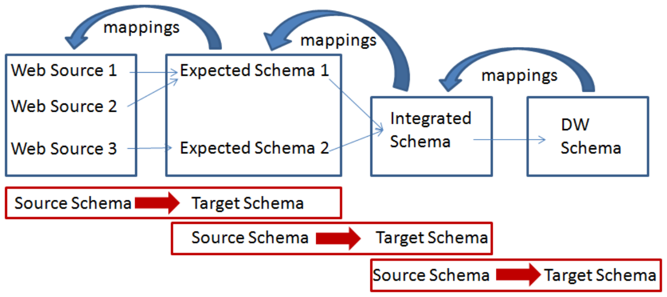


Fig. 3 Definition of schemas and their correspondence

We define a conceptual sub-process that will be used throughout the configuration BP for each schema creation and corresponding mappings, in which in the first place, the user is asked to define the target schema, and then to define the corresponding mappings to the origin schema (or file in the case of the web sources). Fig. 4 shows graphically this conceptual sub-process.

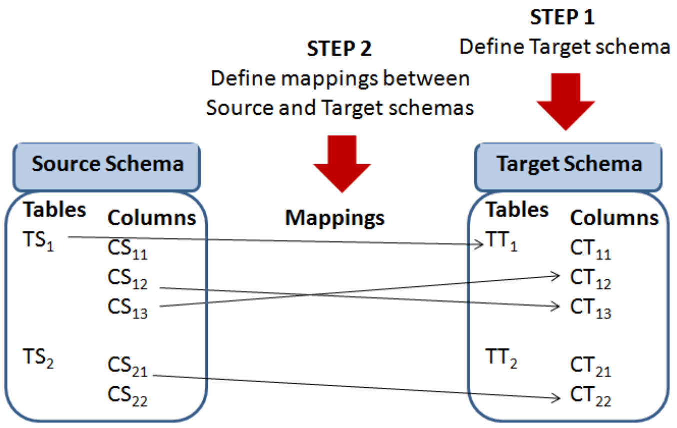


Fig. 4. Conceptual process to define schemas and mappings

Fig. 5 presents the WW configuration process specified in BPMN 2.0 and implemented in the Activiti BPMS. It defines two lanes one for the Domain Expert and the other for the IT professional, which are responsible for the activities defined. The first three tasks correspond to: 1) "Select Domain" for the selection of the WW domain, 2) "Select Web Source" for the selection of existing web sources already registered for the selected domain, and 3) "Create Web Source" for the creation and register of new web sources for the selected domain. Some key domains and corresponding web sources will be already registered in the configuration database for the users to select. If when defining the configuration for a WW the user considerers that the existing web sources are enough, indicating so in the corresponding task 2), the creation of new web sources for the domain will be skipped.

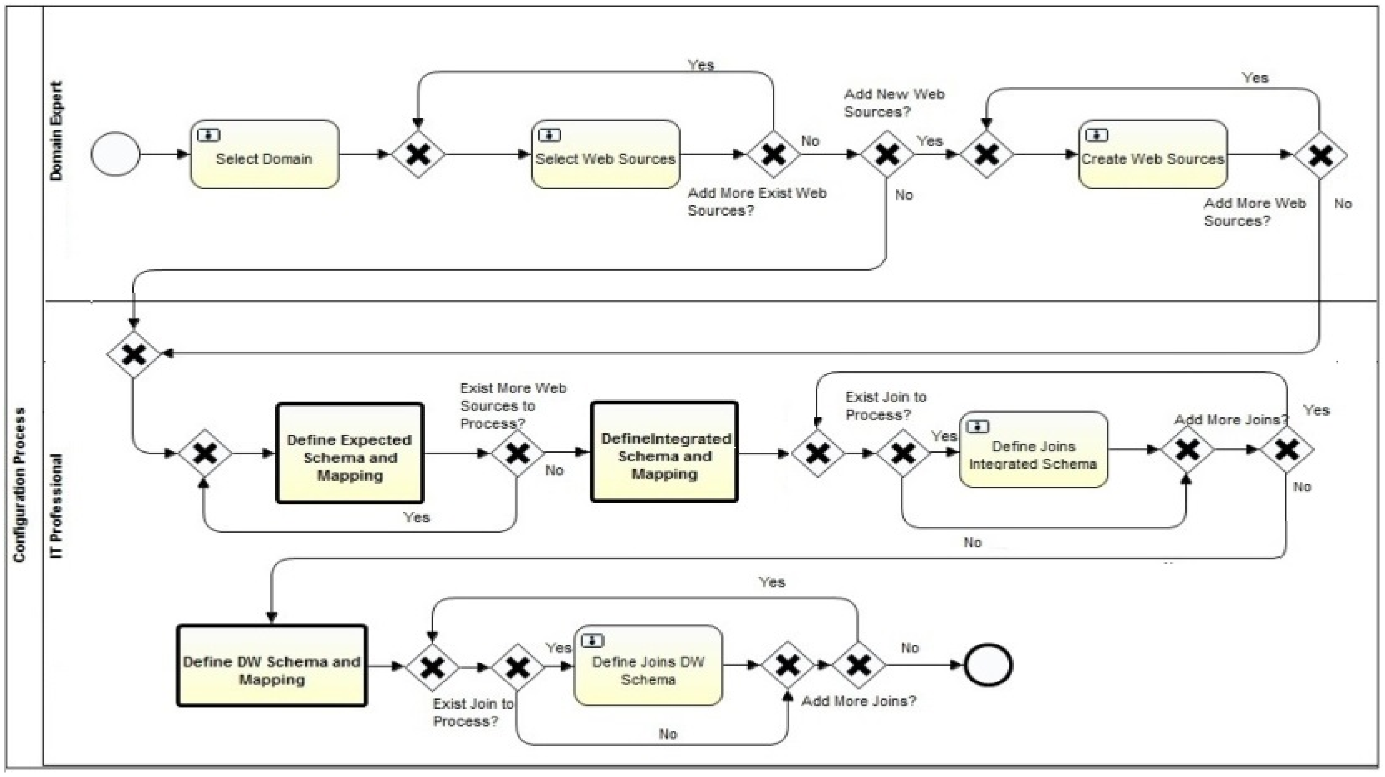


Fig. 5. WW configuration process specified in BPMN 2.0 and implemented in Activiti BPMS

Once these tasks are finished, the system has registered the WW domain and the associated web sources that will provide the data for the WW. The rest of the configuration process will define the way in which the data will be extracted from the web sources, the different schemas that will be defined and used within the process steps, along with the corresponding mappings to select and/or integrate the data.

The first definition of schemas and mappings is encapsulated in the sub-process 4) "Define expected Schema and mapping", which follows the conceptual process as shown in Fig. 4. In this sub-process, for each selected web source the expected schema into which the extracted data will be loaded is defined, along with the corresponding mappings between fields in the web source file and attributes in the expected schema. This sub-process is shown in Fig. 6 (a), where three options to define and use an expected schema are provided. In the first place, an existing expected schema can be selected, the second option allows the user to create it automatically uploading a JSON file containing the definition of the schema, and finally, creating the schema from scratch manually by the user. To do so, data regarding the table name, attribute name and type, for each element defined has to be entered. To continue the loop the flag "Add More Attributes" can be marked, until there are no more attributes and/or tables to define.

In the three options, the expected schema that is created is associated with the selected web source for the current configuration. In the first and second options, the automated task "Process expected schema" is executed, to perform the association or to parse the JSON file to create the expected schema and the association, in the third case the user must define the expected schema in a loop that can be executed until the user indicates to finish the creation. In any case the expected schema must have a name, attributes and their types.

Following the conceptual process shown in Fig. 4, the next step is to define the mappings between the expected schema and the selected web source. To do so, two options are provided: if the web source has a format that is known by the system (e.g. HTML, CSV) the user is presented with the existing column names in the file, if the format is not known by the system, then the user has to manually register each column of the web source file. In any case, the correspondence between columns in the expected schema and columns in the web source has to be defined manually by the user in the task "Mapping expected schema and web source". At the end of the sub-process, the task "Upload Data service" allows the user to define the Data Service (DS) in charge of actually extracting the data from the selected web source in the feeding process. The user form will show a default DS (registered in the system for the format of the selected web source) and the user will be able to change it by selecting another URL for the DS.

The next definition of schemas and mappings correspond to the integrated schema to join the data from the web sources, which is encapsulated in the sub-process 5) "Define integrated schema and mapping" and shown in Fig. 6 (b). To do so, the user has to create each table and the corresponding attributes, which again can be done automatically by uploading a JSON with the schema or manually defining each table and attributes in a loop, until the user selects to finish the creation. For each table the primary keys are marked, generating an initial definition of the Integrated Schema, which is then shown to the user and afterwards with this information, the foreign keys can be marked for each table to finish the creation of the schema.

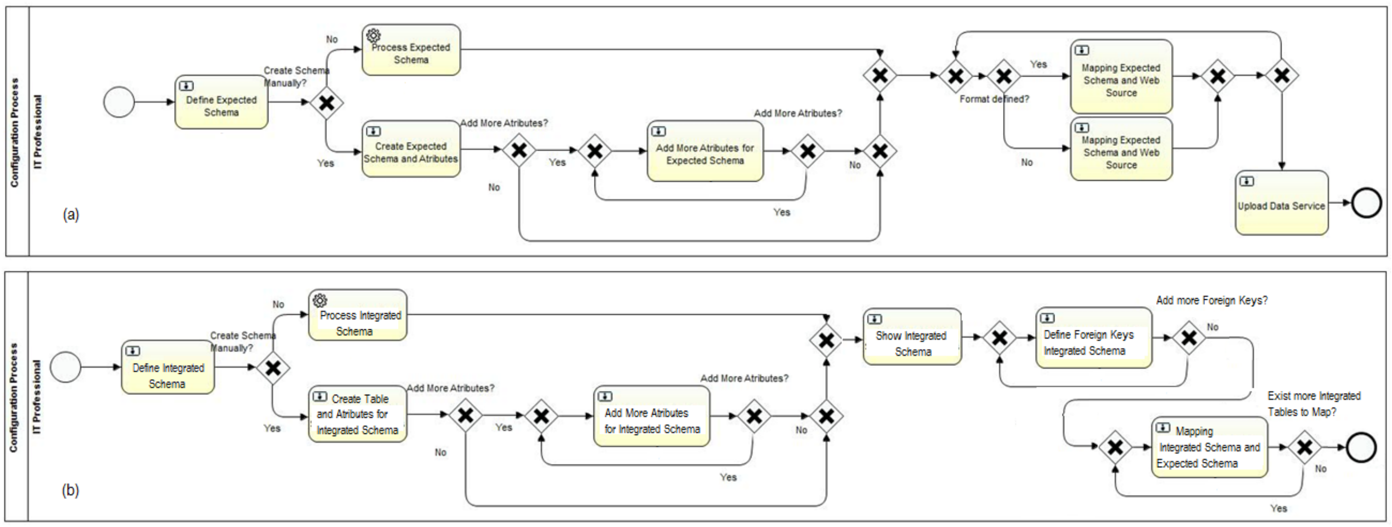

Configuration Process
IT Professional
Define Expected Schema
Create Schema Manually?
No
Yes
Process Expected Schema
Create Expected Schema and Atributes
Add More Atributes?
Add More Atributes for Expected Schema
Format defined?
Mapping Expected Schema and Web Source
Upload Data Service
(a)
Define Integrated Schema
Process Integrated Schema
Create Table and Atributes for Integrated Schema
Add More Atributes for Integrated Schema
Show Integrated Schema
Define Foreign Keys Integrated Schema
Add more Foreign Keys?
Exist more Integrated Tables to Map?
Mapping Integrated Schema and Expected Schema
(b)

Fig. 6. Sub-process to define: (a) expected schemas for each web source and corresponding mappings, (b) integrated schema

At this point we have defined the web sources from where to extract the data which will be placed into the Expected Schemas and then will be joined into the Integrated Schema. To do so and following the conceptual process as presented in Fig. 4, to end with these definitions, the user has to define the mappings between the Expected Schema for each web source and the Integrated Schema that he/she has just defined. To define the mappings the system will show the user with a form in which for each table from the Integrated Schema, it will present each time an existing attribute which the user should associate with an attribute from any of the defined Expected Schemas. This is done in a loop until all attributes of all tables in the Integrated Schema has been mapped. In this case other information has to be gathered from the user, regarding which joins are needed between which tables in the defined Expected Schemas, to be able to load the data into the corresponding ones in the Integrated Schema in the feeding process. The task 6) "Define joins Integrated Schema" allows the user to define t his information.

The final step involves the definition of the DW Schema, from the tables defined in the Integrated Schema. This is done in the sub-process 7) "Define DW Schema and mapping" which also follows the conceptual process as presented in Fig. 4 and is analogous to the definition of the Integrated Schema shown in Fig. 6 (b). In this case, apart from defining each table and the corresponding attributes (which again can be done automatically by uploading a JSON with the schema or manually defining each table and attributes in a loop), the user has to mark fact tables, and the other will be dimensions. After that, for each table the primary keys are marked, generating an initial definition of the DW Schema, which is then shown to the user and afterwards with this information, the foreign keys can be marked for each fact table to finish the creation of the DW schema. In this case other information is also required, regarding the joins that are needed between the tables in the defined Integrated Schema to be able to load the data into the corresponding ones in the DW Schema in the feeding process. The task 8) "Define joins DW Schema" allows the user to define this information, as in the previous definition.

The configuration data that is gathered from the user along the configuration process is stored in the Configuration Database in our own defined data model, to be read during the feeding process to guide it. In Fig. 7 we present some of the key tables, attributes and relationships we have defined in our configuration data model. At the center of the definitions there is a table named "Configuration" to hold the data for each configuration, with an id and the domain selected by the user. Key defined tables refer to the source and target schemas en each moment of the configuration BP (i.e. web sources, expected schemas, integrated schema and DW schema), and the mappings between them that will allow to automate the load the corresponding data in the feeding process.

As an example of each triple defined for the schema and mappings (source schema, target schema and mapping), in Fig. 7 it can be seen that the resulting DW Schema is registered in the Table "dw_schema" which has as attributes the name for the Schema, the reference to the configuration and the name of each table defined, with the mark of fact table or not. The table "atributos_dw_schema" defines each attribute within each table in the schema, referring the previous table primary key, and adding the name of each attribute defined in each table. This pair of tables models the target schema in this last triple. The source table corresponds to the Integrated Schema and is defined in the same way in the "integrated_schema" and "atributos_integrated_schema" tables.

Finally, the mapping is stored in the table "mapping_dw" which refers to each attribute tables (source and target) in the corresponding configuration. As model in the configuration BP following the conceptual process presented in Fig. 4, mappings are defined at the attribute level between each attribute of the target schema to attributes in the source schema, until all have been defined. This same design applies for the mapping regarding the triple Integrated Schema, Expected Schemas and mapping between them, and similarly (adding some extra needed elements) regarding the triple Expected Schema, Web Sources and mapping between them.

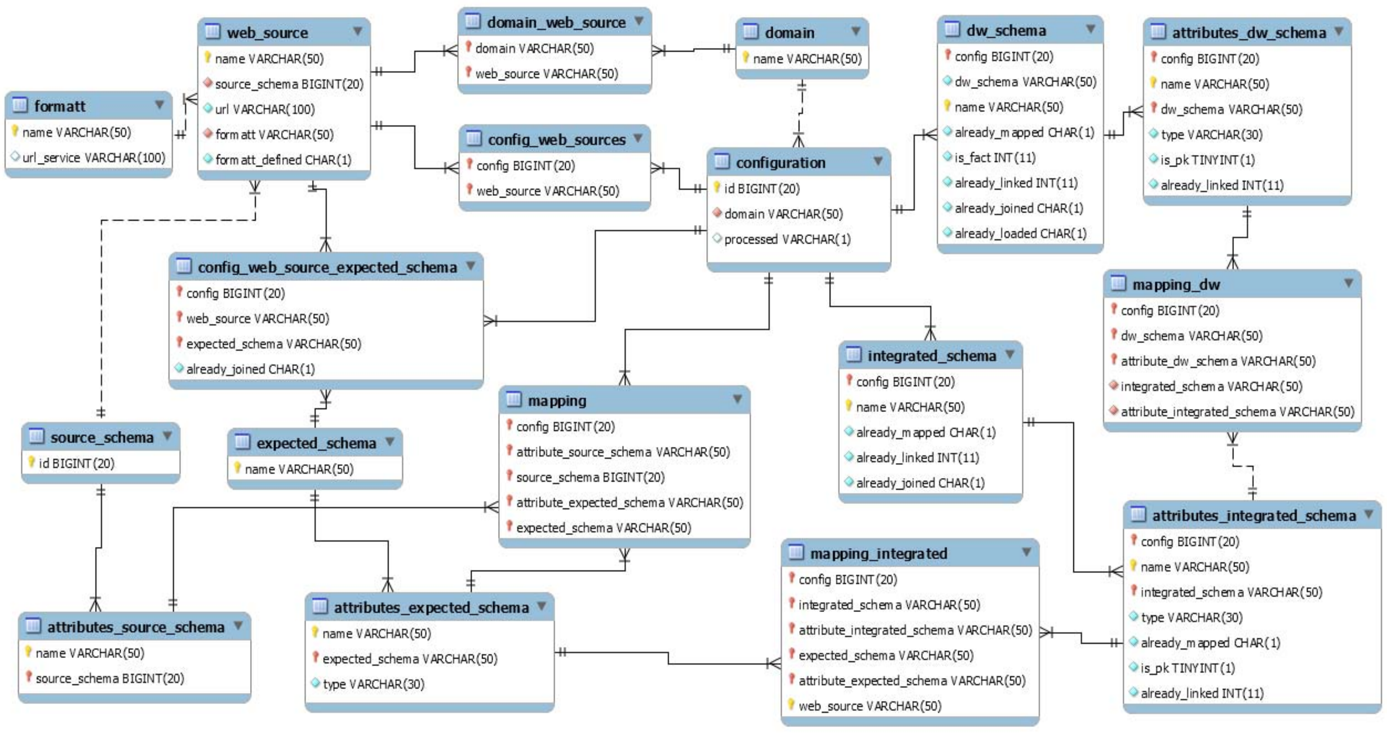


Fig. 7 Key tables defined in the configuration data model

B. *WW feeding process using configuration data*

Once the configuration process has finished, all the relevant configuration data is stored in the Configuration Database. At this stage, the WW is ready to be generated and fed. In the feeding process each component is executed as shown in Fig. 1, but the first step each one executes is the query to the Configuration Database. Fig. 8 shows the examples of the Extraction and the Data Quality Measurement (DQM) processes.

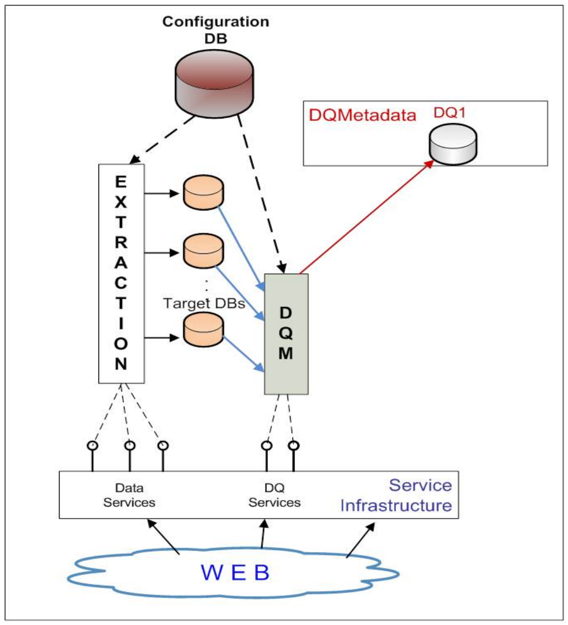


Fig. 8. WW Feeding – Extraction and DQM example

In the case of the Extraction process, it extracts from the Configuration Database the data sources URLs, the identification of the data services that extract data from these URLs, and the expected schema that has to be filled and stored in the target databases. In the case of the DQM process, it queries the Configuration Database in order to obtain the data quality model (quality dimensions and metrics to be applied), the data quality services to invoke, and the data quality metadata identification. The quality data is not yet integrated into the configuration process. Analogously, the other components obtain the needed information from the Configuration Database and execute the corresponding data processing. However, the feeding process is not yet completely defined as there are many data integration issues that need to be addressed to complete its automation.

Fig. 9 shows the proof of concept of the Feeding process we have developed in order to validate the configuration BP and data defined. It can be seen that differently to the configuration BP, most tasks in this process are automatic of type service task, and only a few remain of type user task. This is due to the focus of the process in reading the already registered configuration data, create the needed schemas and perform the Extraction, Transformation and Load (ETL) processes in order to automatically generate the DW. Most of the user tasks are defined to present information to the user in case of integration problems, execution problems and to approve the generation of the final DW schema and data load.

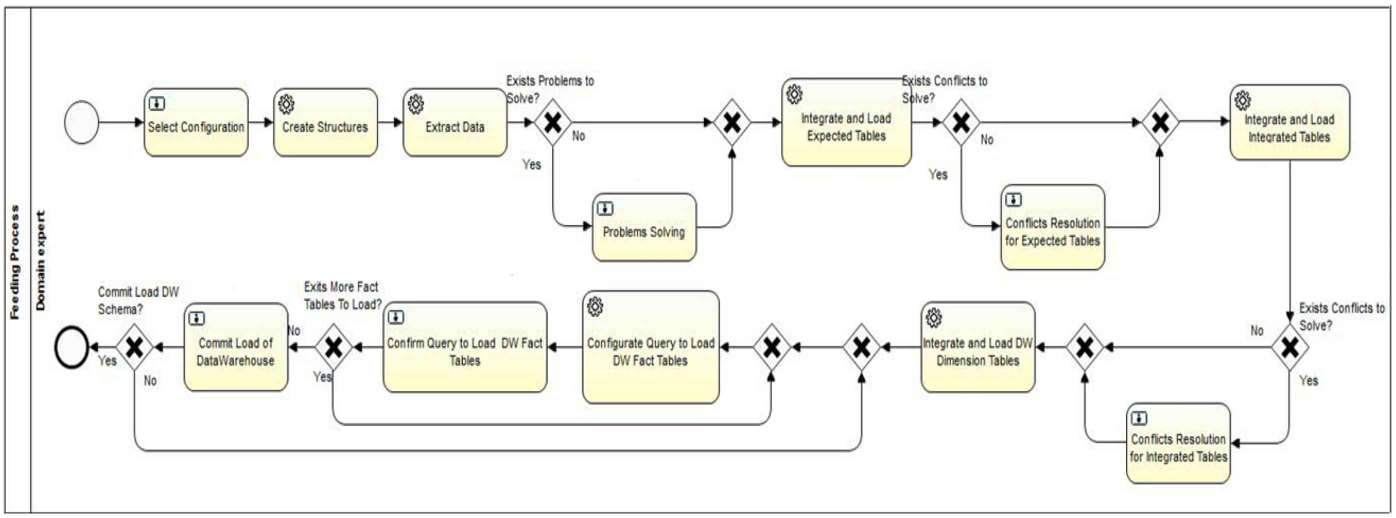

Feeding Process
Domain expert
Select Configuration
Create Structures
Extract Data
Exists Problems to Solve?
No
Yes
Problems Solving
Integrate and Load Expected Tables
Exists Conflicts to Solve?
No
Yes
Conflicts Resolution for Expected Tables
Integrate and Load Integrated Tables
Exists Conflicts to Solve?
No
Yes
Conflicts Resolution for Integrated Tables
Integrate and Load DW Dimension Tables
Configurate Query to Load DW Fact Tables
Confirm Query to Load DW Fact Tables
Exits More Fact Tables To Load?
No
Yes
Commit Load of DataWarehouse
Commit Load DW Schema?
Yes
No

Fig. 9. Proof of concept of the Feeding process to validate the configuration BP and data defined

IV. Case study with open data

There is a considerable amount of information published as open government data in the Tourism domain, in Uruguay. Several datasets that contribute to touristic information, published by different organisms, can be found. In this case we generate a WW where data about events in different places of the country are analyzed together with data about accommodation and transport useful for assistants to the events. The analysis may be done by dates. The analysis indicators are the quantity of events and the quantity of accommodation places. Fig. 10 presents the conceptual multidimensional model for this analysis, following the model of [13]. As most of the available data is in Spanish we will present the equivalence between the concepts from the design of the WW presented here and the ones actually used in the execution of the processes in next subsections.

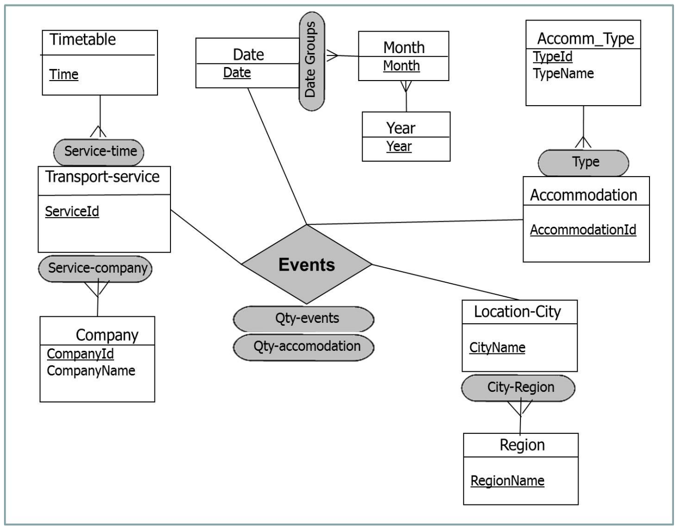


Fig. 10. Conceptual multidimensional model for Events fact.

The dimensions in this DW are: (i) Transport-service, with two hierarchies (Service-time and Service-company), each one containing two levels, (ii) Accommodation, with one hierarchy (Type), containing two levels, (iii) Location-city, with one hierarchy (City-Region) of two levels, and (iv) Date, with one hierarchy (Date groups) which contains three levels. The DW measures are Qty-events and Qty-accommodation. Fig. 11 presents the logical schema of the target DW.

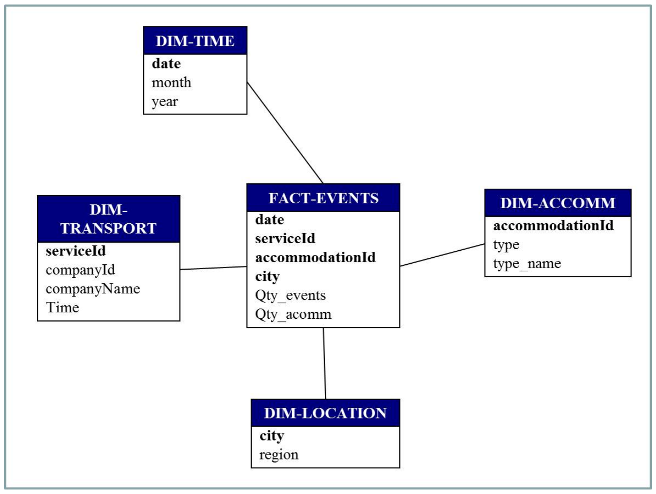


Fig. 11. Logical DW schema.

The main data sources are open data publications from public organisms, although some data (proportionally very few) had to be generated by us. Accommodation data are obtained from publications of the Government Agency of Tourism and Sport (original name: “Ministerio de Turismo y Deporte”) [14], transport data from publications of the National Transport Unit (original name: “Dirección Nacional de Transporte”) [15], some data about events are obtained from publications of Municipality of Durazno City (original name: “Intendencia de Durazno”) [16].

We describe in detail the Accommodation data processing starting from the web sources. The selected Accommodation data was about Hotels, Hostels and Campings. These three datasets are published in three different web pages of the same site. The data published in the three cases are: identifier, name, address, city, region, phone, cellular phone, web, mail, coordinates. In the case of Hotels they also publish a category data item.

For our WW we define the expected schema shown in Fig. 12 (a). This is the expected schema for the three sources, i.e. it specifies the data we require from each of these sources. Mappings between the expected schema attributes and the web sources items are defined during the configuration phase such that the loading from the sources to the target databases can be performed. For example, accommodationId is mapped to the identifier of the hotels source data, and analogously with the hostels and campings sources. The whole integrated schema is shown in Fig.12 (b). Note that in this schema, Accommodation table has a new attribute named type, which will contain one of the values: "hotel", "hostel" and "camping".

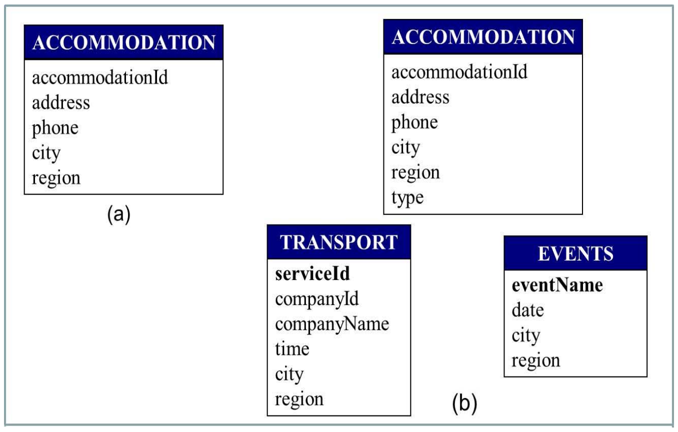


Fig. 12. (a) Expected schema for Accommodation data, (b) Logical Integrated schema.

Integrated schema attributes are mapped to expected schema attributes as shown in Fig. 13. In this case the attribute called type is not mapped. In the feeding phase the value for this attribute will be asked to the user.

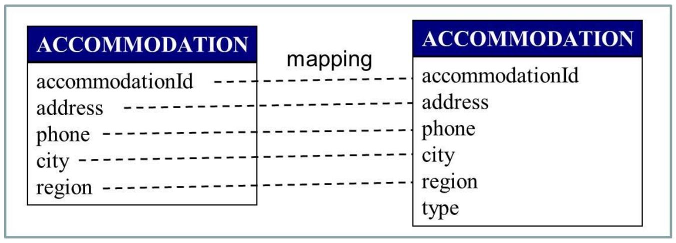


Fig. 13. Mappings between Integrated and Expected schema.

In addition, mappings between the Integrated schema and the DW schema must be performed. Fig. 14 shows only the mappings between the Accommodation table attributes and the DW attributes due to visualization restrictions. Fact-events attributes' mappings are as follows:

- date → Events.date,
- serviceId → Transport.serviceId,
- accommodationId → Accommodation.acommodationId,
- city → Accommodation.city,
- city → Events.city,
- city → Transport.city.

It is important to note that the city attribute of the DW schema is mapped to the three tables of the Integrated schema. This is especially important for the feeding phase, since the city determines which combinations of transport service, accommodation and events must be generated.

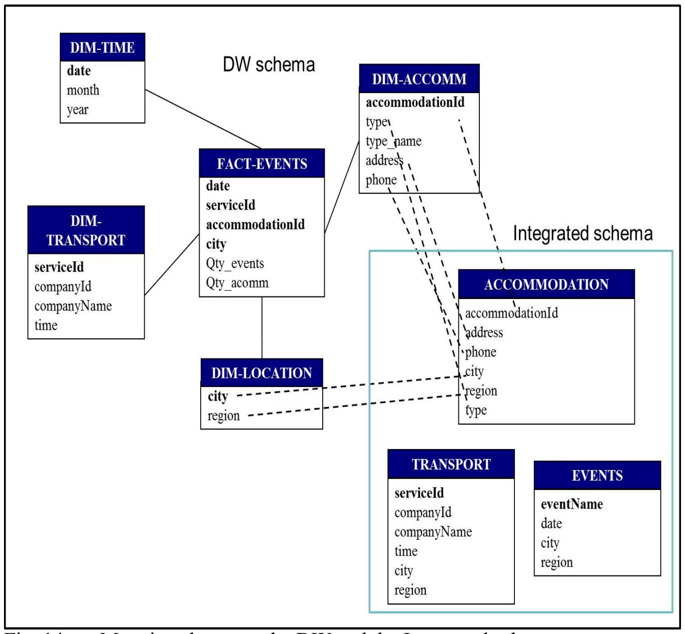
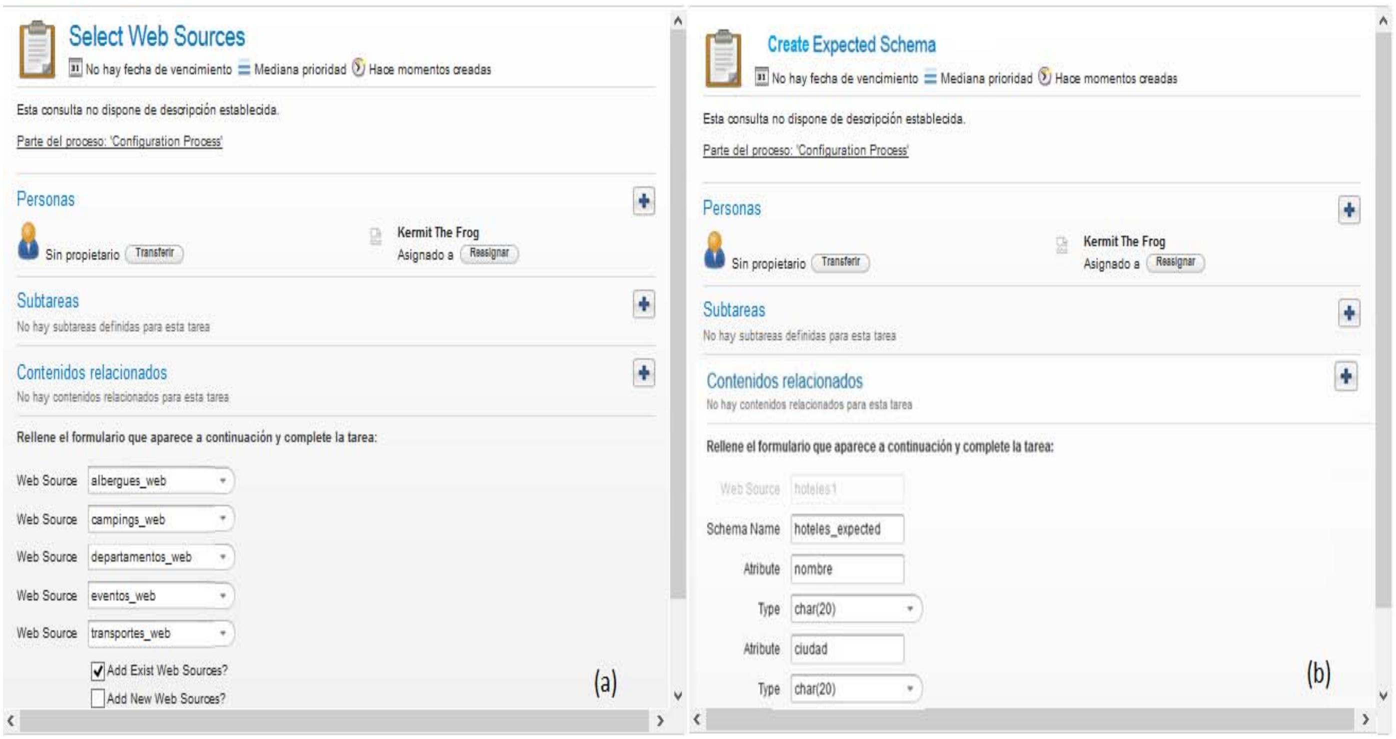


Fig. 14. Mappings between the DW and the Integrated schema.

A. *Execution of the WW Configuration process*

As an example of the execution of the WW Configuration process we have implemented, we present some screenshots of the forms defined to input the configuration data by the user. In the first place, after selecting the domain for the current configuration, the user selects the Web Sources that will be used to extract the data to feed the DW. In Fig. 15 (a) the corresponding Activiti form for the user task is presented, showing the possible selection of existing Web Sources for the seleected domain such as Accomodation (albergues_web, campings_web, hotels not shown), Events (eventos_web) and Transport (transport_web), and Regions (departamentos_web) which finally was not selected. As it can be seen, we present five Web Sources per form (due to Activiti limitations) and the option to continue selecting (which is not shown) to perform the loop and present again the form. We also provide the options for creating new and/or adding existing Web Sources.

After selecting the Web Sources, as defined in the configuration BP, the Data Sources are provided, the Expected Schemas and corresponding mappings to the Web Sources are defined. To create the Expected Schemas, a form is presented to the user to define the attributes and corresponding types, as shown in Fig. 15 (b). As it can be seen, for each Web Source an Expected Schema has to be created, from scratch or selecting already existing ones. In this case we show the selection of the Web Source "hoteles1" which will be mapped to the Expected Schema Accomodation (hoteles_expected) as presented in Fig. 12 (address = direccion, city = ciudad). We present in each form five attributes to be defined with their name and type, and the option to continue creating attributes


Select Web Sources

No hay fecha de vencimiento Mediana prioridad Hace momentos creadas

Esta consulta no dispone de descripción establecida.

Parte del proceso: 'Configuration Process'

Personas

Sin propietario Transferir

Kermit The Frog

Asignado a Reasignar

Subtareas

No hay subtareas definidas para esta tarea

Contenidos relacionados

No hay contenidos relacionados para esta tarea

Rellene el formulario que aparece a continuación y complete la tarea:

Web Source	albergues_web
Web Source	campings_web
Web Source	departamentos_web
Web Source	eventos_web
Web Source	transportes_web

[✓] Add Exist Web Sources?

[ ] Add New Web Sources?

(a)

Create Expected Schema

No hay fecha de vencimiento Mediana prioridad Hace momentos creadas

Esta consulta no dispone de descripción establecida.

Parte del proceso: 'Configuration Process'

Personas

Sin propietario Transferir

Kermit The Frog

Asignado a Reasignar

Subtareas

No hay subtareas definidas para esta tarea

Contenidos relacionados

No hay contenidos relacionados para esta tarea

Rellene el formulario que aparece a continuación y complete la tarea:

Web Source	hoteles1
Schema Name	hoteles_expected
Atribute	nombre
Type	char(20)
Atribute	ciudad
Type	char(20)

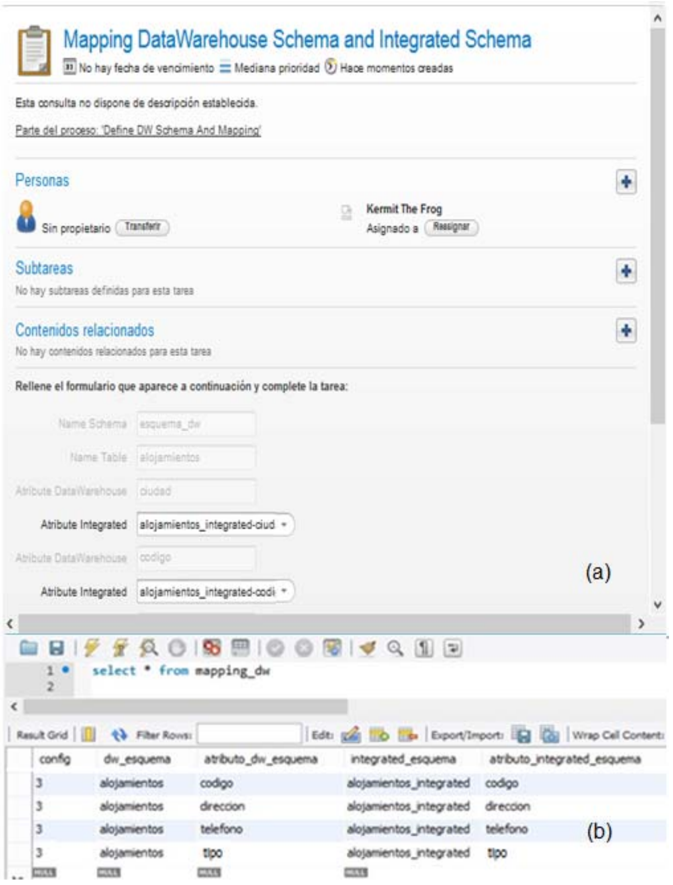
(b)

Fig. 15. Activiti forms screenshots: (a) Select Web Sources for the selected domain and (b) Create Expected Schema for each selected Web Source

which will activate the loop and present the form again, for the same Web Source until it is finished. In this screenshot the options to select existing schemas and to continue adding attributes are not shown, but they are the same as in the previous form. Once the expected schemas are created, the mappings with each Web Source are defined.

Then the Integrated Schema is defined in the same way, and after that, the corresponding mappings to the Expected Schemas defined are established, as presented in Fig. 13. Finally the DW Schema is defined, creating the tables and defining which are fact tables and which are dimension tables. To end the configuration process, the corresponding mappings from the DW to the Integrated Schema are defined. As all mappings are presented in the same way, in Fig. 16 we show the definition of mappings between the DW Schema and the Integrated Schema as an example, along with the data registered in the "mapping_dw" table after executing the task.

It can be seen in Fig. 16 (a) that for each Table in the DW Schema, we present all the attributes defined, which have to be mapped to attributes in the tables of the Integrated Schema. In this case, the DW table corresponds to the Accommodation (alojamientos) one with attributes "city" (ciudad) and "accommodationId" (código), and the Integrated attributes correspond also to the table Accomodations (alojamientos_integrated) with the same names. In Fig. 16 (b) an example of the data registered in the "mapping_dw" table after executing the task "Mapping DW schema and Integrated Schema" is shown. It presents examples of values for the following columns: config, dw_schema, attribute_dw_schema, integrated_schema and attribute_integrated_schema, as defined in the configuration data model presented in Fig. 7.


Mapping DataWarehouse Schema and Integrated Schema

No hay fecha de vencimiento Mediana prioridad Hace momentos creadas

Esta consulta no dispone de descripción establecida.

Parte del proceso: 'Define DW Schema And Mapping'

Personas

Sin propietario Transferir

Kermit The Frog

Asignado a Reasignar

Subtareas

No hay subtareas definidas para esta tarea

Contenidos relacionados

No hay contenidos relacionados para esta tarea

Rellene el formulario que aparece a continuación y complete la tarea:

Name Schema	esquema_dw
Name Table	alojamientos
Atribute DataWarehouse	ciudad
Atribute Integrated	alojamientos_integrated-ciud
Atribute DataWarehouse	codigo
Atribute Integrated	alojamientos_integrated-codi

(a)

```
select * from mapping_dw
```

config	dw_esquema	atributo_dw_esquema	integrated_esquema	atributo_integrated_esquema
3	alojamientos	codigo	alojamientos_integrated	codigo
3	alojamientos	direccion	alojamientos_integrated	direccion
3	alojamientos	telefono	alojamientos_integrated	telefono
3	alojamientos	tipo	alojamientos_integrated	tipo
NULL	NULL	NULL	NULL	

(b)

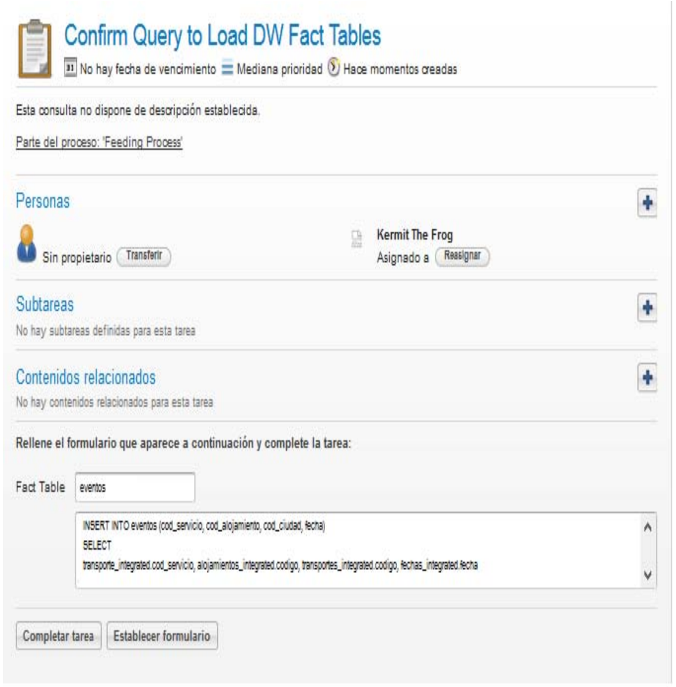
Fig. 16. (a) Activiti form to define mappings between the DW Schema and the Integrated Schema, (b) corresponding data in the "mapping_dw" table

Once the Configuration process has finished, the configuration data has been registered in the Configuration database filling the tables presented in Fig. 7, and the Feeding process to actually load the data into the WW can be executed.

B. *Execution of the WW Feeding process*

To validate the configuration BP and data we have defined, the proof of concept developed for the Feeding process was executed in order to analyze the WW generated. The first form asks the user to select a defined Configuration, in this case the one we have defined, and based on that the process is executed automatically. For each Feeding process a new data base is created with the name "base_id" being "id" the number of the selected configuration. Then the Expected Schemas, the Integrated and the DW schemas are created as defined in the configuration data, with corresponding tables and attributes.

Then the data is extracted from the Web Sources via invoking the Data Sources defined, and loaded into the Expected Schemas as defined. Continuing with the automated process, the data is loaded in the Integrated Schema following the definitions of mappings and joins, and then into the DW Schema as defined. In this case we do not have any integration issues since we have simplified the data to execute the basic scenario of DW loading. The only form that will be presented to the user during the process correspond to the confirmation of the query generated to load the DW fact tables, giving the user the option to change it. In Fig. 17 the Activiti form to confirm the query to load the DW fact tables is shown.


Confirm Query to Load DW Fact Tables

No hay fecha de vencimiento · Mediana prioridad · Hace momentos creadas

Esta consulta no dispone de descripción establecida.

Parte del proceso: 'Feeding Process'

Personas

Sin propietario (Transferir) — Kermit The Frog, Asignado a (Reasignar)

Subtareas

No hay subtareas definidas para esta tarea

Contenidos relacionados

No hay contenidos relacionados para esta tarea

Rellene el formulario que aparece a continuación y complete la tarea:

Fact Table: eventos

```
INSERT INTO eventos (cod_servicio, cod_alojamiento, cod_ciudad, fecha)
SELECT
transporte_integrated.cod_servicio, alojamientos_integrated.codigo, transportes_integrated.codigo, fechas_integrated.fecha
```

Fig. 17. Activiti form to confirm the query to load the DW

As it can be seen in Fig. 17, the fact table as defined in Fig. 14 named FACT_EVENTS (eventos) will be loaded with data corresponding to the dimension tables: DM-TIME (fecha), DM-ACCOMM (cod_alojamiento), DM-TRANSPORT (cod_servicio), DM-LOCATION (cod_ciudad), as defined by the mappings with the Integrated Schema. Finally the data for the DW will be loaded into the defined tables and will be ready to be used for exploitation.

V. RELATED WORK

Most related work we have found only focuses in the ETL phase of the construction of a DW and it mainly uses BPMN for design purposes, not to automating the process. For example, in [17][18][19] BPMN is used for conceptual modeling of ETL Processes. Also, in [20] a model-driven framework for ETL process development is presented. The framework allows modeling ETL processes in a platform independent way, using a meta-model named BPMN4ETL, and generating vendor specific code from these models.

Compared to those, our proposal also supports the ETL phase via a feeding process, including a previous configuration phase in which a configuration process is defined for the user to be able to input data to be used for the actual load of the DW. What is more, the BPMN 2.0 notation is not only used for modeling the configuration and feeding BPs, but also as basis for executing them in the open source BPMS Activiti. The provision of automated support to guide the building process of a DW in the way we do, is to the best of our knowledge, not proposed in the literature.

VI. CONCLUSIONS AND FUTURE WORK

We have presented the modeling, definitions and data needed to support the process of building flexible WW. Our complete approach is based on the definition of two phases with corresponding BPs: a Configuration phase with a configuration BP which aims to provide support for the user to define and input the data needed to build the WW, and a Feeding BP that in a second phase, using the configuration data, actually generates the WW. We have focused on the configuration BP presenting the model specified in BPMN 2.0, and the executing model also in BPMN 2.0 implemented in the open source BPMS Activiti. We have validated our approach with a case study using open government data from Uruguay. Our proof of concept implementation of the configuration BP although conceptually completed, presents some limitations. For example, we had not yet dealt with modifying already registered data, you could say we are assuming an ideal user who defines everything correctly the first time. We plan to add more loops allowing to modify the definitions regarding the schemas and mappings, so to be able to correct or delete things until the user is ok with the definitions. In a similar way, the Feeding process we have implemented was only to validate the configuration data we are registering, and needs more work.

The automated support for building flexible WW we propose will provide organizations with many benefits, such as bridging the gap between the involved roles (Domain Experts and IT professionals) as they can talk about the BP model and understand the different tasks that has to be carried out, provide easier to use definitions and guides as in the execution the process engine presents the tasks to be performed accordingly to the control flow defined in the process, provide automated execution as much as possible, whenever something can be performed automatically based on data retrieve from the user, it will be automated, provide data about the definitions taken by the user i.e. who defined what in what task, along with many other valuable information regarding the execution of the BPs which is automatically registered in the BPMS process engine as BPs are executed.

We are currently working on completely defining the Feeding process to be able to resolved and automate as much as possible some of the issues regarding data integration, and adding the quality view to both the Configuration and the Feeding processes, to complete the automated support for building Quality-Aware flexible WW with BPMS.

ACKNOWLEDGMENT

We would like to thank the students who worked in the development of the Configuration BP and its validation, F. Alonso and M. Mendiola.

REFERENCES

[1] Open Government Data http://opengovernmentdata.org/

[2] Marotta A., González L., Ruggia R., "A Quality Aware Service-oriented Web Warehouse Platform". In Proceedings of Business intelligencE and the WEB (BEWEB), EDBT, Berlín, Germany, 2012.

[3] Weske M., Business Process Management: Concepts, Languages, Architectures, Springer, 2007.

[4] van der Aalst W.M.P.,ter Hofstede A., Weske M., "Business Process Management: A Survey", In Proceedings Int. Conf. on Business Process Management (BPM), The Netherlands, 2003.

[5] Smith H., Fingar P., Business Process Management:The third wave, Meghan-Kieffer, 2003.

[6] BPMN 2.0, OMG, http://www.omg.org/spec/BPMN/2.0/

[7] Delgado, A., Ruiz, F., García-Rodríguez de Guzmán, I., Piattini, M. "Application of SOC and MDD paradigms to Business Processes: a systematic review". In Proceedings 5th Int. Conf. on Software and Data Technologies (ICSOFT'10), Athens, Greece, 2010

[8] Activiti BPMS http://activiti.org/

[9] Delgado A., Marotta A., González L., "Towards the construction of quality-aware Web Warehouses with BPMN 2.0 Business Processes". In procs. IEEE 8th International Conference on Research Challenges in Information Science, RCIS 2014, Marrakech, Morocco, 2014.

[10] Dumas M., van der Aalst W.M.P., ter Hofstede A., Process-Aware Information Systems: Bridging People and Software Through Process Technology, ISBN: 978-0-471-66306-5, Wiley&Sons Inc., 2005

[11] XPDL, WfMC, http://www.wfmc.org/xpdl.html

[12] WS-BPEL, OASIS, http://docs.oasis-open.org/wsbpel/2.0/

[13] Malinowski E., Zimányi E., Advanced data warehouse design: from conventional to spatial and temporal applications. Springer Science & Business Media, 2008.

[14] Ministerio de Turismo y Deporte. http://apps.mintur.gub.uy/datos Abiertos/ Last access: 30/5/2015

[15] Dirección Nacional de Transporte. http://catalogodatos.gub.uy/dataset/ aa2405be-6dde-4482-8f5f-563817154282/resource/e62a3ef8-4e00-458b-bcbe-4b293ce0ebba/download/HorariosOmnibusVerano2015.zip Last access: 30/5/2015

[16] Intendencia de Durazno. http://www.durazno.gub.uy/portal/datos-abiertos/3418-datos-abiertos-de-la-intendencia-de-durazno397 Last access: 30/5/2015

[17] Akkaoui Z. E., Mazón J.-N., Vaisman A., Zimányi E., "BPMN-Based Conceptual Modeling of ETL Processes," in Data Warehousing and Knowledge Discovery, A. Cuzzocrea and U. Dayal, Eds. Springer Berlin Heidelberg, 2012, pp. 1–14.

[18] Oliveira B., Belo O., "BPMN Patterns for ETL Conceptual Modelling and Validation," in Foundations of Intelligent Systems, L. Chen, A. Felfernig, J. Liu, and Z. W. Raś, Eds. Springer Berlin Heidelberg, 2012, pp. 445–454.

[19] Berkani N., Bellatreche L., Khouri S., "Towards a conceptualization of ETL and physical storage of semantic data warehouses as a service," Cluster Comput, vol. 16, no. 4, pp. 915–931, Dec. 2013.

[20] Akkaoui Z. El, Zimànyi E., Mazón J.-N., Trujillo J., "A Model-driven Framework for ETL Process Development," in Proceedings of the ACM 14th International Workshop on Data Warehousing and OLAP, New York, NY, USA, 2011, pp. 45–52.