From ETL Conceptual Design to ETL Physical Sketching using Patterns

Bruno Oliveira[1] and Orlando Belo[2]
[1]*CIICESI, School of Management and Technology, Porto Polytechnic, Felgueiras, Portugal*
[2]*ALGORITMI R&D Centre, University of Minho, Campus de Gualtar, Braga, Portugal*

Keywords: ETL Sketching, Patterns, Pattern-oriented Approach, Conceptual Design, BPMN, Logical Design, ETL Physical Implementation.

Abstract: The ETL systems development has been the focus of many research works, addressing the complexity and effort required for their implementation and maintenance, and proposing several techniques that represent valuable contributions to improve the ETL final quality. In the last few years, we presented a pattern-oriented approach for developing these systems based on patterns that encapsulate well-known design techniques. Basically, patterns embed common practices using abstract components that can be configured for enabling its instantiation according to each pattern rule. However, each ETL system is unique, dealing with very specific data structures and decision-making requirements. Thus, several operational requirements need to be considered and system correctness is hard to validate, which can result in several implementation problems. In this paper, we present a conceptual approach based on patterns covering the main ETL phases, ranging from the conceptual design to its enrichment at logical phases that can be used for the generation of executable programs.

1 INTRODUCTION

Design methodologies are used across software development areas for improving software quality, covering specific domain requirements and different design steps. Complex software pieces are usually supported by structured development processes approaching all development stages, from requirements identification to the implementation and maintenance phases. Data Warehouse Systems (DWS) projects represent a huge job involving personnel with different knowledge domains. The business users are the target customer since they will use DWS for exploring data and planing next tactical and strategic decisions. However, they have an important role in the development of a data warehouse since they know how the business works and together with IT/DWS professionals and consultors, represent an important entity to share and communicate in order to best assure the system quality and adequacy. The evaluation of a DWS highly depends on the data stored in a Data Warehouse (DW) repository. The importance of this aspect cannot be ignored by the development team. A DW is a centralized repository that provides the unified vision over company data, and for that reason data should be carefully managed in order to avoid mistakes that can compromise system usability. Additionally, the data sources involved differ in several aspects that contribute to disparities not only at structural level but also at the instance level. Legacy systems, specific business processes variants in company's department or even bad practices when dealing with operation systems (sometimes even the business managers do not know these problems), represent huge challenges to an ETL (Extract, Transform and Load) team, assuring that all the data involved in re represented in the same structure, but also for guaranteeing that the data involved is consistent and valid according to business rules.

In this paper, we present a methodology for ETL conceptual modelling, the necessary means for mapping produced conceptual models to logical models, and consequently to correspondent physical primitives, with the possibility to be executed directly in a commercial tool. The paper is organized as follows: section 2 presents a brief related work for providing some important findings that contributed to the development of the presented approach; section 3 presents an introduction to the pattern-oriented

Oliveira, B. and Belo, O.
From ETL Conceptual Design to ETL Physical Sketching using Patterns.
DOI: 10.5220/0006807702620269
In *Proceedings of the 20th International Conference on Enterprise Information Systems (ICEIS 2018)*, pages 262-269
ISBN: 978-989-758-298-1

approach we developed and the established conceptual models based on ETL patterns following a simple case study; section 4 highlights the approach idealized in order to allow the construction of logical and physical models based on the developed conceptual representation. Finally, section 5 presents some conclusions and findings about the work done.

2 RELATED WORK

The general idea of using software patterns to build ETL processes was first explored by Köppen (Köppen et al., 2011) that proposed a pattern-oriented approach for supporting ETL development. A general description for a set of patterns such as aggregator, history and duplicate elimination was provided then along with important aspects related to the composition and relationship between properties, describing constraints applied to the use of each pattern. The patterns were presented only at the conceptual level, lacking to support patterns instantiation for execution primitives. However, since ETL processes support very specific needs related to DWS requirements, they can be handled in several ways. The pattern-oriented approach presented here differs from the work presented by Köppen since a set of configurable components that can be configured and used in different ETL development phases are provided, covering initial design phases for producing skeletons that can be enriched to allow its mapping to execution primitives, which we may refer as ETL physical sketching. Fine-grained tasks are encapsulated and generated inside these components, resulting in a new ETL development level, defined using an abstraction layer that simplifies and carries the acquired knowledge between projects.

We idealized the development process using three layers: the conceptual phase describing which patterns are being used and the workflow constraints applied to their usage, the logical phase describing additional properties to enrich the previous conceptual models with detail that can be used at the physical model for generating an initial skeleton of the ETL implementation, at least.

For conceptual representation we choosed the BPMN (Business Process Model and Notation), a notation that was first explored by Akkaoui and Zimanyi (2009) for ETL conceptual modelling, covering also its mapping and execution through the use of BPEL language. Later, they reinforced their original ideas proposing to model ETL generic processes, independently from their potential support platforms (Akkaoui et al., 2011). Their approach revealed a considerable potential, particularly in what was concerned with the mapping of conceptual models in a set of execution primitives.

The findings presented by Akkaoui et al. (2011) revealed that conceptual model specification is quite appropriate since BPMN can represent with accuracy the most common ETL tasks. We explored BPMN to create several ETL layers using the “sub process” activities, coordinated with particular BPMN elements that are very powerful and well known by many companies and professionals of today. BPMN was developed for supporting process execution with constructs (essentially in its 2.0 version), allowing for the representation of workflows at the physical level with the ability to support its execution. Using a very practical way, these authors show how the BPMN notation, originally designed for modelling business processes, could be successfully adopted for ETL systems modelling and interpreted by computers using the interchangeable formats available to support BPMN representation. However, one of their main flaws is related, precisely, with their expressiveness that results in several ways for representing the same scenario, resulting in ambiguous processes that are difficult to map to a set of execution primitives. In fact, this was analysed and discussed by several authors (Dijkman et al., 2008; Ou-Yang and Lin, 2008).
The construction freedom that BPMN provides generates also some problems related to business processes reliability, which can result from an inconsistent identification of business requirements, mainly due to the methods or model validations that are typically manual and error-prone. We believe we can minimize these problems using ETL patterns not only for conceptual representation but also for logical representations using configurable properties expressed with a specific language that can be embedded in the BPMN modelling.

3 REQUIREMENTS ANALYSIS AND CONCEPTUAL DESIGN

The use of software patterns is a reuse-based technique often applied in software developing on a lot of different domains (Gamma et al., 1994), supporting component reuse techniques and sharing the acquired knowledge across applications. Thus, not only the time and costs needed for software development can be reduced, but also the use of well-proven techniques is guaranteed, contributing to final system quality. Creating these reconfigurable

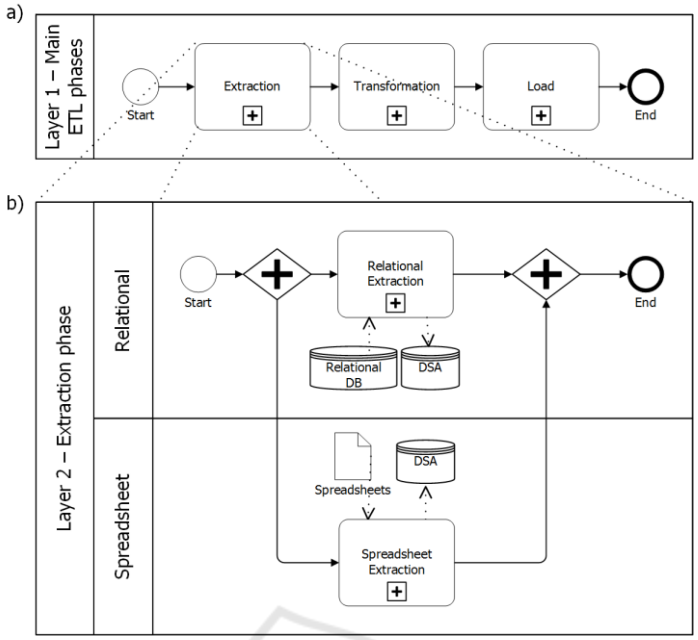

Figure 1: ETL elementary tasks (a) and extraction phase representation (b).

components, avoid the need to rewrite some of the most repetitive tasks that are used regularly. Several tasks, such as surrogate key process generation, data quality enhancement procedures or slowly changing dimensions are just some few examples of usual procedures that are considered standard to deal with specific ETL tasks. This way, users can focus on more general requirements, leaving the complexity of its implementation to others development steps. Consequently, ETL designers only need to provide configuration metadata to enable pattern instantiation for its physical implementation.

The concept of a pattern was idealized as "core" that encapsulates a set of pattern rules for supporting, namely, specific operational requirements and the logic behind it, the input and output interfaces to communicate both with ETL workflow and data layer to produce specific instances, and the communication layer with other patterns. Each pattern provides the construction rules to sustain well-formed ETL workflows based on a specific configuration. The "throwable" and "log" pattern components represent additional ETL metadata that is used to support the error and log strategies for handling errors and pattern unexpected events. Using the communication layer, the "throwable" pattern component uses the input configuration to handle error or exception scenarios through the application of specific pre-configured strategies for each pattern. The unexpected scenarios that cause critical failure scenarios can be configured to use specific procedures to maintain data consistency. For example, the process can be aborted and use a rollback strategy, the performed tasks are reverted to maintain data in a consistent state. The "log" pattern component can be used together with "throwable" component to store the events related to unexpected scenarios, or used for specific support tasks related to the identification of data lineage, bottlenecks or errors. Thus, the ETL process can be analyzed and, for example, specific error trends can be found and specific strategies to minimize them can be followed for reducing the ETL resources needed for subsequent loads, eventually. Log structures can vary in granularity and scope and its entries can be triggered by conditions associated with pattern internal behavior or by more general conditions (such as achieved milestones).

For an initial ETL planning, we idealize the same constructs that can be used for ETL pattern-oriented development, using patterns such as:

- Changing Data Capture, representing the extraction procedures to carry source data to Data Staging Area.
- Data Quality Enhancement, representing a set of data transformations required for cleaning, conforming and standardizing data, considering the target repository constraints.
- Data Conciliation and Integration, representing the conciliation and integration logic needed to obtain an integrate view of the data gathered from heterogenic sources.

- Intensive Data Loading, representing efficient strategies to load data from data staging area to the target data warehouse repository

The use of conceptual models represents a crucial step for software lifecycle development, providing an abstract view that simplifies the problem representation, helping designers to better understand processes and at the same time to provide a framework to discuss the system to develop. As already discussed, several contributions concerning the ETL development simplification were presented so far, with some of them proposing the use of well-known modelling languages such as the use of BPMN (El Akkaoui and Zimanyi, 2009) for ETL conceptual modelling.

To better understand how these aspects can be linked together, let us consider a solid urban waste collection process realized in a city (Belo et al., 2015b). This process represents a significant impact in any city daily life and entities responsible for urban zones are more worry about the effectiveness of their waste collection processes implementation, to maximize the adopted solutions they adopted. To cover some of the most relevant information needs that managers usually have on waste collection management, a business intelligence system especially oriented to monitor, control and predict waste management services is planned.

The company we simulate uses a set of spreadsheet files to store all the data related to the collection activities. The "Picking" spreadsheet contains the year and the reference of the containers that were collected in a specific month/day, the "RecycleCentreResume" spreadsheet identifies the recycle centres data and the related containers associated. Both spreadsheets are related based on recycle centre address.The Picking spreadsheet structure is composed of a set of garbage picking activities performed every day for a specific city in each month. After the month and day identification (first two rows), for each type of garbage collected: 'P' - Plastic, 'C'- Cardboard, and 'G' – Glass (third row), the container reference placed in the remaining rows identifies the container that was collected. The "RecycleCentreResume" spreadsheet stores the relationship between containers and recycling centres for each city. Basically, each recycle centre contains at least three containers for the garbage types described before (Plastic, Cardboard, and Glass). The company uses several recycling centres composed of the three types of containers and spread across several locations. According to a set of routes, the company picks the containers of each recycling centre, storing each garbage pickup activity using the described spreadsheets.

To simplify ETL development, a task clustering technique to group the set of finer grain tasks into a collection of tasks, flows and control primitives is presented, providing a method to organize ETL tasks using abstraction layers to serve specific stakeholders in different project phases. The conceptual representation can be organized using several layers with different detail levels based on the project needs. To support different abstraction levels, the BPMN "collapsed sub processes" can be used to provide process conceptualization to describe complex constructs using distinct levels. This is very helpful for high-level users when presenting, discussing and understanding process concepts.

For showing how BPMN can be customized for designing ETL processes, following the pattern-oriented approach presented, a subset of BPMN constructs were selected to compose the conceptual palette of constructs considered for the development of ETL conceptual models. To describe ETL activities, atomic and composite tasks are represented using "Task" and "Sub process" elements, respectively. Both can be enriched using specific markers (following the same combination restrictions imposed by BPMN specification) to describe specific operational requirements. For example, the "Standard Loop" can be used to describe a row-by-row data processing, the "multi-instance" marker to describe multiple activity instances that handle specific sets of data, and "Compensation" to represent compensation processes responsible to undone/compensate the actions performed by some activity. Additionally, sub processes can be used to express a hierarchical workflow structure, encapsulating several ETL logical parts inside specific containers, helping to represent processes in a more readable way. The ETL patterns are represented using sub processes, more specifically using the "Transaction" sub process type, grouping a set of activities that constitute a logical unit of work that must be executed atomically. To distinguish between ordinary BPMN sub processes from ETL patterns, two approaches can be followed to identify each pattern: "Text annotations" can be used not only to identify patterns but also to provide a pattern general description. Influenced by the work presented by (El Akkaoui and Zimanyi, 2009), we use the BPMN "Text Annotation" artefacts to enrich conceptual model, exposing specific semantic concepts to describe pattern details following a specific structure. The BPMN "Events", "Gateways", "Flow Objects", "Message Flow", "Data" and "Artefacts" have the same semantic when are used in

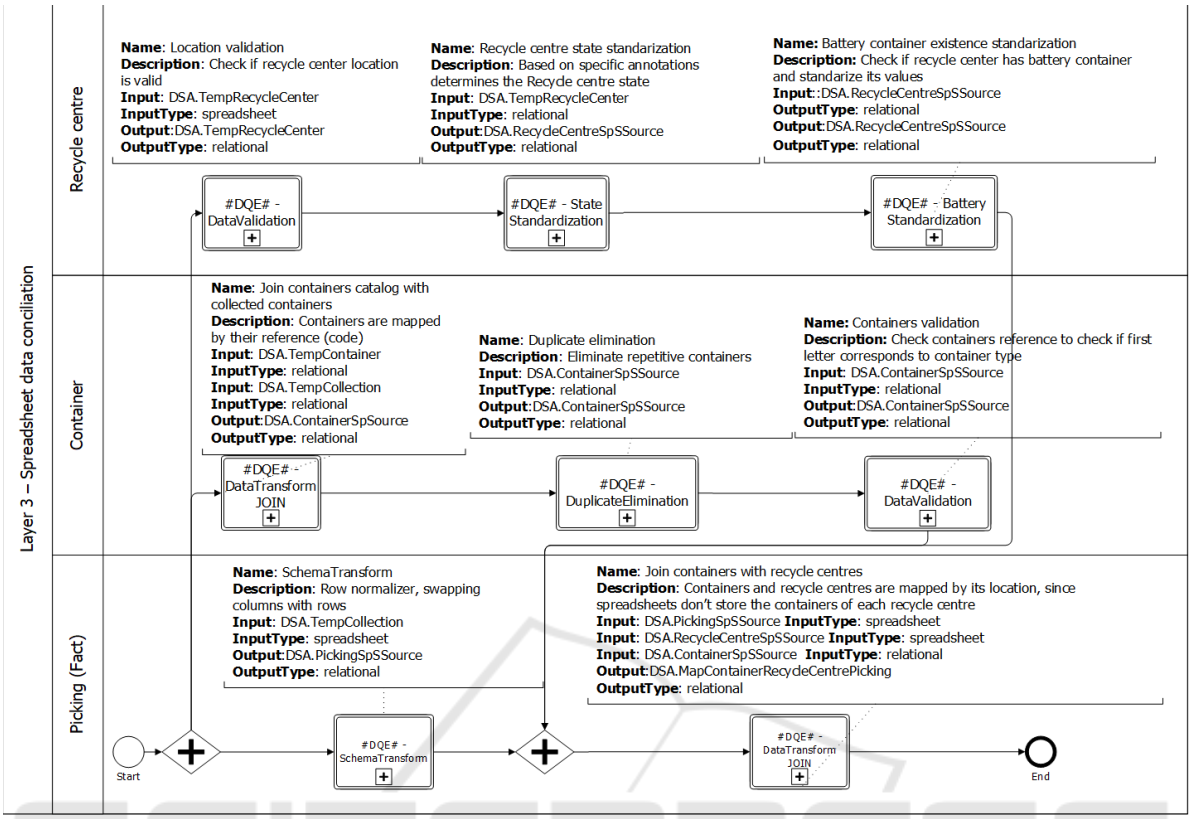

Figure 2: ETL Conceptual model representation using BPMN.

traditional BPMN processes: The BPMN "Pools" and "Lanes" are used to represent process levels that describe different views and roles involved in the modelled process. These perspectives allow for the representation of several ETL layers useful to support different development stages. While BPMN pools are used to represent different process views, the use of "Lanes" allows for the representation of different entities that can be used for the representation of specific data sources and process variants. The "Collapsed pools" are also used to hide process logic and for the representation of the exchanged messages between pools.

To show how these BPMN artefacts can be used for ETL conceptual modelling - Figure 1 presents a simple example. The most generic layer - "Layer 1" (Figure 1a) represents the most abstract level that can be derived from the three main ETL phases (Extract, Transform and Load). The ETL extraction and transformation processes are presented at Layer 2 (Figure 1b), representing the extraction procedures applied to the source data. Two BPMN "parallel gateways" are used to indicate that tasks belonging to the flow comprised by the parallel gateways don't have any dependency. The ETL extraction processes – "Layer 2" (Figure 1b) – begins with data extraction tasks applied over the spreadsheet files ("Spreadsheet Extraction" sub process) and a relational database ("Relational Extraction" sub process). These are the two types of data storage approaches used by company branches. To described this, we used a BPMN pool with two internal lanes, representing each one a different role related to the data source nature (relational or spreadsheet). Each BPMN sub process is composed of several finer grain tasks grouped together that can be described in more detailed layers. Additionally, BPMN artefacts ("Data Object" and "Data Store") were used to describe the nature of each data repository (structured data for data stores artefacts and semi-structured data for data objects artefacts) for each sub process with BPMN "Data association" artefacts describing the relationship directionality: from sub processes to Data Artefacts indicating the output data flow and from data artefacts to sub processes, indicating input data flows.

After data extraction to the DSA, the transformation/normalization processes take place. The Figure 2, presents the "Layer 3" (with patterns instantiation) for the "Transformation" sub process

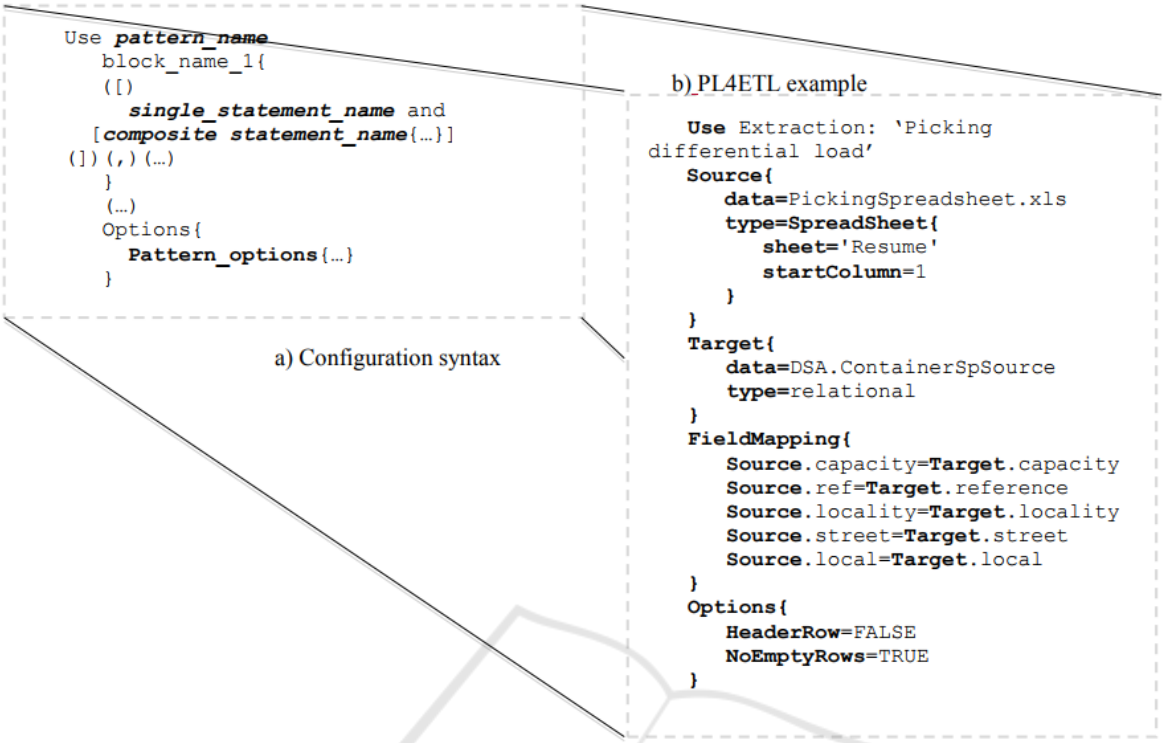

Figure 3: DSL excerpt for a Differential extraction pattern configuration.

from Figure 1a. Several pattern instances are used for aligning source data structures with the target DW schema requirements with each pool representing data transformation procedures applied to the "Recycling centre", "Container" and "Picking "entities for data previously extracted. To deal with the "Recycle centre" data, we used three DQE patterns instances for transforming data from each Recycle centre: "#DQE# - DataValidation", "#DQE# - StateStandardization" and "#DQE# - BatteryStandardization", which are used for data instances validation. For each container, the several DQE patterns instances are applied. The "#DQE# - Data Transform JOIN" used to map the containers data catalogue with the containers used for the picking activities processed, the "#DQE# - Duplicate Elimination" used to remove duplicate containers (because the same container can be collected several times), and the "#DQE# - DataValidation" used to check containers reference values, ensuring the consistency of their code with their type.

For the data related to picking activities, we need to transform the original schema to get the data according to the structure of a typical relational table. Data is decomposed using the "#DQE# - SchemaTransform" pattern instance from the "Picking Fact" pool, generating distinct rows based on each garbage picking activity. Next, when the pattern instances from the other pools are executed (the convergent BPMN gateway ensures that) and having the data about recycle centres and containers normalized, we can proceed to the mapping between containers, recycle centres and picking activities to identify the correct correspondences between them. For that reason, we should develop supplementary efforts for identifying the associated containers for each recycle centre ("#DQE# - DataTransform JOIN") Any record without correspondence or having conflicts can be managed through the definition of a specific exception handler policy for managing unexpected situations and ensuring the process consistency (not included in the BPMN representations).

4 LOGICAL DESIGN AND PHYSICAL DETAILS

The ETL modelling activity should reflect different control flows and data between the represented tasks. Using BPMN for the ETL specification allows the use of expressive notation and orchestration mechanisms that can be used for subsequent development stages. The use of BPMN in this type of modelling allows the

control flow representation between the ETL process activities, as well as the characterization and description of the activities to implement. The real challenge relates to the combination of these two types of flows in one single representation model and, therefore, in a single tool (Wilkinson et al., 2010). While BPMN conceptual models can be used to produce (at least) a first version of the target system a domain-specific language (DSL) can be used to express each pattern behaviour to reduce ambiguities associated to BPMN conceptual models. The DSL is used for pattern configuration to produce logical models with the ability to describe in more detail the logic behind each pattern. The developed DSL covers the general requirements for each pattern category, providing a powerful way to configure each pattern behaviour.

The Figure 3 shows an example of the syntax rules applied to the language we are being developed (Belo et al., 2015a; Oliveira and Belo, 2015), with a correspondent example of its instantiation to support container's data load. The "Source" and "Target" blocks describe input and output metadata, while the data and type properties describe the name and type of each data object used. These blocks can be derived from the annotations used in the conceptual model specification. The "FieldMapping" block describes the field association between the source and target data objects. For input block, a spreadsheet file is used for data extraction based on a specific sheet included in the Picking spreadsheet. The "Sheet" property is used to identify the specific sheet and the "StartColumn" property to indicate in which column the process uses for data extraction (since two properties are described, a composite statement is used). The "Output" block is composed of single statements describing the name ("data") and the type ("type") from target repository. Details such as the database name or server were omitted since they can be configured in further steps. The fields mapping is described using "Source" or "Target" prefixes to identify the source for each field (especially relevant when a field name is ambiguous). After fields identification, the keyword: "OPTIONS" is used to specify each pattern parameter. In this case, for avoiding the first file row and ignoring the rows without data.

Before proceeding to the physical model generation, some guarantees must be assured by earlier modelling process phases to enable physical generation models that can be executed properly and without flaws. The transformation process relies on a converter (a program written in Java programming language) to take the BPMN models, parse respective DSL and construct the final system. Later, the resulted piece of software will be imported to the ETL tool environment selected: the Pentaho Data Integration (Bouman and Dongen, 2009). The conceptual specification described using BPMN is used to identify the patterns to support data transformation processes, its sequence, and control flow constraints that allow for the identification of execution specificities applied to patterns execution. The ETL organization tree can be directly described in BPMN conceptual representation or be inferred based on the patterns nature through the identification and organization of each pattern hierarchy in layers.

5 CONCLUSIONS

The ETL conceptual model development is clearly a great advantage in a DWS project, however, not always the efforts spent on ETL system modelling are rewarded since they are frequently discarded in favour of a more detailed logical model. Based on BPMN and on a set of ETL patterns, a specific ETL development process that enhances the importance of building ETL conceptual models to establish a first executable version of the system – an ETL skeleton, was designed and implemented. Since the ETL processes execution parameters differ in some aspects from traditional business processes execution platform, the conceptual approach presented do not intend to describe how the patterns will be executed, but rather the patterns that are used and how they are coordinated by the coordination mechanisms provided by BPMN specification. With the domain-level view provided by BPMN, the use of patterns enhances the separation of workflow orchestration and data transformation patterns. A specific DSL especially oriented to describe the behaviour of an ETL pattern when integrated into an ETL conceptual model can be followed to mitigate the major drawback between conceptual and practical models: ambiguities that naturally exists due to each representation purposes. The physical model's generation processes mainly depend on the tool methodology and architecture used, which means that a tool expert should be engaged in this job. The parser is responsible for physical model's generation, using the conceptual and logical specifications to extract all operational requirements needed to meet the target tool architecture requirements.

REFERENCES

Belo, O., Oliveira, B., Lopes, C., Marques, R., Santos, V., 2015a. Using a Domain-Specific Language to Enrich ETL Schemas, in: Morzy, T., Valduriez, P., Bellatreche, L. (Eds.), New Trends in Databases and Information Systems. Springer International Publishing, Poitiers, France, pp. 28–35. https://doi.org/10.1007/978-3-319-23201-0_4.

Belo, O., Oliveira, B., Medeiros, M., Faria, P., Leite, M., 2015b. Improving Selective Collection of Urban Waste Using a Business Intelligence System, in: 3rd International Conference WASTES: Solutions, Treatments and Opportunities (WASTES'2015).

Bouman, R., Dongen, J. Van, 2009. Pentaho® Solutions: Business Intelligence and Data Warehousing with Pentaho and MySQL®, Solutions. John Wiley & Sons, Inc.

Dijkman, R.M., Dumas, M., Ouyang, C., 2008. Semantics and analysis of business process models in BPMN. Inf. Softw. Technol. 50, 1281–1294. https://doi.org/https://doi.org/10.1016/j.infsof.2008.02.006.

El Akkaoui, Z., Zimanyi, E., 2009. Defining ETL worfklows using BPMN and BPEL, in: Proceedings of the ACM Twelfth International Workshop on Data Warehousing and OLAP. ACM, Hong Kong, China, pp. 41–48. https://doi.org/10.1145/1651291.1651299.

El Akkaoui, Z., Zimànyi, E., Mazón, J.-N., Trujillo, J., 2011. A Model-driven Framework for ETL Process Development, in: Proceedings of the ACM 14th International Workshop on Data Warehousing and OLAP, DOLAP '11. ACM, New York, NY, USA, pp. 45–52. https://doi.org/10.1145/2064676.2064685.

Gamma, E., Helm, R., Johnson, R., Vlissides, J., 1994. Design Patterns: Elements of Reusable Object-oriented Software, Addison-We. ed. Pearson Education.

Köppen, V., Brüggemann, B., Berendt, B., 2011. Designing Data Integration: The ETL Pattern Approach. Eur. J. Informatics Prof. 13, 49–55.

Oliveira, B., Belo, O., 2015. A Domain-Specific Language for ETL Patterns Specification in Data Warehousing Systems, in: Pereira, F., and Machado, P., and Costa, E., and Cardoso, A. (Eds.), 17th Portuguese Conference on Artificial Intelligence (EPIA'2015). Springer International Publishing, Coimbra,Portugal, pp. 597–602. https://doi.org/10.1007/978-3-319-23485-4_60.

Oliveira, B., Belo, O., 2014a. Modelling ETL workflows using YAWL, in: Hammoudi, Slimane and Maciaszek, Leszek and Cordeiro, J. (Ed.), 16th International Conference on Enterprise Information Systems (ICEIS). SCITEPRESS - Science and Technology Publications, Lda, Lisbon, Portugal, pp. 299–307. https://doi.org/10.5220/0004947302990307.

Oliveira, B., Belo, O., 2014b. On the Conceptualization of ETL Patterns A Reo Approach, in: Almeida, A.M., Bernardino, J., Gomes, E.F. (Eds.), 18th International Database Engineering & Applications Symposium (IDEAS). ACM, Porto, Portugal, pp. 348--351. https://doi.org/10.1145/2628194.2628247.

Oliveira, B., Belo, O., 2013. Using Reo on ETL Conceptual Modelling - A First Approach, in: Cuzzocrea, I.S. and L.B. and A. (Ed.), Proceedings of the Sixteenth International Workshop on Data Warehousing and Olap, Dolap 2013. ACM, San Francisco, California, USA, pp. 55–60. https://doi.org/10.1145/2513190.2513202.

Oliveira, B., Belo, O., Cuzzocrea, A., 2014. A pattern-oriented approach for supporting ETL conceptual modelling and its YAWL-based implementation, in: And, M.H., And, A.H., And, O.B., Francalanci, C. (Eds.), Proceedings of 3rd International Conference on Data Management Technologies and Applications. SciTePress, Vienna, Austria, pp. 408–415.

Ou-Yang, C., Lin, Y.D., 2008. BPMN-based business process model feasibility analysis: a petri net approach. Int. J. Prod. Res. 46, 3763–3781. https://doi.org/10.1080/00207540701199677.

Vassiliadis, P., Simitsis, A., Terrovitis, M., Skiadopoulos, S., 2005. Blueprints and Measures for ETL Workflows, in: Delcambre, L., and Kop, C., and Mayr, H.C., and Mylopoulos, J., and Pastor, O. (Eds.), Proceedings of Conceptual Modeling -- ER 2005: 24th International Conference on Conceptual Modeling. Springer Berlin Heidelberg, Klagenfurt, Austria, pp. 385–400. https://doi.org/10.1007/11568322_25.

Vassiliadis, P., Vagena, Z., Skiadopoulos, S., Karayannidis, N., Sellis, T., 2000. ARKTOS: A tool for data cleaning and transformation in data warehouse environments. IEEE Data Eng. Bull. 23, 42–47.

Wilkinson, K., Simitsis, A., Castellanos, M., Dayal, U., 2010. Leveraging Business Process Models for ETL Design, in: Parsons, J., and Saeki, M., and Shoval, P., and Woo, C., and Wand, Y. (Eds.), Conceptual Modeling -- ER 2010: 29th International Conference on Conceptual Modeling. Springer Berlin Heidelberg, Vancouver, Canada, pp. 15–30. https://doi.org/10.1007/978-3-642-16373-9_2.