Override Traditional Decision Support Systems
How Trajectory ELT Processes Modeling Improves Decision Making?

Noura Azaiez[1] and Jalel Akaichi[2]
[1]*Department of Computer Science, Institut Supérieur de Gestion, University of Tunis, Tunis, Tunisia*
[2]*Department of Computer Science, College of Computer Science, King Khalid University, Abha, Saudi Arabia*

Keywords: Trajectory ELT Processes, Extraction, Loading, Transformation, Trajectory Construction, Trajectory Data Source Model, Trajectory Data Mart Model, Model Driven Architecture.

Abstract: Business Intelligence is often described as a set of techniques serving the transformation of raw data into meaningful information for business analysis purposes. Thanks to the technology development in the realm of Geographical Information Systems, the so-called trajectory data were appeared. Analysing these raw trajectory data coming from the movements of mobile objects requires their transformation into decisional data. Usually, the Extraction-Transformation-Loading (ETL) process ensures this task. However, it seems inadequate to support trajectory data. Integrating the trajectory aspects gives the birth of Trajectory ETL process (T-ETL). Unfortunately, this is not enough. In fact, the business analysis main purpose is to minimize costs and time consuming. Thus, we propose to swap the T-ETL tasks scheduling: instead of transforming the data before they are written, the Trajectory Extraction, Loading and Transformation (T-ELT) process leverages the target system to achieve the transformation task. In this paper, we rely on a set of powerful mechanisms to handle the complexity of each T-ELT task. Wherefore, an algorithm is dedicated to ensure the transformation of raw mobile object positions into trajectories and from there we highlight the power of the Model-driven Architecture approach to transform the resulting trajectories into analytical data in order to perform the Business Intelligence goal.

1 INTRODUCTION

Often Business Intelligence (BI) applications use data gathered from data warehouse (DW) to support a wide range of business decisions ranging from operational to strategic. Data warehousing concept evolves according to technological developments. In fact, the incredible progress related to Geographical Information Systems and positioning technologies imposes moving beyond the traditional to provide the real-time tracking of the movement of mobile objects and generate a new data kind called Trajectory Data (TD). TD collected from trajectory sources reflect the moves and stops of the mobile objects in the real world. However, the trajectory operational data sources are poorly suited to long-term vision and therefore seem inadequate for decision making. Towards this inadequacy, the notion of Trajectory Data Warehouses (TDWs) was appeared (Azaiez and Akaichi, 2016) to support TD. Before their availability for decisional purposes, raw TD have to flow through a set of tasks under the general title Extract-Transform-Load trajectory ETL process. This latter is responsible for extracting data from heterogeneous sources, cleaning them according to business rules, and loading the standardized data into a TDW. However, it is not enough. In fact, the business analysis main purpose is to minimize costs and time consuming. For that, we propose to swap the ETL tasks scheduling by isolating the extract and load processes from the transformation process. Instead of transforming the data before they are written, Trajectory Extract, Load and Transform (T-ELT) processes leverages the target system to perform the transformation task.

In this paper, we rely on a set of powerful mechanisms that ensure the better data transformation, starting from extracting TD from sources, loading them to reach the transformation task that has to occur in the target area. Since the transformation task is the core of the T-ELT processes, we divide the running of this task into two stages. The first sub-task offers an algorithm that focuses on transforming trajectory raw positions, describing mobile objects movements, into full trajectories. The generated trajectories are stored in a Trajectory Database (TDB) as a Trajectory Data Source model (TDSrc). For decisional purposes, we emphasize on the power of the Model Driven Architecture (MDA) (OMG, 2004) to transform TDSrc into Trajectory Data Mart (TDM) models. This presents the second transformation sub-task goal.

To reach goals scheduled above, we propose to organize this paper as follow. In section 2, we discuss some research works that treat the ETL and T-ETL process modeling. Section 3 presents our T-ELT proposed framework. Section 4 is dedicated to illustrate our approach by an Epilepsy patient states case study. We conclude the paper in section 5.

2 STATE OF THE ART

ETL processes emerged to provide a development environment for maintenance, analyzing and discovering. In this section, we present some practical solutions proposed by the research community that ETL design presents its area of interest.

2.1 ETL Processes Modeling

Data Warehousing emerged to collect and structure data at the aim to be the process of a good decision making. ETL processes present a key component of Data Warehousing since misleading data may produce wrong business decisions. Despite its importance, few are the research works that dealt with the modeling of ETL processes.

The first attempt towards a conceptual model dedicated to the design of the data warehouse refreshment was presented in (Vassiliadis *et al.*, 2003). To deal with the ETL modeling problem, authors proposed a generic meta-model for the definition of the data centric part of ETL activities. The customization and the extensibility of the meta-model building allows designer to enrich it with his own re-occurring patterns. Basing on their proposed meta-model, a reusable framework named *ARKTOS II* is introduced. *ARKTOS II* is complemented in (Simitsis, 2003) wherein the author investigated the ETL modeling problem relying on a method that treats the collection of requirements and analyzes the structure and the content of the existing data sources and, their intentional mapping to the common DW.

Another manner to deal with the ETL modeling is offered in (Boussaid et al., 2003) wherein authors proposed a new approach for complex data integration, based on a Multi-Agent System (*MAS*). *MAS* is composed of a set of intelligent agents which are responsible to achieve the major tasks of ETL modeling.

In order to create an active ETL tool, Rifaieh and Benharkat (2002) specified a model based on queries to represent mapping guideline and expressions between the source and the target data. They proved this model by creating a warehousing tool (*QELT*).

2.2 Trajectory ETL Processes Modeling

As it is presented above, several approaches were proposed to solve the problem of the ETL processes modeling. Those solutions seem unable to support Trajectory Data. The research work (Zekri and Akaichi, 2014) dealt with this problem and presented a Trajectory ETL model that is able to clean and manage Trajectory Data. Authors relied on the UML language in order to describe the ETL processes as a flow of activities. Authors focused on the TDW conceptual modeling in order to facilitate the analysis of Trajectory Data in the multidimensional context. In their case, they need to analyze the activities of a mobile medical delegate. Hence, they proposed two algorithms to implement trajectory ETL tasks and construct trajectories.

Marketos et al. (2008) investigated the problem of ETL modeling for TDW design. Their work supports a set of steps for building a real world TDW, from trajectory reconstruction to data cube feeding and aggregating over summary information. They relied on a set of algorithms and equations in order to extract measures that are able to handle the ETL processes modelling.

2.3 Discussion

The literature meets several research works that investigated the ETL modeling in different manners. Following the study of the sample described above, it becomes possible to discuss the strengths and weaknesses of each one. According to a depth study in ETL design field, three major approaches are the basis of authors works: ETL modeling based on a) conceptual constructs (Vassiliadis et al., 2003; Simitsis, 2003), b) UML environment (Boussaid et al., 2003; Zekri and Akaichi, 2014) and, c) mapping expressions and guidelines (Rifaieh and Benharkat, 2002). The ETL processes scheduling is the common point that the majority of works relied on. In fact, authors investigated the problem of the ETL modeling basing on the same strategy; Extraction, Transformation and then Loading. Relying on the ETL processes is a gainful strategy at the time consuming level. In fact, only data relevant to the solution are extracted and processed. Hence, the target area contains only data relevant to the presentation. This potentially reduce development and therefore time consuming; reduced warehouse content simplifies the security regime implemented and thus the administration overhead. However, the ETL strategy complains from several weaknesses especially the lack of flexibility which is explained by the fact of targeting only relevant data for output. That means that any future requirements, which may need data that weren't included in the original design, must be added to the ETL routines. As a result, this increases time and costs involved.

Integrating the concept of mobility into the professional realm offers the possibility of reducing geographical disparities related to organization services. However, few are the works that dealt with the Trajectory ETL problem as (Zekri and Akaichi, 2014) wherein authors relied on the UML language in order to describe the ETL scenario, and (Marketos et al., 2008) in which authors relied on a measures to generate algorithms that serve the trajectory ETL handling. The proposed frameworks serving the Trajectory ETL process modeling are complex and need high level experts to handle all tasks. Thus, we propose a new approach that aims at moving beyond Traditional DSS in order to enhance extracted knowledge quality. After taking a look on the works relying on the MDA approach, we note that it expresses its powerful features in some areas such as the DW evolution (Taktak et al., 2015) wherein authors relied on the MDA approach to automate the propagation of the evolutions occurred in the source database towards the multidimensional DW.

3 OUR APPROACH OVERVIEW

Thanks to the Trajectory ELT process strategy, all data, reflecting the moves and stops of mobile objects in the real world, are extracted and loaded into the warehouse. This, combined with the isolation of the transformation process, offers an easy feasibility of incorporating future requirements into the DW structure; what proves the future efficiency of the ELT process. Minimizing risks is another issue that can be solved relying on ELT process. In fact, isolating the load process from the transformation process removes an inherent dependency between each stage of the DW build processes. This provides an excellent platform for maintenance. As a result, basing on ELT process presents a tremendous payoff strategy for decision making systems. Our proposed framework relies on the Trajectory ELT (T-ELT) process (Figure 1) to emphasize a new gainful strategy that aims at transforming trajectory raw data into trajectory decisional data.

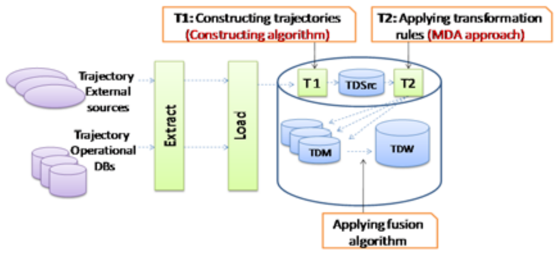
T1: Constructing trajectories (Constructing algorithm)
T2: Applying transformation rules (MDA approach)
Trajectory External sources
Trajectory Operational DBs
Extract
Load
T1
TDSrc
T2
TDM
TDW
Applying fusion algorithm

Figure 1: Our approach framework (T-ELT).

As the T-ELT is considered as a flow of activities, we choose to begin this flow with the identification of geographic points. To do so, we focus on extracting (E) data from external sources. The extraction method chosen by decision makers is highly dependent on the source system. While speaking about trajectory data, we have to extract trajectory data that describe mobile object activities from external trajectory sources. A generator tool is responsible to perform this task. Note that collecting these data requires its equipping with a positioning system as GPS… We call trajectory raw data the data as captured from the positioning device. However, these data are not available for analysis. Here, the main role of the Loading task (L) emerges. In effect, the loading task is responsible to load all extracted TD in the target server to be ready for the required transformations. Achieving this task cannot be ensured by traditional loading tools. Thus, ELT works with high-end data engines such as Hadoop clusters. In fact, Hadoop Distributed File System (HDFS) is able to store everything from structured and unstructured data. Relying on Hadoop technology was not randomly chosen. It presents a profitable strategy from all sides. Starting from rapidity that is required in our work. Hadoop clusters are known for boosting the speed of data analysis applications. If a cluster's processing power is overwhelmed by growing volumes of data, additional cluster nodes can be added to increase throughput. Besides, Hadoop clusters also are highly resistant to failure because each piece of data is copied onto other cluster nodes; which ensures that data are not lost if one node fails. As Hadoop features are conform to the loading goal, espacially for ELT process, we rely on this technology to load all gathered raw TD in the warehouse area.

For decisional purposes, TD recently loaded have to be transformed. Due to its complexity, we propose to divide the transformation process into two sub tasks: T1 and T2. T1 purpose is to convert the unstructured positions received by positioning devices into structured TD in order to describe the continuous objects movements. So, we propose a ***trajectory construction algorithm*** that transforms sequences of raw time-stamped positions location data into meaningful trajectories that are then stored into a Trajectory Database (TDB) as a Trajectory Data Source model (TDSrc).

– *Trajectory construction algorithm description*

Raw points arrive in bulk sets. Each point P (x, y, t) has to be affected to its appropriate Mobile Object (MO). Hence, each time the algorithm meets a new position (P_{new}), it checks whether the MO has been processed so far. If so:

- We have to check if this new point (P_{new}) satisfy a set of conditions to be added to the existing partial trajectory (T) of MO or to create a new trajectory (T_{new}) starting point. For that, we inspire from the work of (Marketos et al., 2008) and (Zekri and Akaichi, 2014) to use the two generic parameters "gap_{time}" and "gap_{space}". gap_{time} presents the maximum time allowed to connect the existing trajectory last point (L) to P_{new}. The same for gap_{space} that pressents the maximum distance allowed to connect L to P_{new}. Exceeding the gap_{time} and gap_{space} creates a new trajectory T_{new}, and P_{new} presents the starting point of T_{new}. Here, we spoke about the *trajectories organization*. This step is ensured by the function ***check_point_state (P_{new}, L, gap_{space}, gap_{time})*** which verify the *distance* between P_{new} and the last point L of the existing trajectory T. If this *distance* is lower than the gap_{time} and gap_{space} then P_{new} will be connected to L. Otherwise, P_{new} becomes the starting point of a new trajectory T_{new}.
- Once P_{new} satisfies the conditions presented above and it is added to the existing trajectory T that describes a mobile object MO activities, the function ***check_linearity (L.LastPoint, L, P_{new})*** checks if P_{new} forms a straight segment with the two last points that are just before P_{new}. If so, the point in the middle L will be removed since the MO has already passed through this point while connecting P_{new} to the trajectory T. Else, P_{new} will be connected to the last point L of the trajectory T, normally, without any remove. Here, we spoke about *trajectories cleaning*.

In case the MO has not been processed yet, the algorithm creates a new trajectory T_{new} for the new MO and P_{new} becomes the starting point of T_{new}.

This algorithm will stop runnning when all existing points are processed. This is cheked by the function ***check_point_occurence()***.

```
Algorithm Trajectory construction
1. Point Pnew ← Extract_Point (x, y, t)
2. Do
3.    If Trajectory.contains (MO) then
4.       Point L ← Trajectory.lastPoint
5.       If (check_point_state (Pnew, L, gapspace, gaptime)) then
6.          Trajectory.add(Pnew)
7.          If (check_liearity(L.LastPoint , L, Pnew)) then
8.             Trajectory.remove (L)
9.          End if
10.      Else
11.         Trajectory Tnew = new trajectory ()
12.         Tnew.add(Pnew)
13.      End if
14.   Else
15.      Trajectory Tnew = new trajectory ()
16.      Tnew.add(Pnew)
17.   End if
18.While (check_point_occurence())
```

Figure 2: The trajectory construction algorithm.

Once the *trajectory construction* process is achieved, the obtained trajectories are stored into the TDB as a TDSrc.

As T1 is well performed, T2 takes place to continue the whole transformation task. T2 focuses on creating the target area models that consists of the TDMs models generated directly from the relational TDSrc model. This task translates the *bottom up* approach features wherein the mapping process plays an important role. Our goal is to identify how a target field could be generated from the source field. To this end, we have already defined a set of textual rules in (Azaiez and Akaichi, 2015) that results the generation of a set of TDM elements directly from TDSrc at the aim to analyze mobile objects trajectories obtained in T1. However, these textual existing rules are inadequate to achieve this task since their application in their state leads to waste time and this is contradictory to our goal. Hence, their translation basing on a transformation language becomes a must. As a powerful mechanism, Model Driven Engineering (MDE) is heavily building on model transformations in converting models. It is a general methodology for software development, which recognizes the use of recurrent patterns to increase compatibility between systems. Model Driven Architecture (MDA) is a specific proposition for implementing MDE. The MDA based development of a software system starts by building *Platform Independent Models* (PIM) of that system which are refined and transformed into one or more *Platform Specific Models* (PSMs) which can be used to generate source code. MDA approach is responsible for transforming the source model into target models. It provides a set of guidelines for structuring specifications expressed as UML models. Each model is abstracted by its meta-model which should be conformed to a meta-meta-model called Meta Object Facility (MOF) (OMG, 2006). Transformations have to be expressed by means of a transformation language as ATLAS Transformation Language (ATL) (ATLAS team, 2006). Figure 3 provides an overview of the ATL transformation (Relational2Multidimensional) that generates the multidimensional modeling, conforming to the meta-model (MMTDM), from a TDSrc model that conforms to its meta-model (MMTDSrc). MMTDSrc, MMTDM and ATL metamodels are expressed using the MOF semantics.

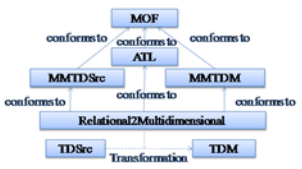

Figure 3: The MDA architecture.

The next figure presents the MMTDSrc.

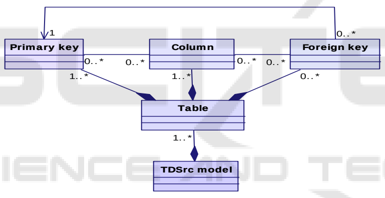

Figure 4: Source meta-model.

Figure 5 describes the MMTDM.

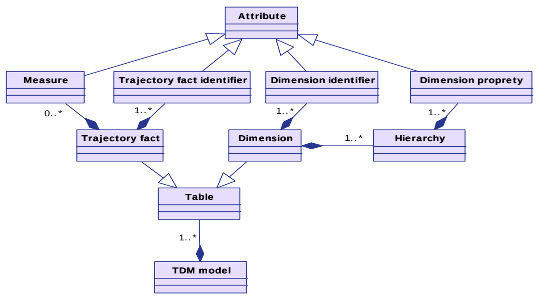

Figure 5: Target meta-model.

In MDA approach, ATL specifications play an important role to translate our proposed transformation rules defined in (Azaiez and Akaichi, 2015) in order to generate the target model. An ATL program is composed of a *header*, a set of side-effect free functions called *helpers* and a set of *rules* (Matched, called, lazy...) that are responsible to perform transformations. Figure 6 presents a little extract of the ATL code.

```
Module Relational2Multidimensional;
Create out: Multidimensional from IN: Relational;

Rule S2S { /* Schema to Schema */
from s : IN!TDSrc_Schema
to t : OUT!TDM_Schema (Tables <- s.Tables)}

Rule T2TF { /* Table to Trajectory_Fact */
from s : IN!TDSrc_Table
to t : OUT!TDM_Trajectory_Fact (name <- 'F'+'_'+s.name,
                                Measures <- s.Columns->collect(a|thisModule.C2M(a))
                                iskey<- s.iskey )}

Rule T2D { /* Table to Dimension */
from s : IN!TDSrc_Table
to t : OUT!TDM_Dimension (name <- 'D'+'_'+s.name,
                          Attributes <- s.Columns->collect(e|thisModule.C2A(e))
                          iskey<- s.iskey )}

Lazy rule C2M { /* Column to Measure */
from s : IN!TDSrc_Column
to t : OUT! TDM_Measure (name <- s.name)}

Lazy rule C2M { /* Column to Attribute */
from s : IN!TDSrc_Column
to t : OUT! TDM_Attribute (name <- s.name,
                         iskey<- s.iskey )}
   ...
```

Figure 6: Relational2Multidimensional.atl.

Our work focuses on the convergence of all TDMs that model organizational business events. Once more than two TDMs are built, the fusion of these brings about the creation of a central repository of data named TDW (Figure 1).

4 CASE STUDY: EPILEPSY PATIENT STATES

We propose to apply our approach to model a Trajectory Decision Support System (TDSS) related to persons suffered from epilepsy. This offers the possibility to analyze the epilepsy patient state evolution trajectory. Each year, thousands of patients die following a seizure of epilepsy. In this context, it is imperative that patients be able to access the careers to contact in case of crisis. The development of a new medical device called *EpiLert* solves the problem. It comprises a wristband that is based on a sensor capable of detecting the movements of the limbs associated with an epileptic convulsion to send automatically an alert signal to caregivers when a person is in crisis while it provides recordings of epileptic events and seizures data in order to obtain additional medical tests. In our case, extracting those data leads to collect a set of raw TD. Once the loading process is achieved, applying the *trajectory construction algorithm* (Figure 2) becomes a must. Each crisis provides a new trajectory belongs to such epilepsy patient. The obtained trajectories of all patients are stored into the TDB as a TDSrc model presented in the next figure. This presents the T1 goal.

Patient (id_pat, first_name_pat, last_name_pat, id_Epilert#, id_disease#)
Docteur (id_doc, first_name_doc, last_name_doc, id_service#)
Epilert (id_Epilert, color, id_device#)
Service Medical device (id_device, marque)
Hospital service (id_sevice, designation)
Disease (id_disease, name_disease)
Trajectory (id_traj, #id_move, #id_stop, #id_pat)
Stop (id_stop, #id_begin, #id_end)
Move (id_move, #id_begin, #id_end)
Begin (id_begin, Begin_time)
End (id_end, End_time)

Figure 7: TDSrc model.

Since the TDSrc model is presented, it becomes ready to be processed according to the second transformation step T2 which goals at generating the target models (TDM) from the TDSrc model basing on the MDA approach. Transforming models using MDA is related to the source model that must be conform to its meta-model and the target meta-model, recently defined, and the transformation rules defined textually in (Azaiez and Akaichi, 2015) and translated using ATL (Figure 6). TDSrc is a relational model which consists of a set of tables and columns that have to be transformed into a set of multidimensional elements (*facts, dimensions*...).

According to our case study, applying the ATL transformation rules leads to create a set of TDM models. For example, the table Trajectory in the TDSrc model feeds a trajectory fact table F_Trajectory in the multidimensional design since it satisfies conditions enumerated in the transformation rule that is destined to identify trajectory facts. The tables Patient, Move and Stop satisfy conditions enumerated in the transformation rule that is destined to identify dimensions. Therefore, they feed respectively, D_Patient, D_Move and D_Stop. The dimension D_Date is required in every multidimensional schema since it contains all the information we need about a certain date, and allows analysts to analyze data as accurately as possible.

5 CONCLUSIONS

In this paper, we presented an overview on solutions proposed by the research community to deal with the ETL modeling problem. We expected the absence of an ETL process that really leads to a better data analysis. Hence, we relied on the Trajectory ELT process to facilitate the propagation of the TD from Operational Trajectory sources towards Trajectory Warehouse area. Since the transformation task is the core of the Trajectory ELT process, we proposed a trajectory construction algorithm to transform raw positions into trajectories and, therefore, generate the TDSrc model. Then, we relied on the power of the MDA mechanism to transform the obtained source model into target models. We illustrated the efficiency of our approach using a medical use case. Currently, we are extending our framework to offer a system which handles easily the evolution of the whole warehousing chain while trajectory sources evolve; this ensures reaching the BI goals, especially extracting pertinent knowledge.

REFERENCES

ATLAS team, 2006. ATLAS Transformation Language (ATL) Home page. *http://www.eclipse.org/gmt/atl/.*

Azaiez, N., Akaichi, J., 2015. How Trajectory Data Modeling Improves Decision Making? *In proceedings of the 10th International Conference on Software Engineering and Applications.* Colmar, Alsace, France, pp. 87-92.

Azaiez, N., Akaichi, J., 2016. What is the Impact of Mobility Data Integration on Decision Support Systems' Modelling and Evolution? *The International Journal of Information Systems in the Service Sector (IJISSS), 8(1),* pp.1-12.

Boussaid, O. Bentayeb. F., Darmont, J., 2003. An MAS-Based ETL Approach for Complex Data. *In proceedings of the 10th ISPE International Conference on Concurrent Engineering: Research and Applications.* Madeira, Portugal, pp. 49-52.

Marketos, G., Frentzos, E., Ntoutsi, I., Pelekis, N., Raffaetà, A., & Theodoridis, Y., 2008. Building real-world trajectory warehouses. *In Proceedings of the International ACM Workshop on Data Engineering for Wireless and Mobile Access.* Vancouver, BC, Canada,pp. 8-15.

OMG, 2004. Model Driven Architecture (MDA). *http://www.omg.org/cgibin/doc?formal/03-06-01.*

OMG, 2006. Meta Object Facility (MOF) 2.0 Core Specification. *OMG Document formal /2006-01-01.*

Rifaieh, R., Benharkat, N. A., 2002. Query-based data warehousing tool. *In proceedings of the 5th ACM International workshop on Data Warehousing and OLAP,* pp. 35 – 42.

Simitsis. A., 2003. Modeling and Managing ETL Processes. VLDB Ph.D. Workshop.

Taktak, S., Alshomrani, S., Feki, J., Zurfluh, G., 2015. An MDA Approach for the Evolution of Data Warehouses. *The International Journal of Decision Support System technology (IJDSST); 7(3),* pp. 65-89.

Vassiliadis, P., Simitsis, A., Georgantas P., Terrovitis, M., 2003. A framework for the design of ETL scenarios. *In the proceedings of the 15th CAiSE*. Velden, Austria, pp 520-535.

Zekri, A., Akaichi, J., 2014. An ETL for Integrating Trajectory Data A Medical Delegate Activities Use Case Study. *In the proceedings of the International Conference on Automation, Control, Engineering and Computer Science,* pp.138-147.