

Ontological On-line Analytical Processing for Integrating Energy Sensor Data

¹Nazaraf Shah, ²Chen-Fang Tsai, ³Milko Marinov, ⁴Joshua Cooper, ⁵Pavel Vitliemov, ¹Kuo-Ming Chao

¹Department of Computing and the Digital Environment
Coventry University, UK
nazaraf.shah, k.chao@Coventry.ac.uk

²Department of Industrial Management & Enterprise Information, Aletheia University, Taiwan, R.O.C.
au1204@mail.au.edu.tw

³Dept. of Computer Systems & Technologies, University of Ruse, Bulgaria
MMarinov@ecs.ru.acad.bg

⁴ Hildebrand Technology Ltd., London, UK

jcooper@hildebrand.co.uk

⁵Dept. of Manufacturing, University of Ruse, Bulgaria
pvv@manuf.ru.acad.bg

Abstract:

In this paper we propose an ontological OLAP framework to integrate distributed energy sensor data. The OLAP data cube in the framework annotated with semantics with other supporting mechanisms can deal with the issues of the schema inconsistency that may result from integration of heterogeneous data sources. The proposed approach provides a way of storing, reusing and composing OLAP cubes in order to increase system usability. A prototype of the proposed framework based on a number of existing tools such as protégé, Jess, and FuzzyJ has been developed to demonstrate its feasibility.

Keywords: Distributed Data Warehouse, OLAP, Data Cube, Semantic Web, Hypertable

1. Introduction

Global warming is an important and serious issue faced by the modern world today. One of the main contributing factors to global warming is emission of CO₂. Increased industrialisation and our life style of relying heavily on increased number of household appliances and devices have drastically increased the energy consumption per capita. This energy consumption is directly linked to the emission of CO₂. Initiatives like United Nation sponsored Kyoto Protocol [60] which legally binds the member states to reduce greenhouse gases and European Climate and Energy Package agreed by EU to reduce greenhouse emission in EU by 20% by 2020 [61] came into existence aiming at addressing the issue of global warming.

These initiatives led to explore methods of reducing domestic energy waste and promoting efficient use of energy. One of such methods is to put consumer in control of their energy usage by allowing them to monitor their energy consumption in real time. Various energy monitoring devices started to find their way into market in order to implement this method.

Availability and affordability of sensing and transmitting devices for monitoring energy consumption in home environment are improving for the last few years. Many manufacturers, vendors and power utilities companies are engaged in manufacturing and promoting of different energy consumption monitoring devices in order to increase the awareness of efficient energy use. The amount of data generated from these sensors is large. Integrating these data and transforming them to useful information for decision making and analysis is a challenge.

This work is a part of our ongoing project The Digital Environmental Home Energy Management System (DEHEMS) [62]. DEHEMS is an EU funded Framework 7 project investigating ways in which state of the art technologies can be used to improve domestic energy efficiency and hence reduce CO₂ emission. The main objective of DEHEMS is to support households to reduce their energy usage through better management and analysis of their energy consumption. The first and fundamental step in achieving this goal is to have devices for monitoring the energy consumption of household appliances. In DEHEMS project we get energy consumption reading of appliances every six seconds. These readings are then sent to server, which then stores these reading in central database. The database maintains energy consumption record of hundred of households. The server receives six hundred readings per hour for each household.

In order to assess the energy consumption of household appliances, heating and cooling systems we need to have access to their energy consumption data. DEHEMS uses monitored energy consumption data to provide live feedback to household on their energy consumption and provide advice to household on efficient use of energy. Such functionality requires a large amount of energy consumption data of households to be stored and analyzed. In this paper we propose a semantic cube based mechanism for summarized data/information reuse and data analysis.

The paper is organized as follows in Section 2 we provide introduction to data warehouse. In Section 3 multidimensional DBMS and data warehouse is discussed briefly. Section 4 describes distributed data warehouse classifications and architectures. Section 5 presents distributed warehouse and associated data model. Section 6 gives an overview of semantic web in relation to data warehouse. Section 7 describes semantic web, semantic web and ETL (Extraction, Transformation, and Loading); and semantic web and OLAP. In Section 8 we provide details of our proposed approach and finally section 9 conclude the paper

2. Data Warehouse

Data warehouse can be generally described as a decision-support tool that collects its data from operational databases and various external sources, transforms them into information and makes that information available to decision-makers in a consolidated and consistent manner [1, 2]. A data warehouse organizes, analyzes and synthesizes huge amounts of raw data - so that it is intelligible and useful to the users [3]. A modern data warehouse is a collection of information stored in a way that improves access to data [4, 5, 6, 7, 8].

The best practices that were implemented as part of the successful data warehousing projects within various sectors (government agencies, military institutions and organizations directly supporting them) are presented in [9]. A generic framework for summarizing distributed data streams to be loaded into a data warehouse is proposed in [10]. The proposed approach limits the load of the central server hosting the data warehouse and optimizes the sampling rates assigned to each input stream by minimizing the sum of square errors. This optimization is dynamic and adapts continuously with the contents of the streams.

Data warehouses can be defined as subject-oriented, integrated, time-variant, non-volatile collections of data used to support analytical decision making [11, 2, 7]. Data warehouse collects all the data for intelligent decision making over a long period of time and hence contains a large amount of data. It

usually contains both detailed data and aggregated data. In terms of size it starts from a few gigabytes to several terabytes. This type of data warehouse requires lots of resources and expenses [4, 11, 12, 3].

The data integrated from the ETL (Extraction, Transformation, and Loading) process can be aggregated and stored in multidimensional cube in OLAP (On-Line Analytical Processing) for decision making. OLAP is designed to store data patterns and summarised information to rapidly answer some high-level questions about, for example, market forecasting and finance reports. A data cube is the core data repository in OLAP and its attributes could be structured in multiple dimensions for utilization.

It is essential that data in a data warehouse be easily accessible and understandable to the user. It must be simple yet intuitive, quick, and easy to use [8, 13, 14]. Also, the data warehouse must present the information consistently and securely to its users. When data is collected from the source systems, it must complete various measures of quality assurance to confirm its accuracy. The data must be verified, appropriately labelled, and fully accounted for before it can be made available to the users. Effective data warehousing can help create a meaningful relationship between information technology and business, facilitating enterprise-level strategic planning and growth [13].

3. Multidimensional DBMS and Data Warehouse

One of the technologies often discussed in the context of the data warehouse is multidimensional DBMS processing often called OLAP processing [4, 15, 16, 14]. Multidimensional database management systems, or data marts, provide an information system with the structure that allows an organization to have a flexible access to data, to slice and dice data in any number of ways, and to dynamically explore the relationship between summary and detailed data [16, 2, 8, 14].

A data warehouse holds massive amounts of data; the multidimensional DBMS holds at least an order of magnitude less data. A data warehouse contains data with a lengthy time horizon (from 5 to 10 years); the multidimensional DBMS holds a much shorter time horizon of data. It allows access to its data in a constrained fashion. Multidimensional DBMS allows unfettered access to its users.

One of the interesting features of the relationship between data warehouse and multidimensional DBMS is that data warehouse can provide a basis for a detailed data that is normally not found in multidimensional DBMS. The data warehouse can contain a fine degree of details, which are lightly summarized as they passed up to the multidimensional DBMS [16, 11, 7].

4 Distributed Warehouse Classification and Architectures

The primary objective of data warehousing is to bring together information from disparate sources and put information into a format that is conducive to making business decisions [9, 13, 20, 21, 18]. This objective necessitates a set of activities that are far more complex than just collecting data and reporting against it [18, 19, 3]. A distributed data warehouse (DDW) can be defined as a logically integrated collection of shared data that is physically distributed across the nodes of a computer network [20]. Distributed data warehouse is classified as follows [18]:

Business is distributed geographically: This type comprises of a local data warehouse and a global data warehouse. The local data warehouse represents data and processing at a remote site, and the global data warehouse represents that part of the business that is integrated across the business. The global needs for information are met by a central data warehouse where information is gathered, but there is also a need for a separate data warehouse at each location. In this case, a distributed data warehouse is needed, where data exists both centrally and in a distributed manner.

The data warehouse environment will hold a lot of data, and the volume of data will be distributed over multiple processors: Logically there is a single data warehouse, but physically there are many data warehouses that are all related but reside on separate processors. This configuration can be called technologically distributed data warehouse.

The data warehouse environment grows up in an uncoordinated manner: In this type many data warehouses appears with the passage of the time. The lack of coordination of the growth of the different data warehouses is usually a result of political and organizational differences. This case can be called independently evolving distributed data warehouse.

Zhao [14] presents an architecture of distributed data warehouse following general idea of data warehouses, i.e. the separation of input from operational databases and output to data marts. This idea is illustrated using the three-tier architecture .The operational database tier, the data warehouse tier and the OLAP tier. In dealing with distributed data warehouses he applies the relational theory of distribution design [11,3,21], i.e. first fragment the schema, then allocate the fragments to the nodes of a network. This allocation may store the same fragment at more than one node. The goal of the distribution is to optimize global performance. Having this goal in mind, fragmentation and allocation mutually depend on each other, and thus both steps are usually combined.

In [22] two types of data warehouse architectures are discussed – federated data warehouses and E-Enterprise warehouses. One of the problems with the centralized approach is that data in the warehouse is not synchronized with data residing in the underlying data sources. Although the goal of data warehousing had been to create a centralized and unified view of enterprise data holdings, this goal has not been fully realized. Many factors contributed to this, such as problem of semantic heterogeneity, and terminology conflicts, etc. However, one of the overriding factors has been the need for organizations to assert ownership over data. Organizations wish to own their data, wish to assume responsibility for the duration, or stewardship of their data, and wish to share their data according to well-defined sharing agreements. Thus, rather than spend large amounts of money on a centralized data warehouse, enterprises are implementing smaller data marts, and integrating them through federated architectures.

The trend away from the centralized data warehouse leads to the notion of a federated data warehouse, whereby independent smaller warehouses within the corporation publish selected data in data marts, which are then integrated. The federated approach is appealing in that common data can reside in a metadata repository, while rapidly changing data can reside in data marts or in the underlying data sources [23]. The e-enterprise is based on inter-enterprise partnerships among customers and suppliers. These partnerships are predicated on the sharing of data, information, knowledge through interoperable business processes, data sharing protocols, and open standards such as XML

In [24] the authors examine how a Distributed Heterogeneous Relational Data Warehouse (DHRD) can be integrated in a grid environment that will provide physicists with efficient access to large and small object collections drawn from databases at multiple sites. DHRD is a data warehouse technology. It is to implement the storage and access of data from warehouse to multiple sites of relational databases like MS-SQL and Oracle. This could reduce the need of massive central computing resources, network delays and provides the transparent local datasets access to the clients. The author proposes a web services framework based on grid services to integrate DHRD with the grid environment as a grid-enabled web services. This could enable the access of DHRD in distributed environment of various platforms.

In [12] Kanchi reviews and summarizes the basic types of data warehouse architectures (the classic Enterprise Data Warehouse, the incremental Architected Data Mart, Enterprise Data Mart Architecture, Data Stage/Data Mart, Distributed Data Warehouse/Distributed Data Mart, and the Federated Data

Warehouse/Data Mart), their pros and cons, and discusses the various approaches to build a data warehouse system, concentrating on two main approaches, top down and bottom up.

5. Distributed Warehouse Data Model

The heart of any data warehouse is its database, where all the information is stored [15, 25, 26]. Most traditional data warehouses use one of the relational products for this purpose. They can manage extremely large amount of data (hundreds of terabytes) mainframe relational databases; such as DB2 are used for some of the world's largest data warehouses. Universal data servers such as those from Oracle or Informix may be a good choice for medium-size warehouses because they manage a variety of data types. Multidimensional databases are becoming increasingly popular, but they limit the size of a warehouse to less than 5 gigabytes [27, 3].

In the record-oriented (relational) storage systems the attributes of a tuple are placed contiguously in storage. With this row store architecture, a single disk write suffices to push all of the fields of a single record out to disk. Hence, high performance writes are achieved, a DBMS with a row store architecture is called a write-optimized system [28]. In contrast, systems oriented toward querying a large amount of data should be read-optimized. Data warehouses represent one class of read-optimized system in which periodically a bulk load of new data is performed, followed by a relatively long period of ad-hoc queries. In such environments, a column store architecture, in which values for each single column (or attribute) are stored contiguously, should be more efficient. With column store architecture, a DBMS need only read values of columns required for processing a given query, and can avoid bringing into memory irrelevant attributes. In data warehouse environments where typical queries involve aggregates performed over large numbers of data items, a column store has a sizeable performance advantage.

A column-store stores each attribute in a database table separately, such that successive values of that attribute are stored consecutively. This is in contrast to most common database systems, "row-stores", where values of different attributes from the same tuple are stored consecutively. The following are some cited advantages of column-stores [29]:

Improved bandwidth utilization: In a column-store, only those attributes that are accessed by a query need to be read off disk (or from memory into cache). In a row-store, surrounding attributes also need to be read.

Improved data compression: Storing data from the same attribute domain together increases locality and thus data compression ratio (especially if the attribute is sorted). Bandwidth requirements are further reduced when transferring compressed data.

Improved code pipelining: Attribute data can be iterated directly without indirection through a tuple interface.

Improved cache locality: A cache line also tends to be larger than a tuple attribute, so cache lines may contain irrelevant surrounding attributes in a row-store. This wastes space in the cache and reduces hit rates.

It seems that data warehouses, with their batch writes, high bandwidth requirements, and query plans rife with table scans are well suited for column-stores. Physically storing and administrating the data warehouse as a distributed database helps in storage and access of large and small pieces of datasets. In the DEHEMS project the implementation of the distributed data warehouse will be based on the Hypertable DBMS. The Hypertable [31] data model consists of a multi-dimensional table of information that can be queried using a single primary key. The first dimension of the table is the row key. The row key is the primary key and defines the order in which the table data is physically stored. The second dimension is the column family. This dimension is somewhat analogous to a traditional database column. The third dimension is the column qualifier. Within each column family, there can be a theoretically infinite number of qualified instances. The fourth and final dimension is the time dimension. This dimension consists of a timestamp that is usually auto assigned by the system and represents the insertion time of the cell in nanoseconds.

Hypertable is an open source, high performance, scalable database, modeled after Google's Bigtable [30,31,32]. Hypertable is designed to manage the storage and processing of information on a large cluster of commodity servers, providing resilience to machine and component failures. We will adopt Hypertable for implementation of data warehouse.

6. Data Warehousing and Semantic Web

One of the key challenges in the distributed data warehousing is to model disparate data sources as a single conceptual model for OLAP and other analysis tools. These data sources have incoherent schemas, are located at different sites, and use inconsistent terms for naming data. The main process of the ETL is to extract distributed data sources, cleanse and transform them as well as load the pre-defined dimensions to be populated with the facts in order to build cubes for OLAP which is a

multidimensional data analysis for specific applications. The appropriate use of semantic technologies in the ETL process could alleviate some of the aforementioned issues. However, a semantic gap between advanced conceptual data model and relational or multidimensional implementations of data cubes still exists [34]. In addition, the structure of data cube requires flexibility and sufficient capabilities to re-organise/re-aggregate data to meet the needs of analysis and decision making tools at design time and runtime or unforeseen future requirements. Codd states that *OLAP is made up of numerous speculative “what-if” and/or “why” data model scenarios executed within the context of some specific historical basis and perspective* [35]. Traditional data engineering approaches may fall short of tackling this level of sophistication, as the design of the data structure is to meet the specific system requirements at a particular point in time. It has less concern on future requirements or other purposes. This requires another layer that is able to annotate these data and schemas and exhibit them, so these required data can be located and synthesized on the fly. Introduction of semantics to the data cubes that can be reasoned by inference engines could help to dynamically synthesize the data to answer this type of high level conceptual questions. The data cubes or data sources are annotated with semantics could also satisfy the intuitive data manipulation to support meaningful data aggregation. With the popularity of web and increasing interests in distributed data warehousing, the semantic web provides a promising technology to facilitate the development and deployment of data warehouse. It is foreseeable that the data warehousing research community adopts the semantic web technology to enhance the modelling and interoperability. Before a review on the existing research is given, the following section gives brief introduction to key concepts of semantic web and its current deployment.

7. Semantic Web

The Semantic Web [36] is increasingly seen as a powerful infrastructure to build sharable and reusable knowledge on the Web. Currently HTML is the language used to display graphics and text, but the contents it describes cannot be processed by the machine. The semantic web is to address this issue by introducing XML (Extensible Markup Language), RDF (Resource Description Framework), RDFS (RDF Schema) and OWL (Web Ontology Language) to describe web contents that enable automated information access and use based on machine-processable semantics of information and service. The theoretical foundation of semantic web is built upon description logics and graph theory. Ontologies are the core of the semantic web for the reuse of formalized knowledge [3, 37]. “An ontology is defined as a formal, explicit specification of a shared conceptualization” [38].

The semantic web layer cake shown in Figure 1 consists of a stack of evolving languages and shows key dependencies such as URI and XML which forms the foundation to the future development in proof and trust.

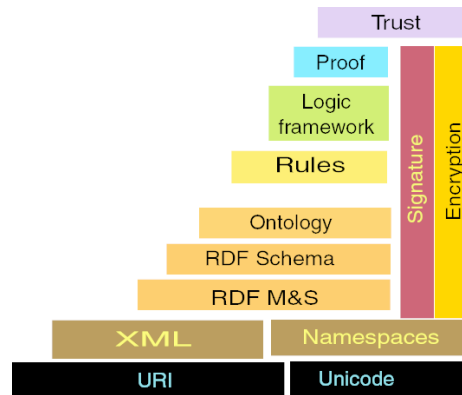


Figure 1: Semantic Web Layers

XML offers an elemental syntax to model document contents. RDF [39] provides a graph model to describe objects and their relations. RDF Schema [40] supports a simple vocabulary and axioms for describing properties and classes of RDF-based resources. OWL [41] adds more vocabulary for describing properties and classes in order to construct ontology.

With popularity and powerfulness of semantic web technologies, it can bring benefits to data warehousing development at different phases, as it can enhance the semantic of data sources, integrate heterogeneous schemas, automat ETL process and facilitate OLAP in data analysis etc. In recent years, researchers have proposed different approaches to bring semantic web and data warehousing technologies together in order to address semantic heterogeneity in distributed data warehousing. Manuel et al. [42] conducted a comprehensive review and analysis on existing work of integrating data warehouses with web data. In their article, they classified the existing research into three categories: use of XML technologies to integrate distributed data warehouse systems; development of data warehouse for semi-structured XML i.e. data-centric XML collections; and the development of data warehouses for unstructured XML which combine with information retrieval techniques to deal with document-centric XML collections. In this work, we are particularly interested in research which applies semantic web technologies to improve the interoperability among distributed data warehouses. This literature can be broadly divided into two groups: Semantic Web for ETL and Semantic Web for OLAP.

7.1 Semantic Web for ETL

Dayal et al [43] stated that the semantic web technologies can be used for (semi-)automatic generation of ETL, but there is a need to develop a systematic approach to support the development of ETL. They have proposed a QoX metric that can be considered as a set of criteria for optimisation to support different phases of ETL design from the business level through to the physical implementation level.

In [44] the authors argued that it is often the case that using natural language to document data warehouse application specifications and data source schemata that hinder the ETL process. As a result, they introduced semantic web technologies to the design process of ETL by providing application vocabulary to deal with naming scheme, annotating data sources and warehouses to select relevant information, generating application ontology to accommodate the needs of conceptual transformation among attributes, and generating conceptual ETL design automatically.

The work in [44] [45] share some similarities, as both attempt to facilitate the automation of ETL design process by using semantic web technologies. However, the latter one introduced a graph-based representation to model datastores (data sources and warehouses) that allow different types of schemas to be represented and processed in a consistent manner. The ontologies were created along with inference mechanism to facilitate the mappings between different datastores. This approach leads to semi-automatic generation of ETL data flow and reduction of efforts developing data warehousing applications.

7.2 Semantic Web for OLAP

To integrate heterogeneous data warehouse applications, semantic conflicts between local cubes with possibly different dimensional models may occur. Tseng et. al [46] analyse these possible semantic conflicts and classify them. The following reconstructed diagram (Figure 2) shows their classification scheme. According to this classification, the authors proposed a set of approaches based on XML and XQuery to transform and integrate these local data cubes to form a global cube.

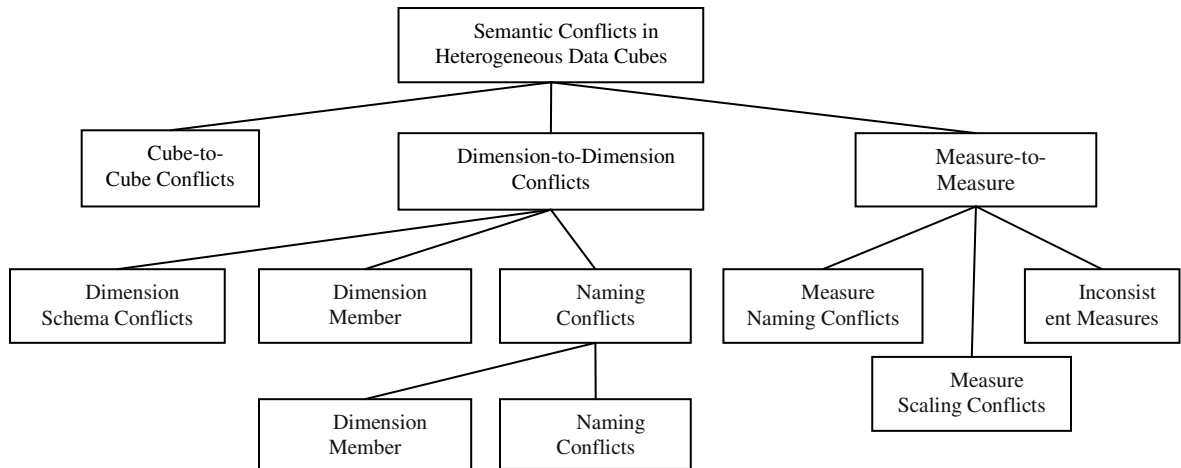


Figure 2. Semantic Conflicts in Data Cubes

According to the above classification, the cube dimensions, in general, are structured in hierarchical manners. The authors in [47] also identified the issues associated with semantic conflicts between local multidimensional cubes while the integration occurs. They attempt to address these issues by using XML Topic Maps (XTM) standard [48] to model metadata of multidimensional OLAP data. In the framework, each local data warehouse is described in a local XTM. An abstract layer which is a unified portal view at federated level in the global map with their associated mapping and transformation functions is proposed to alleviate their heterogeneity. However, the proposed approach is not as flexible as [46], as [46] has improved in the classification fineness in semantic conflicts.

In [49], a federated data warehousing framework is proposed to support interoperability of independent data warehouses using XML. It aims to present a collection of data warehouses to the users as a single data source by adopting a multiple layers approach to form a federation of integrating collection of distributed data warehouses. In the proposed multiple layers architecture, interoperability of data warehouses is realised by uniform schemas and query processing supported in the federated layer and the mediation layer. The federated layer provides federated schemas to allow merging of different data and schemas in order to build multidimensionality with a hierarchy structure. The modules in the mediation layer acts as a mediator to facilitate the integration between different data sources and the federated layer.

Niemi et al. [50] adopted a different approach to resolve the semantic conflicts in integrating heterogeneous data cubes and data sources by applying semantic web technologies to define an OWL/RDF ontology for OLAP data sources and cubes. The proposed approach contains three different types of ontologies: global domain specific ontology to define specific concepts used in the

applications; a generic OLAP ontology to act as an upper ontology for OLAP cubes; and a domain specific OWL ontology of OLAP to model the specific concepts used in application. In this approach, the developers use these ontologies to model the cubes and develop data warehouse applications. So, any inconsistency occurred in the data sources in terms of schema and measure can be handled by using ontology mapping to transform the data to conform to the global domain specific ontology. In addition, the OLAP cube is generated through RDF queries, so the cube is created on the fly to satisfy a specific analysis.

Trujillo et al. [51] proposed a new approach by combining UML and XML to model multiple dimensional data cubes. DTD (Document Type Definition) and XML are used to facilitate the interchange of conceptual multiple dimensional models with a set of defined DTD elements to model the properties. As a result, DTD can be used to directly generate valid XML documents to represent multiple dimensional models at the conceptual level.

In Neimi et al. [52, 53] proposed a new architecture to allow the users to build their desired OLAP cube schema by using MDX (MultiDimensional Expressions) queries which can be analysed to identify the required data and collect them from the data sources (warehouses). It proposed an XML OLAP, in which schema is defined through XML language, to store all the collected data in XML. In other words, XML also is used in representing the actual data in the cube and OLAP cube (schema) relation. An OLAP server contains the cubes and data to be transformed and aggregated to a targeted cube which can be used by analysis tools.

MetaCube-X [54] is a federation architecture for integrating distributed data warehouses. It provides a neutral language, which is based on a XML instance of the MetaCube concept, to facilitate interoperability among different warehousing systems. It supports a protocol to access and search the required data among web warehouses. The architecture is to construct a unified XML documented based multidimensional data models by combining of two types of MetaCube-X: a local MetaCube-X located at a local data warehouse and containing schema for each web warehouse; a global MetaCube-X integrating local MetaCube-X by providing logic to alleviate their differences in terms of dimensions and measures and driving the local web warehousing to conform to the global schema.

Xie et al. [55] realised rapid changes in business environment, they have developed a tool based on Semantic Web technologies to make data warehouses business-friendly. They proposed an extended OWL language to represent the business metadata that includes conceptual enterprise data model and

multidimensional model, so the business semantics can be modelled and used to automatically generate a customised data mart with the required data and an OLAP cube metadata. The business users can employ the terms defined in conceptual enterprise data model to build multidimensional models and these terms can be mapped to data warehouse scheme for the deployment engine to generate a data mart.

The above research has exploited semantic web technologies to alleviate the possible schema inconsistency that can be occurred in integrating distributed data sources. Some used ontology or XML to model OLAP cube in order to take advantage of power of reasoning mechanism supported in the semantic web. However, the issue of reusing these data cubes has been not addressed.

8. A proposed Ontological OLAP

The proposed system is an intelligent information system that allow the DEHEMS users to make queries about the energy consumption and other related information. The system can analyse the requests and collect the relevant data to generate the response with the appropriate information. For example, the users may want to compare this year's total energy consumption with the last year's one, to report their house's energy consumption profile and their neighbour's or compare their house's energy consumption with the average household in this region. In order to answer the above questions, the proposed framework is a semantic OLAP which consists of semantic web technologies and intelligent mechanisms to alleviate schema inconsistency, to improve the cube capability on semantic interpretation and to reuse the existing cubes to build new composite cubes in order to meet new requests.

We assume the users pose different requests on the system for retrieving information, but these requests have relationships. In other words, some requests repeatedly appear in the queries directly or indirectly. For example, household X may want to compare the energy consumption in 2007 with 2006. So, the cubes for 2006 and 2007 energy consumption profiles have been generated when the request was issued in 2007. The cube for 2007 can be reused when the user wants to compare 2008 with 2007. So there is no need to re-generate 2007 energy consumption profiles for household X, if the cube is stored in the repository and is accessible when needed. Another simple example is that we can have OLAP cubes for monthly energy consumption profiles of House X, so the OLAP cube for the quarterly energy consumption profiles of House X can be derived from these monthly ones. The quarterly OLAP

cube can also be reused to generate the annual energy consumption profiles of House X. If each household in a region has similar OLAP cubes for energy consumption profiles, the OLAP cube for average energy consumption profiles of the regional household can be composed through these existing cubes. In this case the term region Y, which is an instance of region object in the ontology, needs to be explicitly defined.

Ontology in the semantic OLAP can be in general divided into two types: Domain ontology, and OLAP Cube Ontology. The domain ontology, for example, includes domestic appliance ontology which contains taxonomy of domestic appliances and their associated attributes, energy ontology which includes energy measurement and property, environment ontology which defines position coordinate, distance, height and others, and action ontology which has a number of controls actions for an actuator that it may take. The schemas of the data collected from different sensors which are stored in the data servers (data warehouses) are mapped to the terms defined in these ontologies.

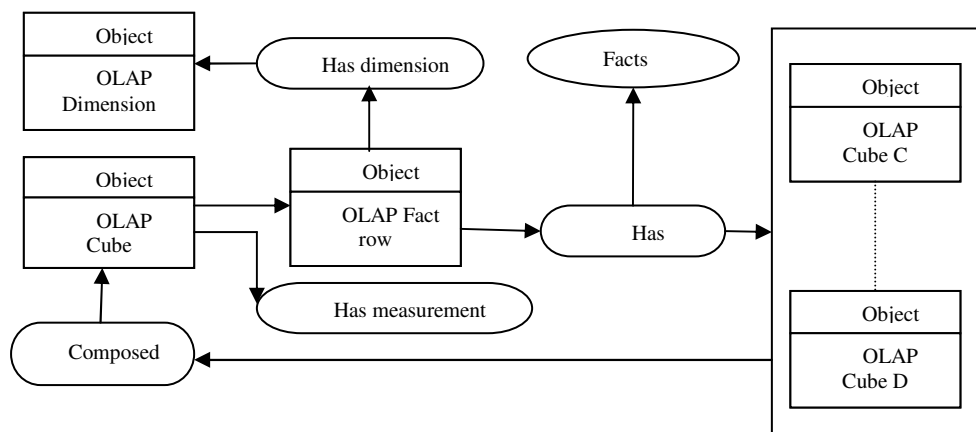


Figure 3. Composite OLAP

The above diagram (see Figure 3) shows the structure of a composite cube which is derived from the composition of existing cubes. A cube can include multidimensional attributes and contain a number of other cubes. A fact row in the cube can be a raw data or a pointer to another cube. The measure is derived from the fact rows for users to analyse or make decisions. These generic objects along with other essential objects (e.g. restriction, property) and their relationships are defined in the ontology. The “composed by” process in the diagram can include a set of procedure steps or heuristic rules to ensure that appropriate ontology will be used to alleviate the schema inconsistency among the cubes when integrating them.

One cube, for example, is to measure the average energy consumption in the area and the other one is to analyse which appliances consume most energy over a period of time in a household. These two cubes can be composed to form a new multidimensional cube for further analysis such as which appliances consume most energy in the area. Each fact row has an attribute to accommodate a reference to a cube such as monthly energy consumption. The term “area” needs to be defined explicitly in the ontology, so the household address can be clearly classified. This may require extra procedures before it can be achieved, apart from normal schema mapping, if some important data is missing. For example, an area covers certain streets which are stated in household address, but this information could have missed in some records. In this case, the post code can be used to derive the street name in order to fill the missing data. Sensors and receivers may have power cut or temporary disconnection, so some data could be missing. Heuristic rules can reason over historical data to make up for the missing data.

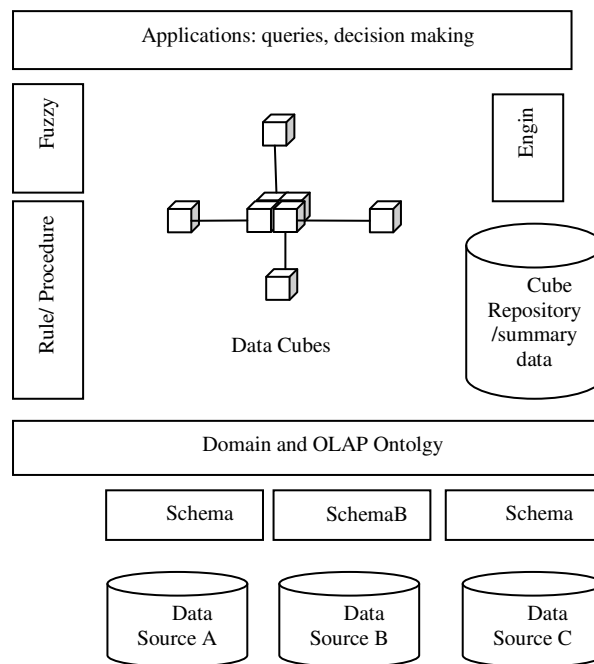


Figure 4. Ontological OLAP Framework

Figure 4 shows the ontological OLAP framework which includes rules, procedural functions and ontology to support schema mapping and OALP cube composition. The fuzzy mechanism in the framework is to support the users to employ the fuzzy terms in the queries. So, the cube attributes (schema) with queries and measure are stored in the cube repository for possible use. The creation of a new cube can either be derived from data sources or compose the existing cubes.

Composing a cube often needs steps to link different cubes together, as they may have attribute differences, missing data, or semantic and functional gaps. Rules, procedural functions, and ontologies can be used to model the necessary steps and inference to bridge the gaps and to conduct attributes mapping and transformation. The ontology can be also used to models different data schema provided by data sources to ensure that the name and semantics of these schemas be consistent in the data cube.

In Figure 5, the tools chosen for implementing the framework is illustrated. The ontology modelled in Protégé can be stored in two knowledge representation languages namely RDF and OWL [56]. The domain ontology and OLAP ontology are modelled via the protégé ontological development environment and stored as OWL for reasoning. However, Protégé by default does not provide facilities to support rules and fuzzy expressions. The Jess [57] tool is a rule engine for the Java platform, which supports production rule knowledge representation and provides an inference engine to reason over data and knowledge from different sources. JessTab [58] is introduced to bridge the gap between Jess and Protégé. So, each OLAP can be instantiated as an instance to capture application data and to produce the required measures. The heuristic rules and procedural functions in the Jess are able to access the instances and perform reasoning over them to reach at a conclusion or take appropriate actions. The fuzzy logic is realized by FuzzyJ[59] toolkit which provides a capability for modelling fuzzy concepts and reasoning in a Java setting. It supports Jess by providing a set of components in the library.

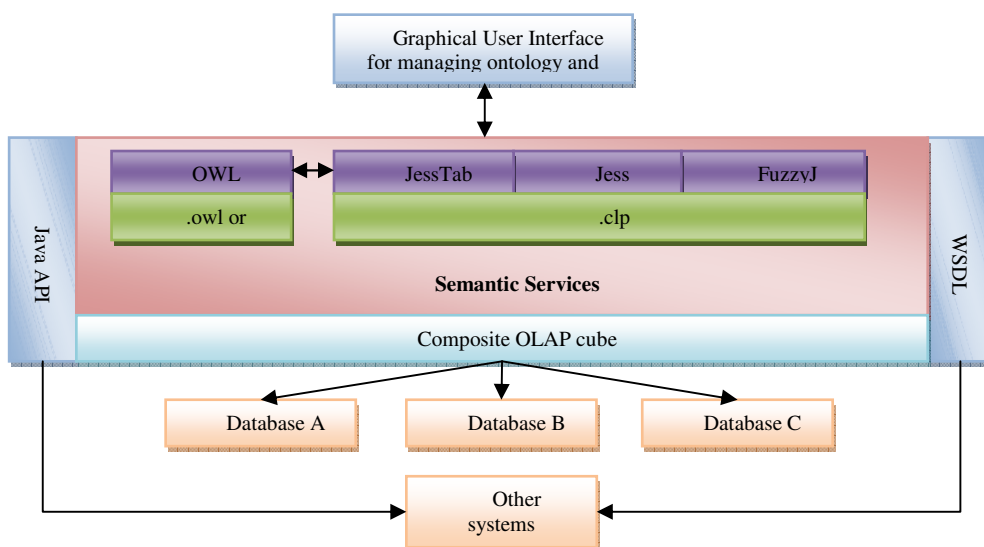


Figure 5. Implementation of Ontological OLAP Framework

The following OWL code shows that each cube contains three main elements “hasDimension”, “hasFactRow”, and “hasMeasure” to model information for decision making or -for further analysis. Each element contains multiple items such as Dimension, FactRow, and Measure defined as classes. Each FactRow has a number of FactVaule objects, and the FactValue contains a slot Sensor_Readings to accommodate data generated by a -sensor. So, this generic framework can be instantiated to model specific cases by populating them with attributes and the data.

```

<owl:Class rdf:ID="Cube">
  <rdfs:subClassOf rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
</owl:Class>
<owl:Class rdf:ID="Factrow">
  <rdfs:subClassOf rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
</owl:Class>
<owl:Class rdf:ID="Dimension">
  <rdfs:subClassOf rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
</owl:Class>
<owl:FunctionalProperty rdf:ID="hasMeasure">
  <rdfs:domain rdf:resource="#Cube"/>
  <rdf:type
</owl:FunctionalProperty>
<owl:FunctionalProperty rdf:ID="hasDimension">
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
  <rdfs:domain rdf:resource="#Cube"/>
</owl:FunctionalProperty>
<owl:FunctionalProperty rdf:ID="hasFactRow">
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
</owl:FunctionalProperty>
<owl:FunctionalProperty rdf:ID="hasCube">
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
</owl:FunctionalProperty>
<owl:FunctionalProperty rdf:ID="FactValue">
  <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
  <rdfs:domain rdf:resource="#Sensor_Reading"/>
</owl:FunctionalProperty>

```

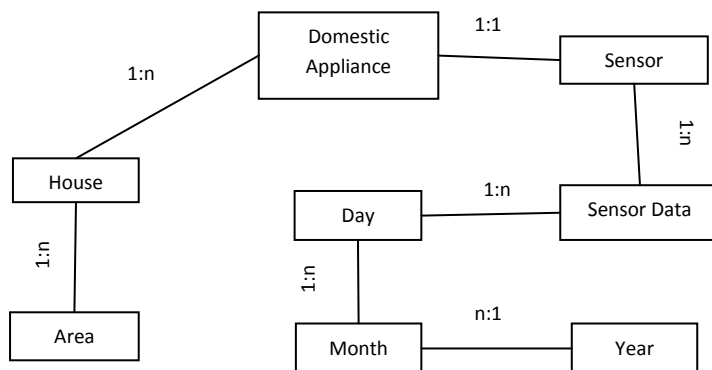


Figure 5 A Relationship Diagram for Sensor Application

Figure 5 shows a relational data diagram for a sensor application domain. Each house has a number of appliances and each appliance has a sensor to monitor the electricity usage. The sensor sends its readings to the server every 6 seconds. The data are collected and stored in the data sources, so the user may wish to use these data to analyse daily, monthly, and yearly energy usage of a certain appliance or certain type. The data, for example, can be used to measure top ten appliances in terms of energy usage. In order to support these possible queries or request, we utilise rule and functions supported by Jess and JessTab to instantiate the cube and to analyse the information.

The following shows part of code creating an instance of cube. Its superclass is OLAP and it creates instances of the slots to store the dimensions objects and others.

```
(make-instance of :STANDARD-CLASS
  (:NAME Cube_1)
  (:DIRECT-SUPERCLASSES OLAP)
  (:DIRECT-TEMPLATE-SLOTS
    (make-instance of :STANDARD-SLOT
      (:NAME Sensors)
      (:SLOT-VALUE-TYPE Dimension))
    (make-instance of :STANDARD-SLOT
      (:NAME Time)
      (:SLOT-VALUE-TYPE Integer)).....)
```

The raw data can be retrieved from data sources via SQL and JDBC and stored in the Jess. Before the data retrieval, the data schema needs to be mapped to the template defined in the Jess. Mapclass function supported by JessTab is the mechanism to link the Jess templates to the classes in OWL. The rules and functions in Jess can carry out necessary transformation between schema and template, if there is a need. The following shows a template in Jess to accommodate the data generated from a sensor every six seconds. The rules summarized them into daily average. These data are fed into the OLAP cube for further measurements such as yearly average, monthly maximum and minimum usage.

The following template is to accommodate sensor data in Jess. The command “slot-set” allows the data to be uploaded to the ontology.

```
(deftemplate Sensor
  (multislot Sensor_Reading)
  (multislot Time)
  (slot sensorID)
  (slot Location)
  (slot-set Sensor1 Sensor_Reading 21,33,44,55,33, 32)
```

The data in the ontology can be downloaded to Jess for reasoning.

```
(defclass FactValue (is-a :THING)
  (slot ID (type integer))
```

```
(slot InFactrow (type FactRow))
(slot Value (type any)))
```

```
(make-instance F1 of FactValue (ID ?x1) (FactRow ?x2) (Value ?x3) )
```

Similarly, the new cube can be composed via different FactRow objects (from different cubes) to produce a composite cube. A measurement for decision making can be generated by rules. The following shows the rule to calculate the average energy consumption for a household.

```
(mapclass :Cube)

(defrule measurement "calculate measurement"
?s<- (object (is-a Sensor)
      (ID ?a)
      (Location ?HouseID))
?c<- (object (is-a FactValue)
      (ID ?a) (FactRow ?factrow) (Value ?factvalue))
?a<- (object (is-a Household)
      (houseAddress ?HouseID))
=>
(slot-set ?c :Measure (Avg ?factvalue ?factrow ?HouseID)))
```

The proposed framework demonstrated its ability to utilise the ontological representation, rules and procedural functions along with their supporting reasoning mechanism to enhance the reusability of existing data cubes. Some researchers have proposed ontology to enrich data cubes, but the reusability has not been emphasised. In this sense, our framework does not only resolve schema consistency, but also consider using the existing data cubes as building blocks. It could short the development time and improve information sharing.

9. Conclusion

In this paper, we have proposed an ontological OLAP framework to integrate distributed energy sensor data. The data cubes in the framework are annotated with semantics and supporting mechanisms to alleviate the issues of the schema inconsistency in data sources. The mechanisms such as rules and procedural functions to support the mapping between data schema and to carry out data transformation have been introduced to the framework. When a cube is generated, it can be stored in a repository for future use. As a result, these existing OLAP cubes can be reused and composed to form a new cube.

The proposed ontological OLAP framework provides mechanisms to support necessary steps required for composition, as the cubes may have attribute differences and missing data. The developers can take advantages of these existing cubes and the supporting functions to produce composite multidimensional cubes for analysis and decision making. A prototype of the proposed framework

based on a number of existing tools such as protégé, Jess, and FuzzyJ has been developed to test the feasibility of the proposed approach. Further development on the system by introducing more domain knowledge and ontologies to knowledge repository is needed.

Acknowledgement

The research leading to these results has received funding from the European Community's Seventh Framework Programme FP7/2007-2013 under grant agreement n° No224609.

References.

- [1] S. K. Goudos, V. Peristeras, and K. A. Tarabanis, "Semantic Web Application for Public Administration using OWL for Public Domain Data Knowledge Representation" WSEAS Transactions on Information Science & Applications. Vol. 4(4), pp. 725-730, 2007.
- [2] R. Kimball, and M. Ross, "The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling" John Wiley, 2002.
- [3] R.A Moeller, "Distributed Data Warehousing Using Web Technology" AMACOM American Management Association, 2001.
- [4] A. Bâra, I. Lungu, M. Velicanu, V. Diaconița, I. Botha, "Improving Query Performance in Virtual Data Warehouses" WSEAS Transactions on Information Science & Applications, Vol. 5(5), pp. 632-641, 2008.
- [5] Y. Ji, "Towards Framework for the Virtual Data Warehouse"
<http://www.cs.ualberta.ca/~zaiane/postscript/thesis/YuanJi2001.pdf> , 2001
- [6] B. Lin, Y. Hong, Z. Lee, "Data Warehouse Performance" Encyclopedia of Data Warehousing and Mining, pp. 580-585, 2009.
- [7] T.B. Pedersen, "Warehousing The World: A Vision for Data Warehouse Research" New Trends in Data Warehousing and Data Analysis, pp. 1-18, 2009.
- [8] K. Schewe, J. Zhao, "Balancing Redundancy and Query Costs in Distributed Data Warehouses" Second Asia-Pacific Conference on Conceptual Modeling, pp. 97-106, 2005.
- [9] L. Pang, Best Practices in Data Warehousing. In: Wang, J. (ed) Encyclopedia of Data Warehousing and Mining. pp. 146-152, 2009.
- [10] R. Chiky, G. Hebrail, "Summarizing Distributed Data Streams for Storage in Data Warehouses". LNCS, Springer-Verlag Berlin Heidelberg, vol. 5182, pp. 65-74, 2008.
- [11] W.H. Inmon, "Building the Data Warehouse. John Wiley & Sons , 2005.
- [12] S. Kanchi, "Selecting the Right Architectures for Successful Data Warehouses"
<http://www.tcs.com/SiteCollectionDocuments/WhitePapers/SelectingtheRightArchitecturesforSuccessfulDataWarehouses.pdf> accessed on 2009.
- [13] A. Simitsis, D. Theodoratos, "Data Warehouse Back-End Tools" Encyclopedia of Data Warehousing and Mining, pp. 572-579, 2009.
- [14] J. Zhao, "Designing Distributed Data Warehouses and OLAP Systems. Information Systems Technology and its Applications", pp. 254-263, 2005.
- [15] K. Jouini, G. Jomier, "Design and Analysis of Index Structures in MultiVersion Data Warehouses" New Trends in Data Warehousing and Data Analysis, pp. 165-186, 2009.
- [16] C. Combi, B. Oliboni, G. Pozzi, "Modeling and Querying Temporal Semistructured Data Warehouses" New Trends in Data Warehousing and Data Analysis. pp. 299-324, 2009.
- [17] Data Warehouse Process, <http://www.gantthead.com/content/processes/9076.cfm>
- [18] L. Bellatreche, K. Boukhalfa, P. Richard "Data Partitioning in Data Warehouses: Hardness Study, Heuristics and ORACLE Validation" LNCS, Springer-Verlag Berlin Heidelberg, vol. 5182, pp. 87-96, 2008.
- [19] L. Bellatreche, M. Mohania "Physical Data Warehousing Design" Encyclopedia of Data Warehousing and Mining, pp. 1546-1551, 2009.
- [20] R. Almeida, J. Vieira, M. Vieira, H. Madeira, J. Bernardino "Efficient Data Distribution for DWS" LNCS, vol. 5182, pp. 75-86, 2008.
- [21] T. Ozsu, P. Valduriez, "Principles of Distributed Database Systems" Prentice-Hall, 1999.

- [22] L. Kerschberg, "Knowledge Management in Heterogeneous Data Warehouse Environments" Third International Conference on Data Warehousing and Knowledge Discovery, LNCS, vol. 2114, pp. 1-10, 2001.
- [23] A. Salguero, F. Araque, C. Delgado, "Ontology Based Framework for Data Integration. WSEAS Transactions on Information Science & Applications, Vol. 5(6), pp. 953-962, 2008.
- [24] S. Iqbal, J. Bunn, H.B. Newman, "Distributed Heterogeneous Relational Data Warehouse in a Grid Environment" Computing in High Energy and Nuclear Physics, 2003.
- [25] E. Malinowski, E. Zimányi, "Conceptual Modeling for Data Warehouse and OLAP Applications" Encyclopedia of Data Warehousing and Mining, pp. 293-381, 2009.
- [26] M. Schneider, "A General Model for Data Warehouses" Encyclopedia of Data Warehousing and Mining. pp. 913-919, 2009.
- [27] W. Litwin, S. Sahri, T. Schwarz, "Architecture and Interface of Scalable Distributed Database System SD-SQL Server" Databases & Applications, pp. 182-187, 2006.
- [28] M. Stonebraker, D.J. Abadi, A. Batkin, X. Chen, M. Cherniack, M. Ferreira, E. Lau, A. Lin, S. Madden, E. O'Neil, P. O'Neil, A. Rasin, N. Tran, S. Zdonik, "C-Store: A Column-oriented DBMS" VLDB Conference, Trondheim, Norway, <http://db.csail.mit.edu/projects/cstore/vldb.pdf>, 2005.
- [29] J.D. Abadi, "Column-Stores For Wide and Sparse Data" Innovative Data Systems Research (CIDR), Asilomar, California, USA, 2007.
- [30] High Performance Scalable Storage, <http://kosmosfs.sourceforge.net/>
- [31] Hypertable: An Open Source, High Performance, Scalable Database, <http://hypertable.org/>
- [32] A. Khetrapal, V. Ganesh, "HBase and Hypertable for large scale distributed storage systems" <http://www.ankurkhetrapal.com/downloads/HypertableHBaseEval2.pdf>, 2006.
- [33] F. Chang, J. Dean, S. Ghemawat, W.C. Hsieh, D. Wallach, M. Burrows, T. Chandra, A. Fikes, R. Gruber, "Bigtable: A Distributed Storage System for Structured Data" 7th Symposium on Operating Systems Design and Implementation (OSDI '06), pp. 205-218, 2006.
- [34] S. Rizzi, A. Abello, J. Lechtenborger, and J. Trujillo, "Research in Data Warehouse Modelling and Design: Dead or Alive?" ACM conference on DOLAP, pp3-10, 2006.
- [35] E.F. Codd, S.B. Codd and C.T. Salley, "Providing OLAP to User-Analysts: An IT Mandate", 1993.
- [36] T. Berners-Lee, J. Hendler, O. Lassila, "The Semantic Web. Scientific American" 284(5), pp. 35-43, 2001.
- [37] R. Studer Benjamins, D. Fensel, "Knowledge Engineering: Principles and Methods", IEEE Transactions on Data and Knowledge Engineering, vol. 25(1-2): pp. 161-199, 1998.
- [38] T. Gruber "A translation approach to portable ontology specifications" Knowledge Acquisition pp. 199-199., 1993.
- [39] K. Graham, C. Jeremy, "Resource Description Framework (RDF): Concepts and Abstract Syntax", <http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/>, 2004.
- [40] Brickley Dan, Guha RV. RDF Vocabulary Description Language 1.0: RDF Schema. <http://www.w3.org/TR/2004/REC-rdf-schema-20040210/>; 2004.
- [41] D. Mike, G. Schreiber, "OWL Web Ontology Language Reference", <http://www.w3.org/TR/2004/REC-owl-ref-20040210/>, 2004.
- [42] J. Manuel Pe´rez, R. Berlanga, M. Jose´ Aramburu, and T. Pedersen, "Integrating Data Warehouses with Web Data: A Survey" IEEE Transactions on Knowledge and Data Engineering, Vol. 20, No 7. 2008.
- [43] U. Dayal, M. Castellanos, A. Simitsis, K. Wilkinson, "Data integration flows for business intelligence", Proceedings of the 12th International Conference on Extending Database Technology: Advances in Database Technology, pp. 1-11, 2009.
- [44] D. Skouta, A. Simitsis, "Ontology-Based Conceptual Design of ETL Processes for Both Structured and Semi-Structured Data" Int'l Journal on Semantic Web & Information Systems, 3(4), pp.1-24,
- [46] F. Tseng and C. Chen, "Integrating Heterogeneous Data Warehouses Using XML Technologies," J. Information Science, vol. 31, no. 3, pp. 209-229, 2005.
- [47] R.M. Bruckner, T.M. Ling, O. Mangisengi, and A.M. Tjoa, "A Framework for a Multidimensional OLAP Model Using Topic Maps" Proc. Second Int'l Conf. Web Information Systems Eng. (WISE'01), pp. 109-118, 2001.
- [48] ISO/IEC 13250: Information technology –SGML applications– Topic Maps, International Organization for Standard, Genf, Schweiz, Dec, 1999.
- [49] O. Mangisengi, J. Huber, C. Hawel, and W. Essmayr, "A Framework for Supporting Interoperability of Data Warehouse Islands Using XML" Third Int'l Conf. Data Warehousing and Knowledge Discovery (DaWaK '01), pp. 328-338, 2001.

- [50] T. Niemi, S. Toivonen, M. Niinimäki, J. Nummenmaa “Ontologies with Semantic Web/Grid in Data Integration for OLAP” Int’l Journal on Semantic Web & Information Systems, PP. 25-49, 2007.
- [51] J. Trujillo, S. Luján-Mora, and I. Song, “Applying UML and XML for Designing and Interchanging Information for Data Warehouses and OLAP Applications ” J. Database Management, vol. 14, no. 1, pp. 41-72, 2004.
- [52] T. Niemi, M. Niinimäki, J. Nummenmaa, and P. Thanisch, “Constructing an OLAP Cube from Distributed XML Data ” Fifth ACM Int’l Workshop Data Warehousing and OLAP (DOLAP ’02), pp. 22-37, 2002.
- [53] T. Niemi, M. Niinimäki, J. Nummenmaa, and P. Thanisch, “Applying Grid Technologies to XML Based OLAP Cube Construction”, Fifth Int’l Workshop Design and Management of Data Warehouses (DMDW ’03), pp. 4.1-4.13, 2003.
- [54] T.B. Nguyen, A.M. Tjoa, and O. Mangisengi, “MetaCube-X: An XML Metadata Foundation of Interoperability Search among Web Data Warehouses ” Third Int’l Workshop Design and Management of Data Warehouses (DMDW ’01), pp. 81-88, 2001.
- [55] G. Xie, Y. Yang, S. Liu, Z. Qiu, Y. Pan, and X. Zhou, “EIAW: Towards a Business-friendly Data Warehouse Using Semantic Web Technologies”, ISWC/ASWC 2007, LNCS Volume 4825/2008, pp. 857–870, 2007.
- [55] OWL Web Ontology Language Overview <http://www.w3.org/TR/owl-features/>, 2004.
- [57] Jess, <http://www.jessrules.com/>
- [58] JessTab, <http://www.ida.liu.se/~her/JessTab/>
- [59] FuzzyJ, http://www.iit.nrc.ca/IR_public/fuzzy/fuzzyJToolkit2.html
- [60] Kyoto Protocol, http://unfccc.int/kyoto_protocol/items/2830.php
- [61] Combating Climate Change: The EU leads the way, <http://ec.europa.eu/publications/booklets/move/70/en.pdf>
- [62] DEHEMS, <http://www.dehems.eu/>