V-DIF: Virtual Data Integration Framework

Ali Zidane El Qutaany\textsuperscript{A}, Ali Hamid El Bastawissy\textsuperscript{B}, Osman Hegazi\textsuperscript{A}

\textsuperscript{A}Information System Department, Faculty of Computers and Information, Cairo University, Egypt.  
\textsuperscript{B}Faculty of Computer Science, MSA University, Giza, Egypt

Abstract

Data Integration is the process of combining data residing at homogeneous, autonomous, and heterogeneous data sources, and providing users with a unified global schema GS. Users pose their queries in terms of the GS, and they expect accurate, complete and unambiguous answers. Data integration system processes users' queries transparently, by translating each query to a set of sub-queries over the participating local sources LSs through the mappings defined between the GS and LSs. Even if none of the participating data sources have internal inconsistencies; mutual inconsistencies appear in the answers of the users' queries due to the integration process. To ensure the unambiguity in answers, the data integration process should be followed by detecting and resolving such inconsistencies. Most of the data integration frameworks introduced in the literature concentrate mainly on data integration process and avoid or ignore the other two processes (inconsistency detection and resolution). A few frameworks consider detecting and resolving the inconsistencies but don’t consider the interfacing or linkage between the three processes. Interfacing means each process tries to serve the successive process through preparing the parameters needed for such process. We developed a Virtual – Data Integration Framework (V-DIF) and tested it over 8 heterogeneous information sources. V-DIF meets most of the users' expectations. In this article the theoretical part of the framework is introduced to ensure the interfacing between the three processes.

Keywords: Data Integration, virtual integration, detectors, data fusion, duplicate and inconsistency detection, duplicate and inconsistency resolution.

I. INTRODUCTION

Data integration is the problem of combining data residing at different, heterogeneous sources and providing users with a unified view of this data, called integrated, mediated or global schema GS to be a reference for users to write their queries. A data integration system frees the user from the knowledge of where and how data are represented in the sources. The interest in this kind of systems has been continuously increased in recent years by the fact that many organizations face the problem of integrating data residing in several different sources \cite{1, 2}. The Materialized Data Integration term is used for combining data residing at heterogeneous data sources, and providing users with a unified materialized global schema, while the virtual data integration term \cite{3, 4} only provides users with a global schema and data are residing on the data sources \cite{5}. Users are posing their queries in terms of the global schema and expecting to receive accurate and unambiguous answers. Any effective data integration framework should be composed of three main basic processes; Data Integration (DI) process, duplicate/Inconsistency Detection (ID) process, and duplicate/Inconsistency Resolution (IR) process. The integration framework supposed to maintain the interfacing between these three successive processes. The integration process must take into consideration the parameters, weights, metadata, and qualifications \cite{6, 7} that assist in the other two processes (ID and IR). The main focus of this paper is to introduce the data integration process and preparing the environment for the two successive processes. The three main basic processes can be described briefly as follows:

1) Data Integration Process

In this process; data are integrated from many different data sources, these sources were created in heterogeneous and independent environment; therefore data quality problems are appeared \cite{8}. Data deficiencies appear in single data collection, such as files and databases. When multiple data sources need to be integrated, the need for data cleaning is increased significantly \cite{9}. This is because the sources often contain redundant data in different representations. Even if the sources are well clean and accurate and the data representations are unified across all data sources; some data quality problems are appeared due to the integration process itself called “due to integration quality problems”. One of these problems is mutual inconsistencies between data extracted from independent data sources which need efforts to be detected and resolved as functions of the successive processes to the integration process. The “due to integration quality problems” appear regardless the integration type we used either materialized or virtual integration. So DI process defines the GS, participating information sources and the mapping M between both. DI includes also the capability to translate the queries coming to GS using M to sub-queries over the data sources, then collects the answers from the sources to be sent to the next process; ID process.
2) Inconsistency Detection Process

In this process, duplicates are detected [10] in preparation to remove the ambiguities in the generated answers and to fuse inconsistencies before passing the answers back to the user. In this process each set of duplicates constructs one cluster with a specific Cluster ID.

3) Inconsistency Resolution Process

Here detected inconsistencies are resolved and/or fused [11, 12, and 13] before passing back the generated answers from the data sources to the users. Here there are 3 types of resolutions for the instances in the same cluster, either take one instance to represent the cluster using predetermined decision, take random instance to represent the cluster, or fuse the instances in the same cluster to construct one instance to be in the final answer.

In the literature, there are 3 different strategies [14] to deal with the inconsistencies, some researchers ignore the conflicts resolving process at all, this strategy called “conflict ignorance”, others are avoiding dealing with conflicts by defining a pre-determined decision to be taken in case of conflicts called “conflict avoidance strategy”, and the rest are trying to resolve/fuse the inconsistencies once detected called “conflict resolution strategy”. Even in the researches which use the “conflict resolution strategy”, the researchers do the data integration, conflict detection and/or conflict resolution processes almost separately, i.e. data integration is done, followed by conflict detection without full linkage or interfacing between the prior process (data integration process) and the same for the conflict resolution process.

In this work, the theoretical part of V-DIF is introduced to achieve the virtual data integration and to prepare for the ID and IR processes. The reminder of this paper is organized as follows, in section II three main components of data integration system are explored, section III introduces related work, while section IV introduces the theory of V-DIF showing how the three processes of data integration system are interfacing, while section V concludes the work and introduces the future work.

II. DATA INTEGRATION SYSTEM COMPONENTS

Data integration system is 3 main components; the global schema GS, the sources S, and the mapping M between GS and S. Thus, data integration system I is formalized in terms of a triple (G, S, and M), where [7, 15, 16, 17, 19]:

1. Global Schema GS

It is the integrated schema to represent the participating data sources with a specific business objective(s), it also called the mediated schema between the users and the data sources. GS either contains the integrated data from the sources (Materialized Integration) or it is only schema and the data are residing at the sources and queried through GS (Virtual Integration). Choosing Materialized or Virtual data integration is tradeoff, and it depends mainly on the objective of the integration process, when the main interest is the performance of the query answering process, then materialized integration is preferred. But when the main interest is to get the up-to-date answers, then Virtual Integration is preferred. V-DIF targets the virtual integration.

2. The Information Sources S

It is the “local Information sources” participating in the data integration process.

3. Mapping M

This is the most important component in the integration process especially in the virtual integration systems. Where the GS is mapped to S. M and GS usually grouped into a component called “MEDIATOR”. The following figure Figure1 shows the three main components of any data integration system.

![Figure 1: data integration system components][15]

Users’ queries are answered in the virtual data integration environment as follows:

1. A user poses a query Q in terms of GS [18],
2. The MEDIATOR applies a query-reformulation procedure over M to translate Q into executable N sub-queries qi to be executed over S, where N is the number of the participating/relevant data sources in the mapping M.
3. The MEDIATOR passes each qi to the WRAPPER of the relevant data source to be executed there and receives the answers.
4. The MEDIATOR collects returned answers from the wrappers and combines them (here the data quality problems due to the integration process appear).

Obviously, one of the main tasks in the design of a data integration system is to establish M between S and GS, such mapping should be suitably taken into consideration in formalizing a data integration system [19, 20, and 21].

Global as View (GaV), Local as View (LaV), and Both as View (BaV) are three approaches used to define M between GS and LS. Here a brief description about each of them is introduced,

---

[15]: Image of the data integration system components.
i. Global as View (GaV) technique

Mapping in data integration systems based on GaV as shown in Figure 2 (a). It associates global relation symbols with views over local relation symbols [19]. Query processing and simple query reformulation is the most important advantages to GaV. While changing any of the data sources structures require a GS designer to revise the GS and the mappings between the GS and source schemas [22].

ii. Local as View (LaV) Approach

The mapping in data integration systems based on LaV as shown in Fig 2 (b). It associates local relation symbols with a view over global relation symbols.

LaV has very good ability for scalability. But automating queries reformulation in LaV has exponential time complexity with respect to query and source schema definitions [19, 22].

Both-as-View (BaV)

BaV is an approach for mapping global schema elements to local schema elements. In BaV For each pair (LSi, GS) incrementally modify LSi/GS using primitive schema transformations [21] to match GS/LSi.

Practically, and when we tried such approaches; we faced many practical difficulties in query reformulations, mapping definitions, adding new data sources, change the GS defined, and others. So we used an enhanced GaV approach which solves most of these impediments.

III. RELATED WORK

Data integration is addressed in the literature in many researches such as:

1) Merging databases under constraints [22]

An operator is proposed for merging databases under constraints; such operator allows one to obtain a maximal amount of information from each database by means of a majority criterion used in case of conflict.

2) CONQUER Data Integration System [25]

A system for efficient and scalable answering of SQL queries on databases that may violate a set of constraints. ConQuer permits users to postulate a set of key constraints together with their queries. The system rewrites the queries to retrieve all (and only) data that is consistent with respect to the constraints.

3) Object fusion in mediator systems [26]

Data integration system with inconsistency resolution module where reliability degrees of the sources can be taken into account in the mapping definition, mapping views indicate how to resolve possible inconsistency

4) SQL-Base Conflict Resolution Technique [27]

It first identifies the objects from the different sources where there is different representation of the same real world object. Object identification is difficult, because the available knowledge about the objects under consideration may be incomplete, inconsistent and spares.

5) Conflict Tolerant Queries in AURORA [28]

Users can indicate in their queries the quality degree of the answer that they want to get; these indications are taken into account in case of conflicting data. However, no information on the quality of the data sources is considered when computing the answers to the query.

6) Tackling inconsistencies in data integration through source preferences [29]

An inconsistency resolution method that exploits information on source preferences for solving inconsistencies is presented, a method that allows one to assign formal semantics to a data integration system whose declarative specification includes information about source preferences, the semantics proposed in that paper is the first semantics for LAV data integration systems that take into account information on sources’ quality for dealing with inconsistent data.

7) HUMMER (HUMboldt MERger) Integration System [30]

HUMMER is a data integration system that contains a tool that allows ad-hoc, and declarative fusion of such data using a simple extension to SQL guided by a query against multiple tables.

8) FuSem (Fusion Semantics) [31]

It is a relational data fusion system which uses schema mappings and information about duplicates to decide what to fuse, i.e., which tuples to merge into one.

9) LDIF - A Framework for Large-Scale Linked Data Integration

LDIF can be used as a component within Linked Data applications to gather Linked Data from the Web and to translate the gathered data into a clean local target representation while keeping track of data provenance. LDIF contains a data quality assessment and a data fusion module which allow Web data to be filtered according to different data quality assessment policies and provide for fusing Web data using different conflict resolution methods [43]. Silk Link Discovery Framework [44] to find different
URLs that are used within different data sources which identify
the same real-world entity, also it uses Sieve Data Quality Assessment and Data Fusion Framework [45]. The Sieve data quality assessment module assigns each Named Graph within the processed data one or several quality scores based on user-configurable quality assessment policies.

10) Formal Model for Multidatabases (Multiplex)

It is a formal model for integrating multidatabases. Its principles are:
- GS is in the relational model R1… Rn,
- The GS is mapped to local schemas, by means of view pairs, each pair associates a view of the GS Vi with a view over a set of elements of the LS Ui.
- A query Q on the GS must be translated to a query over V1… Vm.

A global query is answered by multiple sets of records from participating data sources which almost contain mutual inconsistent data. An approximation for the true answer is provided by “sandwiching” it between two answers [6, 32] complete and accurate answers. Complete answer is the union of all fragments generated from relevant data sources, where accurate answer is the intersection between the fragments.

The proposed Multidatabases model has many limitations:
- No attempt at object identification
- It performs only the data integration and avoids the inconsistency handling.

11) An Inconsistency Resolution Methodology Using Data Fusion

A data integration system with inconsistency resolution module is addressed in [6, 7, 12], where a methodology for inconsistency resolution called Fusionplex is introduced. Fusionplex assumes the multidatabases model offered in [32] with some extensions, and tries to avoid the defined limitations of that model, where it computes answers to user queries by applying pre-defined conflict resolution policies defined with the GS based on the quality of data at the sources and further quality parameters provided by the user, it defines a policy for each GS attribute and uses these policies when facing conflicts [33]. But the problem is that it doesn’t consider any parameters for the duplicate detection process. After investigating all the trials in the literature, the following table 1 shows a comparison between the fragments.

The proposed Multidatabases model has many limitations:
- No attempt at object identification
- It performs only the data integration and avoids the inconsistency handling.

As the best of our knowledge: there is no data integration framework so far, takes into consideration the full interfacing between the three data integration system processes for data integrated from heterogeneous data sources. In the next section V-DIF is introduced.

IV. VIRTUAL DATA INTEGRATION FRAMEWORK (V-DIF)

Any effective data integration framework consists of three main processes; data integration process, duplicate detection process, and inconsistency resolution process. As shown in the previous section there is no data integration framework in the literature that achieves the three main processes of data integration framework along with full interfacing between the processes. V-DIF is a data integration framework that consists of the three processes and the full interfacing between them. V-DIF uses a store called Meta Fusion Information Repository (FMD) to keep all data that serves the three processes and their interfacing. Here and in this section V-DIF is detailed showing how FMD is serving the three processes and their interfacing.

1.1. Meta Fusion Information Repository / Fusion Meta Data (FMD)

FMD is a repository contains data about:

1) Data Sources S: data about the participating data sources in the integration process.
2) Global Schema GS: FMD contains the global schema relational constructs (schemes, tables,
relationships, attributes, and business constraints) in addition to the semantic correlations between attributes of GS, correlation between the attributes is useful for the inconsistency resolution process, as the correlated attributes must be fused together from the same data source if they were available together in the same data source.

3) **Detectors D**: For each global schema entity; a hierarchical set of detectors sets are defined and marked. Each detectors set consists of an attribute or group of attributes where domain expert expects they can identify the objects in case of duplicates. In the duplicate detection process V-DIF uses a hierarchical duplicate detection algorithm, which gives better results when the participating information sources in the integration process are not federated, and not all information sources agreed on the same set of detectors.

4) **Mappings M**: each global view Vi is mapped to a local schema view Ui; each entry in the mapping is called contribution. Here the enhanced mapping technique is used. Each global schema entity columns mapped to N views linked using union operator, where N is the number of local sources participating in the integration system.

5) **Local Schema Qualifications**

   Fusion or resolving of inconsistencies, maybe done using sources’ qualifications (meta-data based fusion) or data-content based fusion. The FMD stores the allowed metadata qualification along with the value of available qualification per source, Entity, Schema, and/or attribute level. Such qualifications may be calculated or extracted from the source metadata to be used in the fusion using metadata based fusion. FMD also stores fusion policies for the inconsistencies as described in 6).

6) **Global Schema Attributes’ Fusion Policies.**

   FMD stores fusion policies to be used in case of inconsistencies. The policies either defined based on the qualifications stored in FMD for the sources and their constructs, or based on the data itself to fuse the inconsistencies. The FMD has a repository for these policies, each policy constructed as follows

   - For AttributeList (mandatory)
   - Fuse From datasource
   - Fuse By Qualification
   - Keep What

   The first clause “For AttributeList” is mandatory. But only one of the 2nd, 3rd, and 4th clause is mandatory.

**1.2. Data Integration and Query Answering**

The GS is defined and the S, D, and M parts are fulfilled in the FMD. Then the main objective of this process becomes to get all data which could be in the answers of the user query from the data sources. Query answering process is done as follows:

a) Users pose their queries in terms of the GS,
b) Query Q is parsed against the GS,
c) The queried GS relations are extracted from Q to be used by the query translator to get the detectors sets D for each R from FMD,
d) Union between Q and D to construct Q*;
e) Determine the relevant mapping contributions to Q* (here only the contributions of the queried R),
f) Q* is translated to sub-queries {q1, q2, …qi) over GV1,
g) Each qi is executed on its own data source where it returns query fragment as an answer to qi, data in the query fragment are concatenated with the source name to mark the source of this data; source name aids the inconsistency detection and resolution processes. Returned source name helps the detection process to avoid detecting duplicates among data of the same source, while it helps inconsistency resolution process to fuse data of confident sources.

h) Do traditional union between query fragments to construct the Polyinstance data set (outer union was a common operator if we used any of the existing mapping approaches, but our own developed mapping approach for V-DIF simplifies the outer union to be traditional union). Polyinstance data set S with the Q will be the input of the second process (Duplicate detection process) w

**1.3. Inconsistency Detection**

The inconsistency detection process works as follows:

a) The inputs for this module are S and Q,
b) The duplicates are detected using the pre-determined detectors sets, as prioritized in the FMD. The first set is used first, and mark detected duplicate with clusterId, where it gives the tuples representing the same object the same value for cluster Id. And then the second set does the same, but using only the tuples with clusterId = null, and so forth for the rest of the detectors sets. N-1 rounds are done over the S to detect the duplicates where N is the number of the detector sets.

c) The output of this process is a set of real world objects represented by more than one tuple for each object,
d) Project only over Q + Data Source column
e) Pass the detected duplicates with their cluster Id to the inconsistency resolution process.
1.4. Inconsistency Resolution

The inconsistency resolution process achieved in steps as follows:

a) The input of this process is the detected duplicates with their cluster Id and Data Source column,

b) Exclude the tuples with null value for cluster Id, because they will be in the final answer to the user query.

c) For each attribute in the result set; query the FMD to get the correlated attributes with such attribute, and construct a set of vertical views, each view contains the attributes correlated to be fused together, and also contains cluster Id and the data source name (these 2 attributes are mandatory attributes in each view). If the underlying attribute does not correlated with any other attributes, it will be alone in the view with the mandatory attributes.

d) Fuse each attribute to one value using the attributes’ preferences and fusion policies defined in the FMD.

e) Join the views after fusion using the cluster Id.

1.5. Full Procedure for V-DIF

The following figure shows full architecture of V-DIF, and the full procedure of the V-DIF including the interfacing between the main 3 processes is shown in the following steps.

The V-DIF is processed as follows:

1) The GS designed and the FMD populated.

2) The users pose queries in terms of the GS, and the data integration process uses the FMD to answer the user queries to get answers for the inconsistency detection process, with the detectors, here the interfacing between data integration and inconsistency detection processes achieved.

3) The duplicate/inconsistency detection process uses the retrieved answers with the detectors to detect the duplicates and add cluster Id for each detected duplicates, and pass the detected duplicates with cluster Id to the inconsistency resolution process.

4) The inconsistency resolution process uses the duplicates passed from the inconsistency detection process, with the FMD to fuse the duplicates in terms of the cluster Id added by the detection process. Here the interfacing between the inconsistency detection and resolution processes achieved.

V. CONCLUSION

Users are posing queries to any data integration system and expecting accurate and unambiguous answers. Hence any data integration framework should consist of three main processes data integration, inconsistency detection, and inconsistency resolution in addition to their interfacing. None of the data integration frameworks in the literature performs these three processes with full interfacing, so V-DIF is introduced to perform the three processes and their interfacing through storing the required information in a metadata store called Meta Fusion Information Repository or Fusion Meta Data (FMD), this store is used by three processes as follows:

- **Data integration process** uses FMD for aiding data about data sources, global schema and their mapping. This process passes the retrieved raw answers with the detectors to the next process.

- **Inconsistency detection process** uses retrieved answers with the detectors passed from data integration process, then detect the duplicates to prepare the data for the inconsistency resolution process, it adds a cluster Id to the detected duplicates to be used in the resolution process.

- **Inconsistency resolution process** uses FMD to get the correlated attributes and the fusion policies of each attribute in the result set passed from the inconsistency detection process.

REFERENCES


[23] N. Leone et all, “INFORMIX System for Advanced Information Integration” SIGMOD ’05 Baltimore, Maryland USA, 2005


