Individual decision making based on a shared context

Hassane TAHIR\textsuperscript{a,b} and Patrick BREZILLON\textsuperscript{a}

\textsuperscript{a}Laboratory of Computer Science-Paris 6, University Pierre and Marie Curie (UPMC), Paris, France \{Hassane.Tahir, Patrick.Brezillon\}@lip6.fr,
\textsuperscript{b}Steria France, Energy & Utilities Department, 11 avenue du Maréchal Juin 92366 Meudon-la-Forêt Cedex France, hassan.tahir@steria.com

Abstract. The performance of data migration processes is an important issue when transferring data from existing source information systems to new target systems. Such a process is called ETL (Extraction, Transformation and Loading). Addressing this issue is one of the main tasks and challenges of database administrator (DBA). This activity is important because it must be accomplished as rapidly as possible to make the target system available to other systems on time. Under such a pressure, a DBA has to collaborate with several experts working on the different parts in the migration environment to solve the problem at hand. Exchange of information and knowledge between the DBA and project members becomes effective if there is a common interpretative focus and a shared context where all actors can understand each other and exchange their experiences. In this paper, we present how context sharing can help in individual decision making when dealing with ETL-processes failures, thanks to a platform called Contextual Graphs (CxGs-Platform) that allows a uniform representation of knowledge, reasoning and contexts. Our work provides a basis for the development of an experience base that will be used by a decision support system for DBA experts.

Keywords. Contextual Graph, Data Migration, DBA, decision support system, ETL process, experience base, Shared Context.

Introduction

Companies and organizations face the fast growing change in business and IT infrastructure services. As a consequence, they have to replace their existing information systems sources by applications with new functionalities to satisfy their needs and adequately address their challenges. However, data migration and integration from old sources to new target systems often is a critical mission because there are different risks when a bad strategy of data migration is applied. Risks can be financial (high cost), loss of important data, reduced performance, poor security of data, etc. (see examples in [9][10][14]). Database Administrators (DBAs) make many such strategic decisions every day that are based on a combination of judgment and information coming from staff, clients, research literature, as well as knowledge gained from previous projects and contextual knowledge shared with other actors cooperating with the DBA. Ideally, all relevant information should be brought together before judgment is executed.
Data is collected in large volumes like data warehouse using different techniques of Extraction, Transformation and Loading (ETL). The ETL process is an important phase in data migration during which the DBA can perform a variety of tasks including database design, performance monitoring and tuning, database availability, data integrity, backup and recovery.

The DBA must be constantly ready to identify, analyze and correct failures as well as readjust procedures to manage a multitude of specific situations that differ from the generic one by few contextual elements. For example, in database tuning a DBA must apply fastidious procedures with excessive checking of performance-related statistics and, thus, the DBA is unable to focus on what is really important and not aware of the context in which the problem occurs. The easy way is to apply some “simple” rules to tune their databases. Past experiences have shown that results are not always as good as expected. One reason is that all the rules used to check whether a given ratio (or value) is acceptable are defined independently of actor’s experience.

Interacting with other actors, a DBA develops practices that deal with shared contextual elements to find the requested information for problem solving during task realization. Exchanges of information and knowledge between the DBA and members of his organization can only be effective if there is a common interpretative focus and its associated context to understand each other and communicate effectively. All the elements flavoring understanding and communication are called the shared context. The concept of context plays an important role in collaboration, particularly in communication, interaction and knowledge sharing. Contextual knowledge depends on each member’s skills, experiences and background, and also on common activities, conditions, facts, and situations faced during the administration of the database and the setting of the environment.

The above motivation shows that a decision support system for DBAs must, on the one hand, have an explicit representation of the shared context, and, on the other hand, work on an experience base that capitalizes both individual and collaborative past experience. The decision support system depends crucially on the experience base. It contains the set of database management procedures that have been chosen and the practices that the DBA has developed to accounting for different contexts of application. This supposes that the experience base is developed in a uniform representation of elements of knowledge, reasoning and context.

Our work is a continuation of research on context modeling and management [5][6][7][8]. Research works in ETL processes include also conceptual modeling and implementation issues (e.g. see [11][12][13]). This paper relies on a Contextual-Graphs formalism [2] for implementing actors’ activities and actions according to the different contextual elements. The main advantage of Contextual Graphs is that the experience base will be enriched during the use of the system, thanks to the possibility to enrich incrementally the system with new knowledge and practice learning capability [2][3]. Moreover, a contextual graph is a good communication tool to helping (1) the DBA in taking his decision to solve complex database problems, (2) actors of the organization in easily exchanging their experiences and viewpoints with the DBA at the right granularity for each of them.

Section 1 presents a DBA activity showing the limits of existing systems for automating some of the database administration procedures in the ETL processes and the need to contextualize such procedures to build a real decision support for DBA (Database Administrator) Experts. Then, Section 2 presents our shared context approach in individual decision making. Section 3 develops the use of contextual
graphs in the modeling of an ETL process and the representation of a DBA procedure for solving a database problem such as performance tuning and how it is easy to add practices developed the DBA in collaboration with different actors to resolve a complex problem. Finally, Section 4 concludes and gives some perspectives of our work.

1. A problem in an ETL process

The database problem in ETL processes that is discussed in this paper is based on a current project in the enterprise of one of the author. It concerns data migration from existing source systems to a new target application. The application is a billing system that will be operated by an Energy supplier to satisfy new authority regulations and standards (see Figure 1). The application also should ensure customers to pay the right amount of energy and protect them from large unexpected bills and give the Energy supplier the incentive to get billing right every time.

![Figure 1. ETL Process in Data Migration.](image)

In the migration, we focus on performance problems encountered in the ETL process when contextual elements are not shared between the migration team members. Consider a new context in which some of the common steps must be updated or removed, and all actors should be aware of this change. We distinguish two parts. The first part concerns contextual elements relevant at a given time (e.g. cache size, state of workflow, response time). The second part addresses the values of these contextual elements at that moment: cache size (70%), full, state of workflow (running, finished, aborted, stopped), response time (excellent, good, bad). Migration members involved in the process must be aware of the contextual elements and their known values, but also they must agree on the instances (i.e. the values) of contextual elements corresponding to the situation at hand. The actors involved in the project are the Migration Manager, Business experts, Data Analysts, ETL Developers, Database Administrators (DBAs), Data architects, ERP Consultants and Testing Consultants. They have different roles and viewpoints about the different activities that they should carry out together. For
example in the extraction phase of the ETL design process, some examples of questions that may be asked by the migration members are:

- What is the response time? (i.e. excellent, good or bad)
- Are all parts of the ETL system causing bad performance identified? (i.e. code of the ETL Software or data injection programs, database, or network infrastructure)
- Are database backups rescheduled to support and take into account the new ETL workflow jobs durations and planning?
- Should indexes be dropped and recreated, respectively before and after each data loading and how? (i.e. manually, automatically)
- Is the DBA aware of the new ETL process constraints? (i.e. long duration of the new ETL jobs, fast growing data)

Such questions concern some of the different contextual elements that intervene in the data extraction phase of the ETL process (with their known values). For sharing contextual elements, data-migration actors must make explicit their viewpoints. Therefore it is necessary to coordinate all viewpoints and make them closer and compatible to each other to avoid bad system performance and failures. In other words, each actor has to determine the most relevant contextual elements and communicate the instance in the current situation to other actors. Our objective is to develop a shared context in which contextual elements considered by each category of the actors and accepted by all the other categories are made explicit: “Make explicit what is implicit”.

Response time is a crucial problem of data migration in our project, because data should be migrated in the new system in a period that should not be longer than the brief cutover period where the production environment is taken offline. Each ETL system depends on a Database Management System (DBMS), which is a set of subsystems executing specific tasks and compete for system resources allocated by the DBMS. There are two main tasks in DBMS performance tuning. The first problem is about “diagnosis” to determine which resources are responsible for the performance problem. The second problem is about “resource adjustment” (or “tuning”) that involves altering resource allocations to achieve better performance. Figure 2 shows a procedure for identifying the performance problems in an ETL application. The procedure consists of several steps that have to be executed in a specific order. For example, the first step is to perform an analysis of the ETL application code in order to determine if the response time problem is due to one of the three categories: application code, database or network (or all other components supporting the application).

Several calls for the automation of the diagnosis and tuning processes have been made in recent years [1] in order to allow the DBMS to manage its own resources and reduce the amount of time that a DBA must spend attending to a system. Unfortunately, the solutions that are suggested currently for automating the diagnosis process didn’t consider the working context that it is thus not shared by all actors. In the following, we show how Contextual-Graphs (CxG) Platform allows the improvement of the representation of this procedure by incremental addition of practices (as contextualization of the procedure) and illustrate the shared context between actors that cooperate together to help one actor in his decision-making. Note that all actors cooperate in the problem solving, but only one actor make the decision. The next section presents the shared context approach.
2. Shared Context

The knowledge needed by any business for making decisions effectively belongs to different actors of different services of the organization. Database-management decisions can be improved by an efficient collaboration between database experts (i.e. the DBA) and key individuals such as Developers, System Engineers, Project managers and Business users. Each actor has his own viewpoint of what is the best course of actions and the wished effects. One of the greatest challenges is to find a common
means to define and reach a target goal (fixed by one individual or actor) because the increasing number of actors from different backgrounds and domains affects the number of practices used for problem solving. In addition, when cooperation implies a large number of actors, some compromises are required to facilitate the cooperation. One way is to offer actors the opportunity to understand each other’s needs and use their capabilities to make their viewpoints closer to each other with respect to the target goal.

Group work supposes to manage the group context explicitly, not individual contexts only because the group context is not the sum of individual contexts. A group member needs to have some knowledge about other members, but also the context in which the knowledge he needs from others is operational. Thus, each member knows about the other but also is able to interpret and extrapolate others’ behavior. Our approach is based on context sharing to make actors’ viewpoints compatible and closer to each other in their different interactions [16][17] and collaboration [4][15]. A shared context is built from contextual elements introduce by an actor and accepted by others, i.e. assimilated in their own contexts. The shared-context building thus results from an incremental enrichment of contextual elements coming from individual contexts of actors. Thus, a contextual element proposed by an expert will enter the shared context if accepted (validated) by other experts.

The actor’s focus is guided by the mental representation that the actor has of the task in its current step, the situation in which he is and the local environment in which resources are available. The mental representation expresses what the individual context is. In our application, the shared context is built from elements coming from the mental representations of actors intervening directly or indirectly in the task realization. A contextual element provided by an actor must be: (1) extracted from the actor’s mental representation, (2) transformed in information to share, and (3) integrated in other experts’ mental representation, i.e. transformed in a piece of knowledge coherent in the new mental representation. This is the way to make the different viewpoints compatible among the participants in the collaboration, and not necessarily identical because all mental representations are different. In the example of Figure 3, the DBA shares context (oval intersections) with all other actors.

![Figure 3. Shared Context in the ETL process.](image-url)
Making focus explicit in a task realization allows the refinement of its context in three parts: external knowledge, contextual knowledge, and proceduralized context [3]. The external knowledge is the knowledge that has nothing to do with the focus at its current step. The contextual knowledge is the knowledge that is more or less relevant for the current focus. The proceduralized context is the part of the contextual knowledge (i.e., a subset of contextual elements) that is explicitly used by the actor to address the task realization at the current focus. The proceduralized context is an ordered sequence of instantiated contextual elements that is invoked, organized, structured and situated to be used at a given focus of the task realization. The shared context is related to the current focus of attention of actors in the data migration. It is composed of all the proceduralized contexts that have been built during (and even previously) the task realization (i.e., for all the successive steps of the focus).

This conceptual framework takes its meaning through the context-based formalism of representation that is associated to him, namely the Contextual-Graphs formalism [2] [3]. The following section explains how Contextual-Graphs formalism is used to implement the shared context between actors involved in the ETL process of the data migration.

3. Contextual-Graph Representation for DBA’s Decision Making

3.1. Presentation of Contextual Graphs

A contextual graph (CxG) is the key element to enhance the modeling of decision-making support in database management and administration. A CxG is the unified representation of the different practices (i.e., the paths in the graph) that have been developed for realizing a task (see Figure 4). It is an acyclic directed graph with a one input, one output, and a serial-parallel organization of nodes connected by oriented arcs. A path (from the input to the output of the graph) represents a practice (or a procedure), a specific task realization with the application of methods selected according to the context at hand. There are as many paths as practices. A contextual graph is an acyclic graph because user’s tasks are generally realized in ordered sequences of actions.

Figure 4 shows a contextual graph that represents collaboration between actors interacting with the database administrator (DBA) in his identification and solving of performance problems. The Developer cooperates with the DBA to help him to understand parts of the ETL application code causing the performance problem. The Developer takes into account all recommendations and required updates for SQL queries and PL/SQL procedures in the ETL transformations created by the Developer.

There are four items in a contextual graph: action, contextual element, activity and parallel action grouping. In Figure 4, actions and contextual elements are, respectively, represented using “green squares” and “bleu circles”.

- An action is the building block of CxGs (at the chosen granularity of description of the task realization), which can appear on several paths.
- A contextual element is a pair of nodes, a contextual node and a recombination node. A contextual node has one input and N outputs (branches) corresponding to the N instantiations of the contextual element already encountered. The
recombination node is \([N, 1]\) and shows that even if we know the current instantiation of the contextual element, once the part of the practice on the branch between the contextual and recombination nodes corresponding to a given instantiation of the contextual element has been executed, this knowledge does not matter anymore. At the conceptual level, this means that a new proceduralized context is built without this contextual element that goes back to the contextual knowledge with the previous proceduralized context.

- An activity is a sub-graph that is identified by actors as a specific subtask that is considered on different paths of a CxG or in different CxGs. An activity is itself a contextual graph that corresponds to a task realization under the responsibility of another actor. Thus what is a simple action for the DBA (e.g. what is the execution plan for a query?) could be a complex task for the developer (e.g. Need to create indexes? What is the Status of database instance?). Thus a contextual graph gives a representation of the reasoning that is directly understandable by database administrators.

- A temporal branching (not used in this example) expresses the fact (and reduce the complexity of the representation) that several groups of actions must be accomplished but that the order in which action groups must be considered is not important (or depends of elements at a too fine granularity), or even could be done in parallel, but all actions must be accomplished before to continue.

![Figure 4. Contextual Graph for the Developer interacting with a DBA (Developer’s viewpoint)](image_url)
The next section shows how shared context help in a “Two by Two” collaboration between the DBA and a Developer.

3.2. Illustrating Sharing Context

The example in Figure 5 discusses the collaboration between the database administrator and the developer to resolve the application performance problem. The two actors cooperate to optimize parts of the database code used by the ETL application especially the one including SQL Queries and PLSQL procedures. In particular, the DBA must give a feedback to the Developer about whether or not the optimization is performed successfully or not and if a solution is found to the bad system performance and poor response time. In the other hand, the Developer should take into account all the DBA recommendations and required updates in the ETL transformations he has created.

Figure 5. Contextual Graph for the DBA interacting with a Developer (DBA’s viewpoint)

The above contextual graphs in Figures 4 and 5 show that actors share a set of contextual elements (CE) to make understandable their viewpoints by the other. Table 1
compares the values of some of the shared CE (i.e. Optimization Status, Response time status, Nature of optimization). Notice that each shared CE may have different values from one actor to the other. For example, the response time status is “KO” for the Developer whereas it is “Good” for the DBA. When the DBA understands that the response time is KO from the developer viewpoint he will try other techniques to process the performance problem. He may also ask the developer for other contextual elements (i.e. environment and configuration parameters) that may be determinant for finding the failure causes. On the other hand, the Developer needs to know the nature of optimization the DBA has performed. For instance, the DBA must communicate the SQL queries he has optimized in order for the Developer to take them into account in the different parts of the ETL application code especially in future releases. Therefore the shared contextual elements values are very important and may be determinant for actor (i.e. DBA) to take the best decision for solving a critical problem as the one presented in the case study.

<table>
<thead>
<tr>
<th>Shared Context</th>
<th>DBA Viewpoint (Figure 4)</th>
<th>Developer Viewpoint (Figure 5)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Optimization Status</td>
<td>In progress</td>
<td>Completed</td>
</tr>
<tr>
<td>Completed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Not applicable</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Response time status</td>
<td>Excellent</td>
<td>OK</td>
</tr>
<tr>
<td>Good</td>
<td></td>
<td>KO</td>
</tr>
<tr>
<td>Bad</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Nature (or kind) of Optimization</td>
<td>Add Indexes</td>
<td>Add Indexes</td>
</tr>
<tr>
<td>Optimize/Update SQL Queries</td>
<td>Optimize/Update SQL Queries</td>
<td></td>
</tr>
<tr>
<td>Optimize/Update PL SQL proc.</td>
<td>Optimize/Update PL SQL proc.</td>
<td></td>
</tr>
<tr>
<td>Software script update</td>
<td>Don’t Know</td>
<td></td>
</tr>
</tbody>
</table>

In the above example, we have illustrated that it is so easy to use contextual graphs to represent both the initial procedure and the new collaboration practices that have been developed by the DBA during his interaction with other actors (i.e. Developer). These practices consider the various shared contextual elements about situations the DBA faced. The shared approach discussed in this case study will be used to build an experience base for a decision support system to assist DBAs in their decision making when performing complex database administration tasks. It can also be extended to several other computing areas such as monitoring systems, computer security and network management.
4. Conclusion

The paper presents the use of contextual graphs (CxGs) to represent actor’s activities for solving complex problems when running ETL processes. The work is based on the notion of shared context that has been used in many applications particularly in collaborative work in software design. We show how making shared context explicit can help in individual decision making (i.e. DBA actor) during a collaboration and, thus, avoiding conflicts between actors.

This paper has shown that Contextual Graphs is an efficient tool for representing different actors’ viewpoints during the solving of complex database problems, particularly when running ETL processes (but we also have used this formalism for other problems in our applications). We have also shown on a concrete case study the importance of context sharing between actors in improving the DBA decisions. Our future work will continue by studying the following aspects:

1) The extension of CxG use for modeling all actors’ viewpoints and thus have an improvement of the collaboration among the actors in the different parts of the ETL process.
2) The exploration of interactions between CxCs that represent different viewpoints and their consequences at a meta level during a collaboration.
3) The design and implementation of a context-based decision support system that uses an experience base to help the ETL experts and data migration actors in general.
4) The generalization of the context-based decision support system to other domains of applications.

References

[10] IBM, Best practices for data migration.


