Self-Healing Data Exchange Process Under Evolving Schemas: a New Mapping Adaptation Approach based on Self-Optimization

H. Assoudi, H. Lounis
LATECE, Département d'Informatique, Université du Québec à Montréal
Case postale 8888, succursale Centre-ville, Montréal QC H3C 3P8, Canada
assoudi.hicham@courrier.uqam.ca, lounis.hakim@uqam.ca

***Abstract*—Today, more than ever, enterprises are relying on highly complex IT solutions to respond flexibly and rapidly to the constant changing business environment. Yet, the increasing complexity of IT solutions presents significant challenges. In this paper, we propose a solution to reduce the human intervention needed to maintain data exchange processes after a schema evolution (changes impacting source or target system schemas participating in a data exchange scenario). Our approach, toward reliable self-healed data exchange processes under evolving schemas, is called DEAM (Data Exchange Autonomic Manager).**

Keywords-data exchange, autonomic computing, self-managed systems, dependability, fault tolerance, sufficient correctness, schema matching, schema mapping, mapping adaptation, machine learning.

I. Introduction

Among the IT solutions for which the need has steadily increased over the years, we find data exchange. The need for data exchange has become more pronounced in recent years with the emergence of new technologies and paradigms promising cost reductions by preserving existing IT investments (modernizing silo-based and monolithic applications).

Data exchange is the problem of finding an instance of a target schema, given an instance of a source schema, and a specification of the relationship between the source and the target [1]. Data exchange is strongly related to two distinct but complementary concepts: schema matching and schema mapping [2-7].

As data exchange becomes more prevalent, and as more data is exchanged between more applications, manually maintaining mappings (even simple mappings) after each schema evolution (schema changes) is impractical. Effective support for schema evolution is challenging since schema changes may have to be propagated, correctly and efficiently, to the deployed artifacts [8]. Maintaining the consistency of mappings under schema changes by finding rewritings that try to preserve as much as possible the semantics of the mappings is called Mapping Adaptation [9, 10].

In this paper, we propose a self-healing data exchange approach, called DEAM (Data Exchange Autonomic Manager), to answer the following question: How can data exchange processes be resilient to schemas changes (Schema Evolution)?

At the core of our approach, we find the combination of ideas from the field of autonomic computing and the field of sufficient correctness:

- Autonomic computing idea is based on the analogy with the nervous system (central and autonomic) of the human body which adapts itself to many situations automatically, without explicit human intervention [11-15].
- Sufficient correctness idea argues that everyday software must provide cost-effective service with reasonable amounts of human attention. Dependability for the everyday needs arises from matching dependability levels to actual needs, achieving reasonably low failure rates at reasonable cost [16].

We believe that combining these two ideas is a key to answer the question of resilience for data exchange solutions (recovering automatically from failures due to schema evolution).

II. Data Exchange Autonomic Manager

The role of DEAM is not only to recover automatically from mapping degradation without any human intervention, but also to do it with maximum of reliability (self-healing plan generated to rewrite broken mapping does not need any human validation), and the minimum of the availability downtime (self-healing process should not keep the data exchange process, which is a real time process, down for a long period of time). To do so, we introduced, as part of the decision making cycle of DEAM, a new concept called self-optimized mapping adaptation, based on "service-level utility" function.

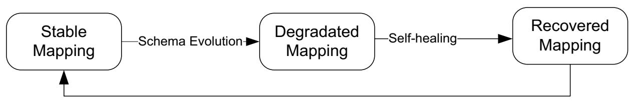

Figure 1. State diagram of a self-healed Data Exchange mapping

The novel idea on which DEAM is based is the use of an appropriate optimization technique, in conjunction with an appropriate mapping adaptation technique to achieve the self-healing. We think that using self-optimized mapping adaptation approach based on a service-level utility function is an efficient way to cope, in the same time, with reliability and availability downtime that a self-healing approach, in the context of data exchange, could have.

Injecting self-healing capability to a data exchange process will be achieved by having an autonomic manager monitoring in real-time the data exchange processes, and in case of failure, performing all the steps needed to recover from the faulty situation, and bringing the data exchange process to a normal state with maximum reliability and minimum downtime.

Our autonomic manager (DEAM) will have to perform the following steps as part of its self-healing process:

1. Detect a failure at the data exchange process and translate it to a standard event format.
2. Analyze the event, diagnose a mapping degradation issue (correlation rules) and isolate the broken mappings. During this step the original and the evolved schemas will be compared to capture the changes and to classify them into a set of primitive actions. An example set of primitive actions [9] includes (1) adding/deleting elements, (2) merging/splitting elements, (3) moving/copying elements, (4) renaming elements, and (5) modifying constraint. Note that the modifying constraint will not be relevant primitive actions for DEAM.
3. Plan a mapping adaptation by selecting, based on Knowledge Base, the best mapping adaptation approach. The chosen mapping adaptation plan will have to give the insurance that the broken mapping corrections (rewriting) can be applied without the need of an expert having to go through them and verify their correctness.
4. Apply the mapping adaptation plan. During this step DEAM will have to modify the data exchange artifacts such as schema definition artifacts and mapping rules artifacts (transformations).

The availability downtime is a critical factor when talking about fixing issue related to real time processes. Hence, the idea of having a self-healing data exchange process that can detect, diagnose, and recover from degraded mapping issue appears as the ideal way to have a resilient process that can be back on track, after a schema change in a timely-efficient and reliable way. Figure 2 shows the high level architecture of DEAM.

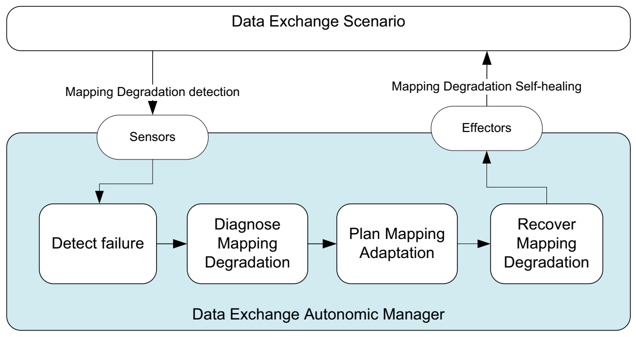
Data Exchange Scenario
Mapping Degradation detection
Mapping Degradation Self-healing
Sensors
Effectors
Detect failure
Diagnose Mapping Degradation
Plan Mapping Adaptation
Recover Mapping Degradation
Data Exchange Autonomic Manager

Figure 2. DEAM's high level architecture

We said that during the planning step, DEAM should make sure that the new generated mappings are reliable. So how can DEAM insure that? We also said that one of the critical factors for this self-healing process is the availability downtime which should be kept all the time to the minimum.

Although a lot of progress has been made in the field of mapping adaptation techniques (and all the other related techniques such as schema matching, schema mapping and mapping generation), yet seeking a very high accuracy level from an automatic matching/mapping process could still be a time consuming task. Of course, it's a trade off compromise, the more accuracy we want, the more it cost. But in the context of DEAM, we need to have both. In fact, DEAM should offer a high degree of confidence that the on the fly generated mappings are correct without having the need to perform any post-validation from human expert. In the same time, DEAM self-healing process should be performed within the time constraint imposed by the real-time nature of the data exchange process. In order to answer this trade off dilemma, we introduced a new concept called: Self-Optimized Mapping Adaptation.

III. SELF-OPTIMIZED MAPPING ADAPTATION

Ensuring the reliability of the entire data exchange process mapping adaptations can be reached by ensuring the reliability of mapping adaptations for the critical attributes (subset).

We think that our autonomic manager DEAM during its decision making cycle, should decide, based on the criticality of the attributes, whether or not a high degree of accuracy should be reached. In other words, the self-healing process should invest a high amount of time, sacrificing the availability of the process, to find the most correct adaptation for the degraded mapping only if this degraded mapping is related to a critical piece of information.

To decide about the criticality of a schema attribute in the context of a given data exchange scenario, we can say that a human expert can give a criticality factor (e.g. high, medium, low) to each attribute of the schema during the design time, and during runtime the autonomic manager can query it's knowledge base to find this information. But, the problem is that under schema evolution attributes are meant to change, for instance, they could be renamed, thus the initial effort of tagging schema attributes with a criticality factor will become gradually obsolete after each schema evolution.

In our opinion this criticality based optimization, that should happen during the analysis and planning phases of the autonomic manager's control loop, should be done in accordance with high-level guidance from a human expert. We think that the best way to implement this high-level guidance is classifying the schema attributes into high-level concepts and then associate the criticality factor to them. The initial classification effort will be done by a human expert, and then maintained under schema evolution by the autonomic manager.

Although this classification could seen as similar to the categorization approach implemented by Madhavan, Bernstein et al. [6] for the schema matching tool cupid but we think that our process of classification need more precision thus basing it only on "linguistic similarity" it's not enough. We see our classifier as a machine learning based approach that would learn how to map accurately new attributes to concepts by inferring patterns, not only from the attribute names, but also from the attributes schema instance historical data.

A schema could be involved in numerous data exchange scenarios, for this reason the service-level utility for each attributes classification and the service-level utility of each class (concept) is data exchange scenario specific.

Each concept (class of semantically related attributes) has a service-level utility function specifying the business value of providing a given level of service to users of a given data exchange scenario. In fact, the utility function should reflect the terms of service-level agreements with users, such as levels of acceptable accuracy, availability downtime ranges, etc. The table bellow describes high level self-optimization rules for mapping adaptation scenarios.

TABLE I. HIGH LEVEL SELF-OPTIMIZATION RULES FOR MAPPING ADAPTATION

Primitive actions	**Concept related**	**Service-level utility**	**Comment**
Adding	No	N/A	No degradation for the mappings
	Yes	N/A	
Deleting	No	downtime	Remove degraded mapping
	Yes	Error	No attribute available to adapt the mapping
Merging, Splitting, Renaming	No	Downtime	Select the appropriate matcher for minimum downtime
	Yes	Accuracy	Select the appropriate matcher for maximum Accuracy
Moving, Copying	No	Downtime	Select the appropriate matcher for minimum downtime. The attribute names remain the same.
	Yes	Downtime	

The following algorithm depicts the high level steps of DEAM's planning phase.

```
- For each degraded mapping do:
    o New attributes Classification
        ▪ based on the primitive action, decide if the new attribute(s) should be mapped to an existing concept.
        ▪ If Yes then find the concept to which the new attribute(s) should be associated.
    o Utility Value Resolution
        ▪ decide which service-level the mapping adaptation process should give
    o Matcher Selection
        ▪ decide which matcher will meet the service-level agreements
```

IV. CONCLUSION

In this work, we have tried to show how data exchange processes could become resilient under evolving schemas. In future work, we will explore and work on different machine learning approaches and algorithms in order to develop the most lightweight and efficient way to classify new attributes to existing concepts. Also, to develop an algorithm, e.g. based on correlation rules and machine learning, to automate the steps of diagnose and analysis of mapping degradation symptoms. Other self-management properties, such as self-protection, could be considered for DEAM. In fact, we are also looking to answer how data exchange processes could become self-protected in order to prevent potential anomalies or failures due to bad data.

REFERENCES

[1] M. Arenas, and L. Libkin, "XML data exchange: consistency and query answering." pp. 13-24.

[2] R. Fagin, P. G. Kolaitis, and L. Popa, "Data exchange: getting to the core." pp. 90-101.

[3] A. Gal, "Evaluating Matching Algorithms: the Monotonicity Principle," 2003.

[4] A. Doan, P. Domingos, and A. Halevy, "Reconciling schemas of disparate data sources: A machine-learning approach." pp. 509-520.

[5] H. Nottelmann, and U. Straccia, "Information retrieval and machine learning for probabilistic schema matching," Information Processing & Management, vol. 43, no. 3, pp. 552-576, 2007.

[6] J. Madhavan, P. Bernstein, and E. Rahm, "Generic schema matching with cupid." pp. 49-58.

[7] E. Rahm, and P. Bernstein, "A survey of approaches to automatic schema matching," the VLDB Journal, vol. 10, no. 4, pp. 334-350, 2001.

[8] M. Hartung, J. Terwilliger, and E. Rahm, "Recent advances in schema and ontology evolution," Schema Matching and Mapping, pp. 149-190, 2011.

[9] Y. Velegrakis, R. J. Miller, and L. Popa, "Preserving mapping consistency under schema changes," The VLDB Journal, vol. 13, no. 3, pp. 274-293, 2004.

[10] Y. Velegrakis, R. J. Miller, and L. Popa, "Adapting mappings in frequently changing environments," 2003.

[11] Z. Zhenxing, G. Congying, and D. Fu, "A survey on autonomic computing research." pp. 288-291.

[12] M. Salehie, and L. Tahvildari, "Autonomic computing: emerging trends and open problems," SIGSOFT Softw. Eng. Notes, vol. 30, no. 4, pp. 1-7, 2005.

[13] H. Müller, H. Kienle, and U. Stege, "Autonomic Computing Now You See It, Now You Don't," pp. 32-54, 2009.

[14] J. O. Kephart, and D. M. Chess, "The vision of autonomic computing," Computer, vol. 36, no. 1, pp. 41-50, 2003.

[15] A. G. Ganek, and T. A. Corbi, "The dawning of the autonomic computing era," IBM Syst. J., vol. 42, no. 1, pp. 5-18, 2003.

[16] M. Shaw, "Everyday dependability for everyday needs."