A Framework for Spatial Analytics using Heterogeneous Data Sources

Cláudio de Souza Baptista, Tiago Eduardo da Silva,
Brunna de Sousa Pereira Amorim

Federal University of Campina Grande
Campina Grande, Paraíba, Brazil
baptista@computacao.ufcg.edu.br,
tiagoes@copin.ufcg.edu.br,
brunnasousa@copin.ufcg.edu.br

Hugo Feitosa de Figueirêdo
Federal Institute of Education, Science and Technology of Paraíba
Esperança, Paraíba, Brazil
hugo.figueiredo@ifpb.edu.br

José Mário Pereira Dantas
Court of Accounts of the State of Acre – TCE-AC
Rio Branco, Acre, Brazil
mario.dantas@tce.ac.gov.br

Abstract—**Several solutions for the integration of Business Intelligence (BI) and Geographic Information Systems (GIS) have been proposed in recent years aiming at improving the decision making process. The term Spatial Analytics has been coined to tools that perform analysis on spatial and conventional data organized according to the multidimensional approach. Nevertheless, no consensus has been reached regarding the best way to accomplish such integration, making it difficult to perform analysis on spatial cubes from heterogeneous multidimensional data sources. In this article, we investigated the state of the art on Spatial Analytics, and propose a framework that enables spatial analytics on cubes from heterogeneous multidimensional data servers. The proposed framework provides a visual query language for the spatial analysis. To validate the proposed framework, a practical example is conducted and applied to accountability processes of the Court of Accounts of the State of Acre, in Brazil. To perform such case study, the framework was extended to access cubes from Microsoft SQL Server Analysis Services (SSAS) and GeoMondrian.**

Keywords- Business Intelligence; GIS; Analytics; GeoBI; SOLAP.

I. Introduction

With the increasing volume of data coming from a large variety of sources, there has been a considerable increase in investments on technologies capable of extracting information from these data and, consequently, help managers in the decision making process. *Business Intelligence* (BI) tools provide a historical, updated and predictive view of business operations of a company, enabling the identification of patterns, the availability of new functionalities and products, and improving the relationship with costumers. *On-line Analytical Processing* (OLAP) is one of the most used BI tools. An OLAP tool enables rapid exploration and analysis of data stored in multiple aggregation levels, according to the multidimensional approach. In this context, most companies are adopting BI tools in order to become more competitive in the marketplace.

In addition to this, most companies heavily deal with the spatial dimension in their datasets. Hence, it is important to investigate how to explore that dimension in order to improve the decision making process. However, traditional BI technologies do not take advantage of spatial data. On the other hand, Geographical Information Systems (GIS) were designed to work on georeferenced data using the Online Transaction Processing (OLTP) approach, and thereby preventing an efficient and deep data analysis.

More recently, corporations have demanded the integration of GIS and OLAP technologies arising a new category of tools known as *Spatial Online Analytical Processing* (SOLAP), or simply Spatial Analytics.

This article proposes a new framework that enables the analysis of spatial cubes coming from multiple and heterogeneous multidimensional data sources. This article is an extended version of the Geoprocessing 2016 Conference paper by Silva et al. [1]. In this extended version, we included a discussion on the design of the spatial cubes, provided more details on the framework extension points and addressed more spatial cube operators.

The integration of GIS and BI technologies may happen through three distinct approaches: prioritizing the resources of GIS (GIS-dominant), overlapping visual and graphical resources of OLAP tools (OLAP-dominant), and the full integration approach (SOLAP) that aggregates the functionalities of GIS with graphs, tables and maps [2].

The need for integrating GIS and OLAP technologies have boosted new SOLAP academic solutions. Recently, several research works have been published on SOLAP addressing several approaches. Aissi et al. address the use of recommendation on SOLAP tool [3]. Li et al. propose a map-reduce architecture for SOLAP [4]. Leonardi et al. discuss SOLAP for trajectory data [5]. Diallo et al. focus on mobile GeoBI [6]. Nonetheless, there is no consensus on how to properly integrate GIS and OLAP technologies. The proposed solutions differ in several aspects, mainly the data model. Without a consensus on the data model, it is very hard to provide spatial cubes on the Web.

Spatial cubes are characterized as data cube that contain spatial data in the fact table or in dimensions, or in both. Vector data is used with at least three data types: points, linestrings and polygons, or collection of those. Spatial operators include topological, metric, set, directional and network. Topological operators include: inside, meets, crosses, cover, overlaps, contains, disjoint and equals. Metric operators include: area, perimeter, length, distance, far, near, and buffer. Set operators are Union, Intersection, and Difference. Direcional operators are: left, right, above, below, north, south, east, west, northeast, northwest, southest, and southwest. Network operators include connected, next, previous, shortest_path. Other spatial operators include: minimum bounding rectangle, centroid, convex_hull [7].

Still regarding GIS and BI systems integration, some works investigated and proposed ways to provide data on the Web to ensure interoperability. Dubé et al. [8] present a XML format to provide and exchange SOLAP cubes via Web Service. In 2011, the Open Geospatial Consortium (OGC) [9] published a white paper that analyzes how the OGC standards (i.e. WMS, WFS, WPS, etc.) could be extended in order to intensify the use of geospatial information and the interoperability of GeoBI applications [10].

Some core features of Spatial Analytics solutions were observed:

- Queries through visual specification language using spatial operators;
- An integrated view of multidimensional and spatial data; using maps, tables and charts; and
- Extensibility to provide access to heterogeneous multidimensional data sources (cube servers).

We enumerate the desirable key requirements of Spatial Analytics solutions:

I. to enable the creation of queries through a visual query language: visual languages improve usability specially concerning Spatial Analytics where queries are quite complex to express;
II. to provide an integrated view of both multi-dimensional and spatial data: enables to analyze data on maps, graphics and tables, simultaneously;
III. to support spatial operators to enable deep analysis: spatial data demands spatial operators such as topological, metric, directional; which enhances user experience in such dimensional data;
IV. to provide access to heterogeneous multidimensional data sources (cube servers): the access to heterogeneous cube servers promote interoperability and data integration without needing to migrate data among these cubes;
V. to enable geocoding data to provide spatial analysis in non-spatial OLAP sources: currently, some cube servers are not spatially aware. Hence, geocoding enables to use such cube servers in a SOLAP solution.
VI. to use open technologies to reduce costs: enabling the use of free source and open technologies may increase ROI (Return of Investment) in SOLAP projects;
VII. to be extensible so that new features can be added: the conception of a white box framework enables code reuse so that programmers may incorporate further features to fulfill new requirements; and
VIII. to enable data visualization through maps, tables and graphs: incorporate into the dashboards maps, tables and graphs to enhance usability in the decision making process.

The lack of consensus for GIS - OLAP integration and standards for the provision of spatial and multidimensional data hinders the use of different data sources at the same time. Hence, the extraction of useful information to improve the decision making process at corporations is impaired.

Our proposed framework can be classified as an Enterprise Application Framework, as it is concerned with the OLAP domain [11]. According to Sommerville, a framework is a software that can be extended to create a more specific application [12]. The main contributions of our research is the proposal of a framework for Spatial Analytics that contains interfaces and abstract classes that can be implemented and extended to support new data sources, making easy the integration of heterogeneous data cubes. Hence, we offer a reusable software to interoperate spatial datawarehouses from heterogeneous data sources.

Furthermore, in order to validate the proposed framework we present a case study on the accountability analysis of the TCE-AC (Court of Accounts of the State of Acre – Brazil). In the case study, we extended our framework for Spatial Analytics to access cubes stored in two different sources: *Microsoft SQL Server Analysis Services* (SSAS) and *GeoMondrian*.

The rest of this article is organized as follows. Section II discusses related work on Spatial Analytics. Section III addresses the architecture of the proposed framework. Section IV presents a case study involving the Court of Accounts of the State of Acre – Brazil. Finally, Section V highlights the conclusions and further work to be undertaken.

II. Related Work

SOLAP has been a very active research area for a long time. Surveys on SOLAP can be found in [13][14][15]. Salehi et al. propose a formal model for spatial datacubes [16]. Aguila et al. address a conceptual model for SOLAP [17]. Baltzer focuses on spatial multidimensional querying [18]. Glorio and Trujillo highlight the optimization of spatial queries [19]. Ziouel et al. propose an approach for cartographic generalization of SOLAP applications [20].

Several SOLAP tools have been developed over the last years in many contexts. Rivest et al. propose a generic SOLAP tool, called JMap Spatial OLAP, that provides an interactive data exploration through charts and maps. The proposed tool is based on Relational OLAP (ROLAP) architecture and supports the three types of spatial dimensions: geometric, non-geometric and mixed. A disadvantage of the proposed tool is that it does not allow the use of spatial operators (metrical or topological).

Bimonte et al. developed the GeWOlap SOLAP tool, highlighting the synchronization of different forms of data visualization [21]. Their architecture comprises three layers: data, SOLAP server and client layers (user interface). Nonetheless, the authors' proposed tool does not enable the use of spatial operators (metric or topologic) and uses proprietary technology. In [22] the authors discuss the use of the GeWOlap tool in the domain of agriculture.

Escribano et al. proposed a tool called Piet, integrating GIS and OLAP technologies and executing the precomputation of the map layers [23]. The Piet architecture also comprises three layers: data, SOLAP server and client layers. The query processor can process four types of queries: geometric, geometric aggregation, OLAP and GIS-OLAP. A language named GISOLAP-QL is proposed. This query language has two parts: the first part retrieves spatial data and the second one retrieves multidimensional data. The Piet tool does not have an interactive interface that hides query language details. Piet consists of two applications: a Web application for OLAP queries and a desktop for spatial queries. Therefore, it does not have an integrated view of multidimensional and spatial data.

Another challenge faced in SOLAP solutions is the issue of aggregation performance when queries involve considerable amounts of spatial data. Li et al. combine SOLAP approach with the Map-Reduce model for processing large amounts of data in parallel [24]. Aissi et al. propose a multidimensional query recommendation system aiming to help users to retrieve relevant information through SOLAP, improving the data exploitation process [25].

Scotch and Parmanto [26] propose the SOVAT tool (Spatial OLAP Visualization and Analysis Tool) to help public health researchers and professionals in the decision making process. SOVAT main features include performing statistical and spatial analysis, providing a detailed data exploration, and viewing data through charts and maps. However, the tool does not provide metric and topologic spatial operators.

The Golapware tool was developed and used to process GeoMDQL, a geographic-multidimensional query language to spatial analysis [27]. The SOLAP server is an extension of the Mondrian OLAP server. The server contains an engine responsible for SOLAP processing called GOLAPE (Geographical Online Analytical Processing Engine) and it supports spatial queries using the GeoMDQL language. The Golapware tool does not offer interface components for visual interaction of queries with the user, resulting in a more complex data analysis.

Stole and Hanrahan [28] present a data analysis interface that extends the Pivot Table interface. Polaris is an interface used in exploratory analysis of large multidimensional relational databases, and has a set of graphic components to specify relational queries visually and to view data (visual specification language). Polaris interface enables visual analysis through a visual specification language called VizQL. However, this language does not provide support for spatial operators and can manipulate only points (latitude and longitude). Although it supports map overlay, where these layers may come from another data source, it does not allow overlapping of layers originated from geometrical fields. On the other hand, our proposed solution extends the VizQL language to visual spatial analysis. This extension aims to overcome spatial analysis drawbacks found in that language. The tool was built to relational databases, so it neither supports hierarchies and levels, concepts related to multidimensional cubes, nor allows the use of spatial operators.

Lamas et al. [29] developed an integration of the MapServer with the Saiku Analytics analysis tool (OLAP), making possible spatial dimension data visualization using maps. With the Web tool, thematic maps that represent spatial distribution of a particular cube measure can be created using different color tones. The color gradation represents different intervals of the selected measure in territorial units. The Saiku tool provides an interface to navigate and select the cube metadata (measure, dimension, etc.) building a thematic map. For this purpose, the tool offers two components (columns and rows) in which users may select the dimensions and measures to analyze, respectively. It is not possible to create spatial filters to build maps because the tool does not provide spatial operators.

Dubé et al. presented an XML file exchange SOLAP cubes through web services [8]. The proposed XML file does not depend on the OLAP/GIS tool and represents all the necessary data (facts and members) and metadata (scheme), besides supporting spatial dimensions. The advantage of exchanging data through Web Services is that the communication is not limited to traditional client-server platforms, but also supports ubiquitous mobile computing environments.

The aforementioned solutions differ in the data model and there is no standard for provision and analysis of spatial data. In this perspective, the Open Geospatial Consortium (OGC) published in 2012 a report (white paper) containing an evaluation of the ways that the OGC standards (e.g., WMS - Web Map Service, WFS, WPS - Web Processing Service, etc.) could be extended, in order to promote the use of geospatial information and the interoperability of GeoBI applications.

Table I presents a comparison among the related work concerning the eight desirable requirements presented in Section 1. Cells that contain an "X" mean that a given solution implements a given feature and those that contain a "–" mean that a given solution does not implement a particular feature.

This article presents a framework for Spatial Analytics, known as SOLAP_Frame that enables the connection to multiple and heterogeneous data sources. The framework was developed using open source technologies. Furthermore, the proposed framework presents an integrated visualization of multidimensional and spatial data, allowing for the creation of queries by means of a visual specification language with support to spatial operators and data visualization through maps, tables and charts. Finally, SOLAP_Frame also enables the geocodification of the data and is extensible, providing support for addition of new functionalities, such as new operators or data visualization methods. To the best of our knowledge, this is the first work to propose a framework for Spatial Analytics that is able to interoperate with new heterogeneous data sources.

TABLE I. RELATED WORK COMPARISON

Solutions	**Requirements**							
	I	***II***	***III***	***IV***	***V***	***VI***	***VII***	***VIII***
JMAP	X	X	-	-	-	X	X	X
GeoWOLAP	X	X	-	-	-	-	-	-
SOVAT	X	X	-	-		-	-	X
Piet	-	-	-	-	-	-	-	-
Golapware	-	X	X			X	-	X
Polaris	X	X	-	-		-	-	X
Saiku	X	X	-	-	-	X	-	X

All works addressed in this article have in common an architecture in which the client is strongly connected to the SOLAP server, which prevents the analysis of data from other SOLAP servers.

Piet and Golapware solutions do not have interactive interfaces in which the user can specify a query. This obliges the user to master a certain query language. The Piet solution has two different interfaces: one for multidimensional analysis and another for spatial analysis. In other words, they do not provide a data integrated view. SOVAT and Piet solutions lack of a Web interface, so the remote access via Internet is not possible. GeoWOLAP, Piet and Mapwarehouse solutions do not present one or more of the following data visualization forms: chats, graphics and maps.

Except Golapware and Mapwarehouse solutions, the other tools do not allow using spatial operators to slice and dice data cubes, minimizing the query's power expression. The analyzed solutions support only one type of data integration: either integrated or federated. All solutions, except Piet and Golapware, have a language to visually specify the query. However, only the Polaris solution defines a specification language for data visualization and query definition. GeoWOLAP, SOVAT, Piet and Polaris solutions do not use open source technologies, and GeoWOLAP, SOVAT, Piet and Golapware are not extensible.

Several attempts have been expended toward building SOLAP tools. However, in the studied solutions, it is not possible to reuse the client layer due to the lack of standards on communication between SOLAP server and client. Hence, based on the key features analyzed in Table I, it is clear that no solution addressed in the state-of-the-art can be considered satisfactory.

III. THE SOLAP_FRAME FRAMEWORK

This section presents the SOLAP_Frame architecture. In the next subsections, we describe the architecture and the extension points of our proposed SOLAP framework.

A. The Spatial Cube Data Model

In our framework the data structure is a cube composed of measures and dimensions. The dimensions have hierarchies composed of levels and are responsible for the categorization of the SOLAP cube. Levels, on the other hand, are composed of members. The class diagram that represents the cube can be seen in Figure 1.

In each cube dimension, the members are grouped into levels, that are arranged into hierarchies. A hierarchy defines different levels of detail. In a hierarchy, the levels follow an order that represents the hierarchy level depth. Members relate with themselves, that is, a member of a certain level is son of a superior level member.

Levels and members have properties, which have a type and a value. The property type characterizes the property domain a number, text, date or geometry. The value is an element of the property domain. The level lists properties common to members, so it is possible to know the properties of the members in advance.

Measures are the quantitative values used to measure characteristics of the analyzed phenomenon. In the cube, they are organized as a special dimension called measures dimension, whose members are measure names organized in a special hierarchy called measures hierarchy. This hierarchy has a special level called measures level. In the proposed model, the attributes *isMeasureDimension*, *isMeasureHierarchy* and *isMeasureLevel* of the *Dimension*, *Hierarchy* and *Level* classes, respectively, are responsible for identifying the measures dimension, hierarchy and level.

Each cube component should have, at least, two attributes: unique name and name. The unique name unequivocally identifies the cube's component, and the name is presented to the user. Each member of the measures dimension has a set of measure values associated to it. There is a measure value for each combination of different dimensions. In the proposed framework, the measure values are not associated to the cube itself, but to the result of a multidimensional query.

Cube components are considered spatial if a level member, a hierarchy or a dimension contain a spatial property, a spatial level or a spatial hierarchy, respectively. Lastly, a cube is spatial if it has, at least, one spatial measure or spatial dimension.

B. Extension of the Vizql Query Language

Based on Polaris, an interface proposed by Stolte and Hanrahan [30], the goal of the interface proposed in this work is to facilitate the process of data analysis. The framework's interface proposed here has a visual specification language that is an extension of the VizQL [31] formalism. We extended VizQL specifications to retrieve and visualize spatial data in multidimensional cubes.

Through the tabular algebra, VizQL partitions the data according to the visual specification defined for table settings, that is, VizQL tab expressions. The available VizQL tabs are: Columns, Rows, Filters and Tableaus.

Tab Layers was added to VizQL to support spatial analysis (Figure 2). Polaris tab Layers is used only for data segmentation and in Polaris commercial version, it was renamed to Tableau.

A fundamental concept in Geographic Information Systems (GIS) is overlay. A GIS system has data organized in layers that can be overlapped, facilitating the data analysis through visual data correlation. In a cube, data is organized into hierarchy levels, so a spatial level can be used as a map layer. However, there is no tab in the VizQL language for the overlay spatial operation.

A new tabular algebra operator called *SpatialConcatenation* was added to VizQL. For this new operator, the operands must be spatial fields and their names are assigned to the set of names resulted from the application of this operator, where the fields can be either qualitative or quantitative.

In the VizQL language, each table cell has a panel that displays data relative to that cell. To allow data visualization in maps with overlaying, a new type of panel was added: the spatial panel. Thus, panels can display spatial or conventional data.

The spatial panel is a result of panel overlapping of the tabular algebra resulting tables applied to the tab “Layers”. This means that, to show a query result in the map, the tables are overlapped and merged into one, unlike the tab “Tabular” that shows one table at a time. Another difference is that the conventional panel aggregates measure values while the

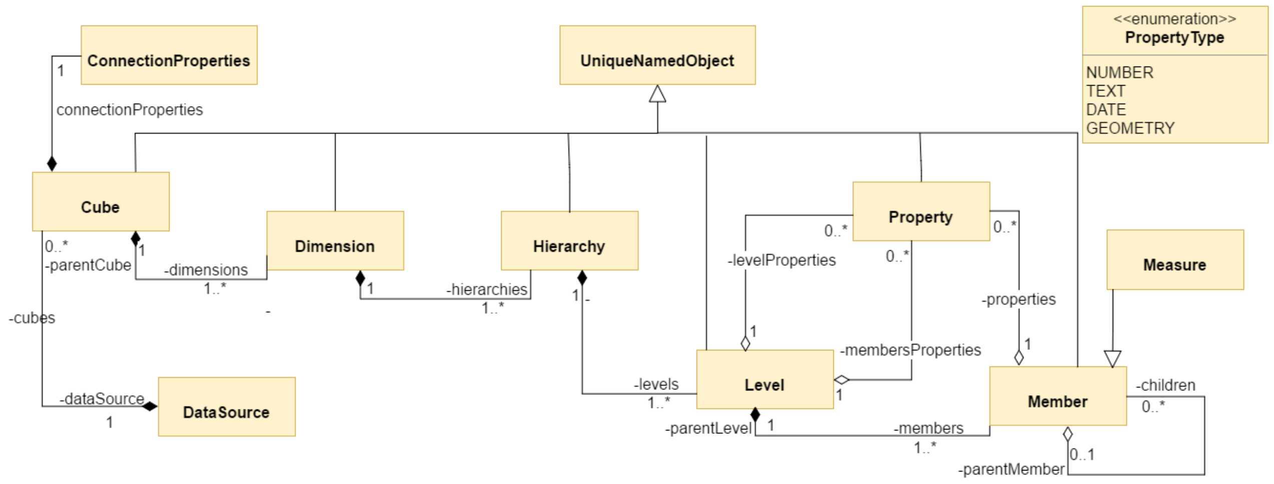
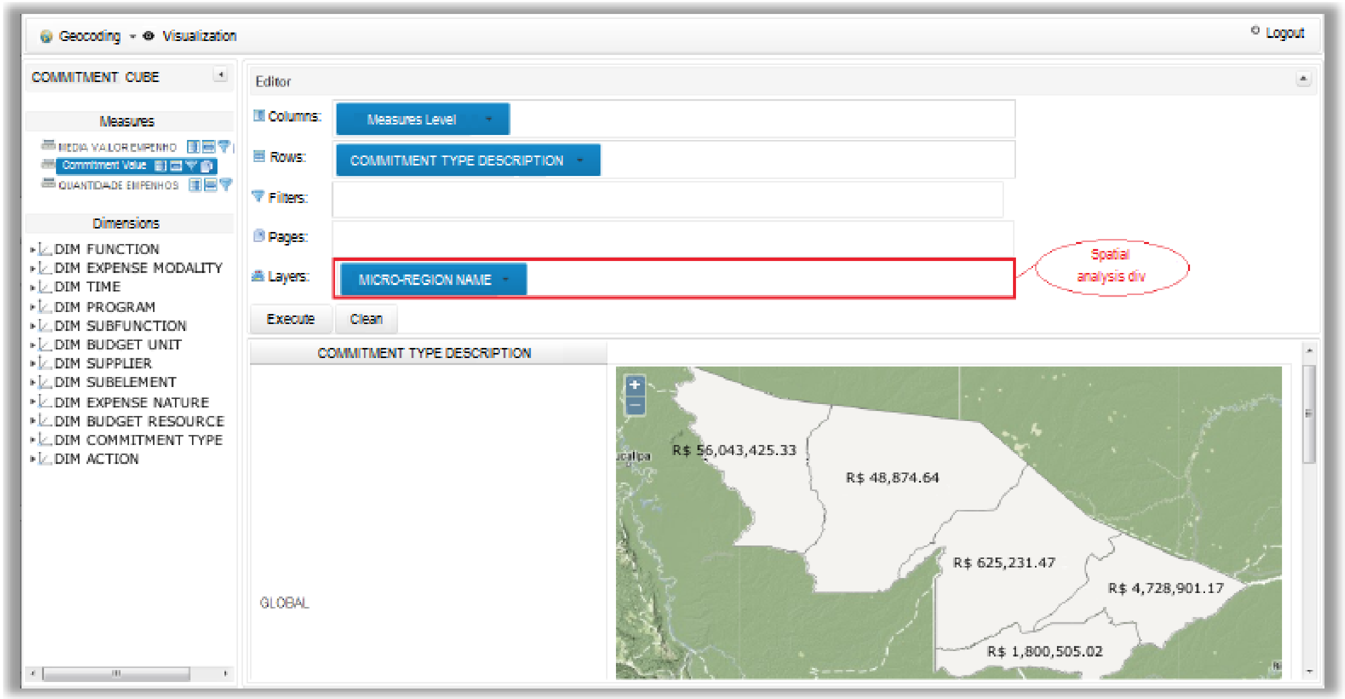


Figure 1. Spatial cube data model



Columns: Measures Level
Rows: COMMITMENT TYPE DESCRIPTION
Filters:
Pages:
Layers: MICRO-REGION NAME

Figure 2. Layers div for spatial analysis

spatial panel aggregates measures by geometry, also known as feature.

The features are grouped by fields. Each set created by this grouping is a layer, making layer overlap possible, for example, in a map. When adding levels to the tab “Layers”, the table cell will be composed of layers. In addition to the tab “Layers”, spatial constraints are specified in VizQL, according to OGC specification. The OGC (Open Geospatial Consortium) developed standards to model, access, store and share geographic data. The topological operators defined by OGC, based on DE-9IM model (Dimension Extended Nine-Intersection Model) include Touches, Within, Crosses, Overlaps, Disjoint, Contains, Equals, Intersects and Relate. These operators are available for the geographic filter specification. The OGC also specifies Buffer, ConvexHull, Envelop, Boundary, Centroid and PointOnSurface operators. They can be used in geometric filters.

C. Architecture

The framework architecture comprises three layers: client, application and data layers, as shown in Figure 3.

The client layer comprises a set of graphical Web interfaces in which the user can connect to a multidimensional spatial cube, geocode members, pose queries by means of a visual specification and visualize the result set.

The data layer comprises the multidimensional data sources to be analyzed and the spatial data repository. The framework is capable of accessing several multidimensional servers (cube servers), employing different technologies and manufacturers. The framework also supports the geocoding of cube members, enabling the spatial analysis of non spatial OLAP cubes. The application data repository is stored in the *PostgreSQL* DBMS, with the PostGIS spatial extension. The spatial data resulting from the geocoding of members of the cube are stored in this repository, characterizing a Data Warehouse federated approach. Any spatial DBMS can be used for this purpose, simply extending the proposed solution through its extension points.

The application layer is responsible for the implementation of the whole application logic. This layer has six modules: visual query specification, data visualization, map manager, spatial data repository access and the SOLAP engine. We highlight the SOLAP engine as the main module of this layer, providing communication between the application and the multidimensional servers (OLAP or SOLAP) attached to the data layer.

The visual query specification module controls the query execution and result set visualization, turning the interactions between users and the graphical interface into objects that compose the query visual specification. After receiving the result set from a visual query, the visual specification module forwards this result set with its markup to the data visualization module so that the data can be transformed and presented in the specified format. Depending on the markup type, data may have to be transformed, for example, grouped to compose the map layers or graphic axes. After transformed, the data set is forwarded to the most appropriate component of the interface for visualization (e.g., tables, maps, charts, text and caption).

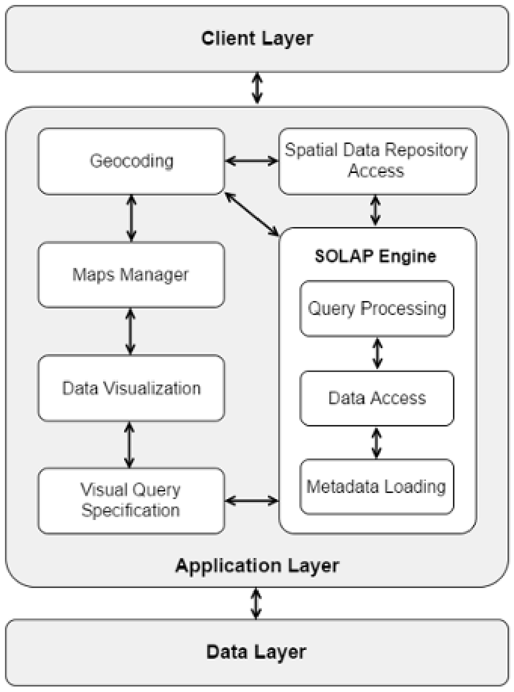


Figure 3. The SOLAP_Frame architecture

The map management module is responsible for displaying data on maps. For such, this module receives, from the data visualization module, a set of spatial and numerical data, query results, and the markup. The repository access module, in turn, was implemented to retrieve the metadata and the data from the spatial tables stored in the spatial data repository. The metadata and the data in the spatial tables are used by the geocoding module, which is accessed using the Java Database Connectivity (JDBC) driver for the PostgreSQL database management system (DBMS) with the PostGIS spatial extension.

The SOLAP engine is composed of three sub-modules: data access, metadata loading and query processing. This engine is responsible for: implementing the connection to a given multidimensional data source; loading of metadata from cubes to be analyzed; translating the visual specification into the destination query language; submitting the translated query; and receiving the result set.

The implementation of the SOLAP engine depends on the manufacturer of the SOLAP server to be accessed.

The data access module enables the connection to the multidimensional data source and the choice of the cube to be analyzed. This module is also in charge of executing queries in the language of the accessed technology and of returning the results of these queries. To accomplish the data access module, it is necessary to inform the connection properties that match both source and cube properties. The source properties state where and how to connect to the multidimensional data source, while the cube properties address which cube belonging to a given source should be accessed. The data access module knows how to handle heterogeneous sources.

The metadata loading and the query processing modules of the SOLAP engine interact with the data access module, which is specialized, that is, its implementation depends on the adopted technology.

The metadata loading module is responsible for retrieving cube metadata. In order to connect to a multidimensional data source, the *ConnectionProperties* and *DataSource* objects must be informed. The metadata coming from this connection will be turned into Cube objects, which will be loaded into memory for posterior use by other modules of the proposed framework.

The query processing module is responsible for translating the visual queries into the target technology native language; executing them using the data access module; and returning the result set. In order to retrieve the data, besides the connection properties, a visual query is passed as parameter to the processing query module.

D. The SOLAP_Frame Extension Points

SOLAP_Frame contains extension points that enable to connect it to heterogeneous data sources. In this section we give details of the communication interfaces.

1) The SOLAP Engine Facade

The main extension point of the proposed framework is the implementation of the SOLAP engine facade. This facade is responsible for the communication between the proposed solution and the multidimensional data source. The message exchange between our framework and the SOLAP engine consists of requesting metadata and data from a cube. The facade standardizes this message exchange. To request metadata from a cube to the SOLAP engine, the facade contains the *loadCube* method that receives as parameters the connection properties of both the data source and cube and returns an object that represents the cube.

To request data from a cube to the SOLAP engine, the facade contains three methods: *processQuery*, *getLevelMembers* and *filterLevelMembers*. The *processQuery* method receives as parameter an object that models the query, which is part of the visual specification defined by the user. This visual query should be translated into the query language of the source technology; and then executed. The query result must be modeled in a return object called *VisualQueryResult*.

The *getLevelMembers* method receives as parameter an object that represents a hierarchical level. This level is used to retrieve the members of the cube. Finally, the *filterLevelMembers* method receives as parameter, besides the level, a filter that can be either conventional or spatial. This filter will be used to select the members to be retrieved.

Our framework will automatically identify the implementation of the front end by means of the Contexts and *Dependency Injection services* (CDI) present in the Java Enterprise Edition platform, and will register it. A name and a type must be associated to the SOLAP engine in order to be presented to the user. The type is used by the BI engine manager to ensure the mapping between types and implementations available.

2) The Connection properties interface

The communication process requires that the user provides the connection properties. This information will be used every time the SOLAP engine needs to communicate with a data source. Thus, another extension point in our framework is the implementation of this user interface.

The connection properties depend on the technology to be used. Hence, the parameters that must be informed vary according to the technology.

The front end for the SOLAP engine contains a method called *getLoaderPopup*, which returns an object called *LoaderPopup*, which, in turn, contains the necessary information for the exhibition of the component. This object is used by the interface, that lists all the available. The *LoaderPopup* object is composed of another object called *LoaderBean*, that needs to be implemented. The *LoaderBean* is the controller responsible for preparing the component for exhibition and for enabling access to the properties of connections created by the user.

3) The XMLA engine

In order to provide access to several heterogeneous multidimensional data sources, we also developed a SOLAP engine for servers that provide their data through the XMLA protocol. To enable the XMLA engine to access a specific technology, it is necessary to implement the abstract classes described in the following. In the implementation of the SOLAP engine for XMLA, we used the XMLA driver supplied by the Open Java API for OLAP (olap4j), which is also an open specification for the construction of OLAP applications based on the JDBC protocol. Once connected to the data source, the user chooses the cube that will be analyzed. After that, an alias is assigned to the cube. This alias will be used to identify the cube in the system.

After the selection of the cube, its metadata will be loaded. For such, it is necessary to convert the metadata from the native format into the target one. The metadata loading is carried out by the *Olap4jXMLACubeMetadataDAO* class, and the abstract class *AbstractOlap4jXMLACubeConverter* implements the basic methods necessary for the conversion of the cubes from the native format to the format used in the solution.

The methods of the *AbstractOlap4jXMLACubeConverter* abstract class are spatial related and depends on the technology used by the XMLA server. This is due to the fact that XMLA does not specify a standard format for the transportation of spatial data. Furthermore, the Multidimensional Expressions (MDX) query language, used by the XMLA server, does not specify spatial functions. Figure 4 presents a class diagram for the XMLA engine.

To load the data, the *AbstractOlap4jXMLAQueryDAO* class supplies the basic functionalities necessary for the correct operation of the framework. However, it is necessary to implement the method responsible for translating MDX filters into the language for the chosen technology. The abstract method is necessary due to the fact that our framework deals with spatial filters. Since these spatial filters are not standardized for MDX, they vary according to the technology used.

IV. Case Study Applied To The Court Of Accounts Of The State Of Acre – Brazil

In order to evaluate the SOLAP_Frame, we ran a case study on public accountability of the Court of Accounts of the State of Acre – Brazil (TCE-AC). We used a real dataset from that Court. The aim of this case study was to help in the decision making process related to the definition of more efficient management strategies to achieve an effective control of public spending. More specifically, citizens may inspect public budget and expenditure from a macro-level (e.g. the whole State) to a micro-level (e.g. a specific city).

To run this case study, the framework was extended to connect to two multidimensional data sources: *SQL Server Analysis Services* and *GeoMondrian*, of which the first one provides access to conventional data, and second one provides access to spatial data. Three cubes are available for the analysis: Commitment, Liquidation and Payment. The Commitment cube uses the opensource *GeoMondrian* server, while the other ones utilize the Microsoft SSAS.

The fact tables to be analyzed are represented by the measures: Commitment values, Liquidation values and Payment values. Besides the measures modeled in the *Spatial Data Warehouse* (SDW), the cubes have two additional measures: number and mean of the values.

The Spatial DW used in the cube implementation is modeled by the conceptual schemata presented in Figures 5, 6 and 7. These DW projects are tailored for the TCE-AC domain.

The facts have the following dimensions in common: Action, Supplier, Function, Expense Nature, Program, Expense Subelement, Subfunction, Time, Budget Resource Source Type and Budget Unit. The Commitment and Payment cubes have the Expense Modality dimension in common. The Liquidation and Payment cubes have the Summary Commitment dimension in common and the Commitment Type dimension features only measures of the Commitment cube, while the dimension Bank Account features only measures of the Payment cube.

Some dimensions have members organized by hierarchies, such as: Action, Supplier, Subelement, Bank Account, Time and Budget Unit. The members of these dimensions were organized in the following level of detail.

- Action Dimension: members organized in action type and action;
- Supplier Dimension: members organized in supplier type, cpf, cnpj and supplier name;
- Subelement Dimension: members organized in expense element and subelement;
- Bank Account Dimension: members organized in bank, agency, account type and account;
- Time Dimension: members organized by year, month and day;
- Budget Unit Dimension: members organized in state, meso-region, microregion, city, management unit and budget unit. These regions are defined by a Brazilian Government Agency called IBGE (www.ibge.gov.br).

The Budget Unit dimension is a spatial dimension because of its spatial attributes: State Name, Mesoregion Name, Micro-region Name, City Name; and its spatial hierarchy: State – Meso – Micro – City. The spatial portion of the DW that refers to the Commitment cube was migrated to PostgreSQL so the GeoMeondrian can access these data.

A. Query Examples

In order to assess the framework in the case study we analyzed some queries involving Spatial Analytics.

Query 1: “Display a map with the average of Commitment values detailed by state, mesoregion, micro-region and city.”

In order to solve this query, the Commitment cube was used. The Average Commitment Value measure was added to the tab “Columns”, while the hierarchy State – Mesoregion – Micro-region – City of the Dim Budget Unit dimension was added to the tab “Layers”. This hierarchy has four spatial levels: State, Mesoregion, Micro-region and City.

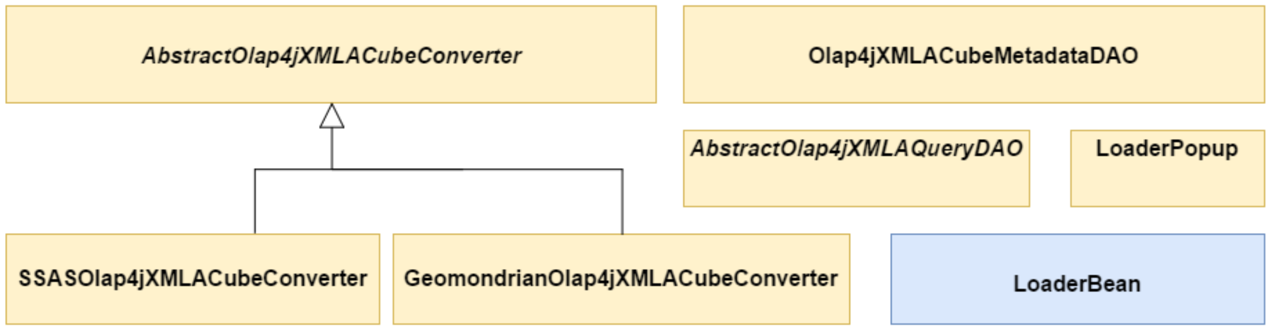

AbstractOlap4jXMLACubeConverter
Olap4jXMLACubeMetadataDAO
AbstractOlap4jXMLAQueryDAO
LoaderPopup
SSASOlap4jXMLACubeConverter
GeomondrianOlap4jXMLACubeConverter
LoaderBean

Figure 4. Class Diagram for the XMLA Engine.

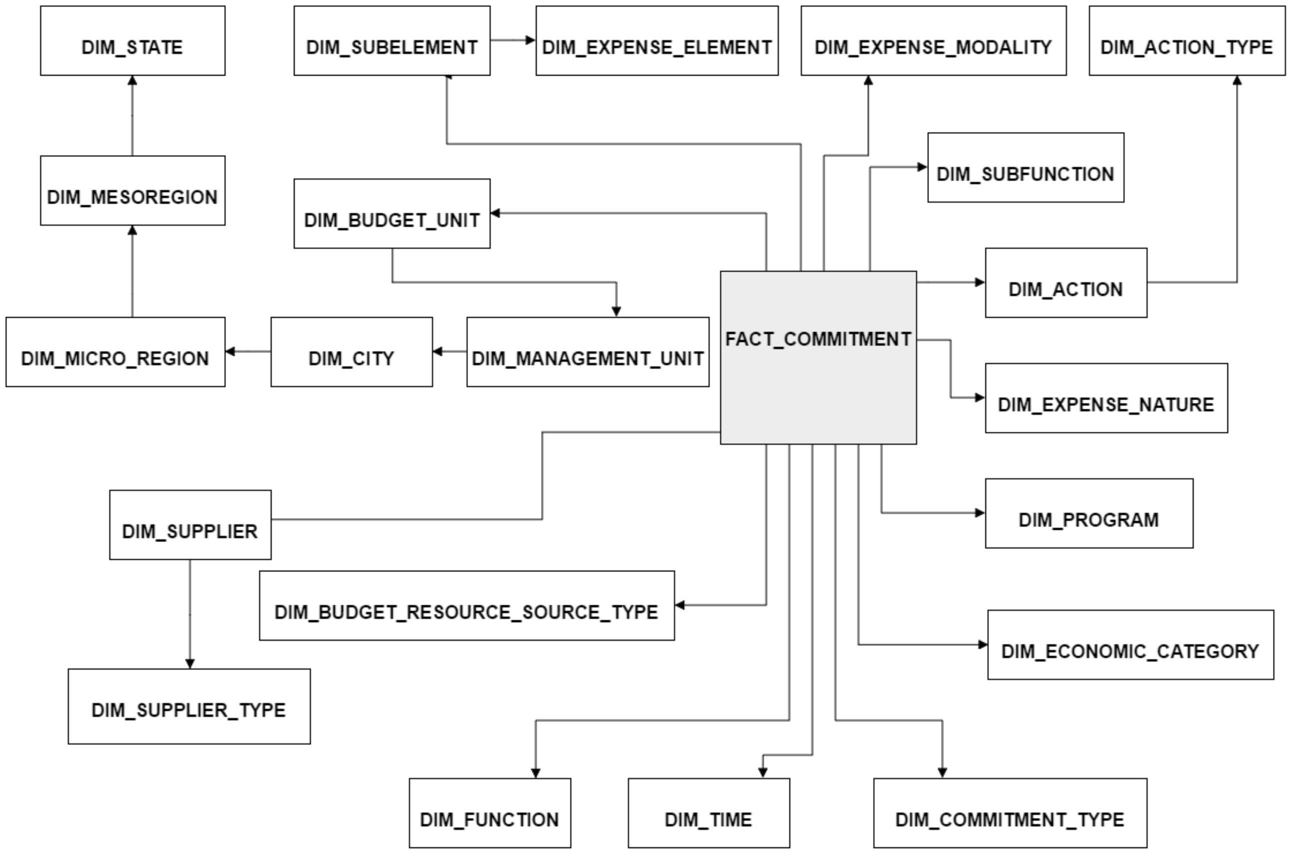


Figure 5. DW schema for the Commitment cube

Adding the hierarchy to the tab, its less level of detail is displayed with the navigation icon +, that is used to increase the level of detail (Figure 8). For data details, the + icon was used to State, Mesoregion (Figure 9) and Micro-region levels. Data visualization was specified by selecting the spatial panel type and the text type marking, where the selected measure was used as a geometry label. If there is no measure value associated with the geometry, the geometry will not be displayed on the map. The goal of this query is to exemplify the hierarchy navigation and the text type visualization in spatial panels (maps).

Query 2: "Display a thematic map with the sum of the Commitment values for each city."

Also, using the Commitment cube for this query, the Commitment Value measure was added to the tab "Columns" while the spatial hierarchy City Name was added to the tab "Layers". The marking type Caption was used for data visualization in the map. (Figure 10).

The caption created for the measure Commitment Value is of type value range, and the amplitude of the measure value is used to group geometries. It is also possible to create captions for the group type, where records are evenly divided in the ranges of values. The level City Name was used as a label to the geometries. This query exemplifies the display of the type Caption in maps.

Query 3: "What is the sum of the liquidated values in the neighbor cities of the city of Rio Branco, concerning the functions Administration, Agriculture and Legislative?"
This query demonstrates the use of the spatial filters available in the framework. In this example, the cube Liquidation was used; the Liquidation Value metric was added to the tab "Columns", the hierarchy Function Description was added to the tab "Rows", and the hierarchy District Name, to the tab "Layers".

The members of the Function Description level were filtered. A geographic filter was added to the cities, and the spatial operator Touches was used to filter neighbor cities of the city of Rio Branco. To visualize the data, we used a caption in spatial panels. Figure 11 presents the query result.

Query 4: "How much was spent with undergraduate education, elementary education and regular education in 2012, detailed by month, in the Rio Branco micro-region and Tarauacá city?"

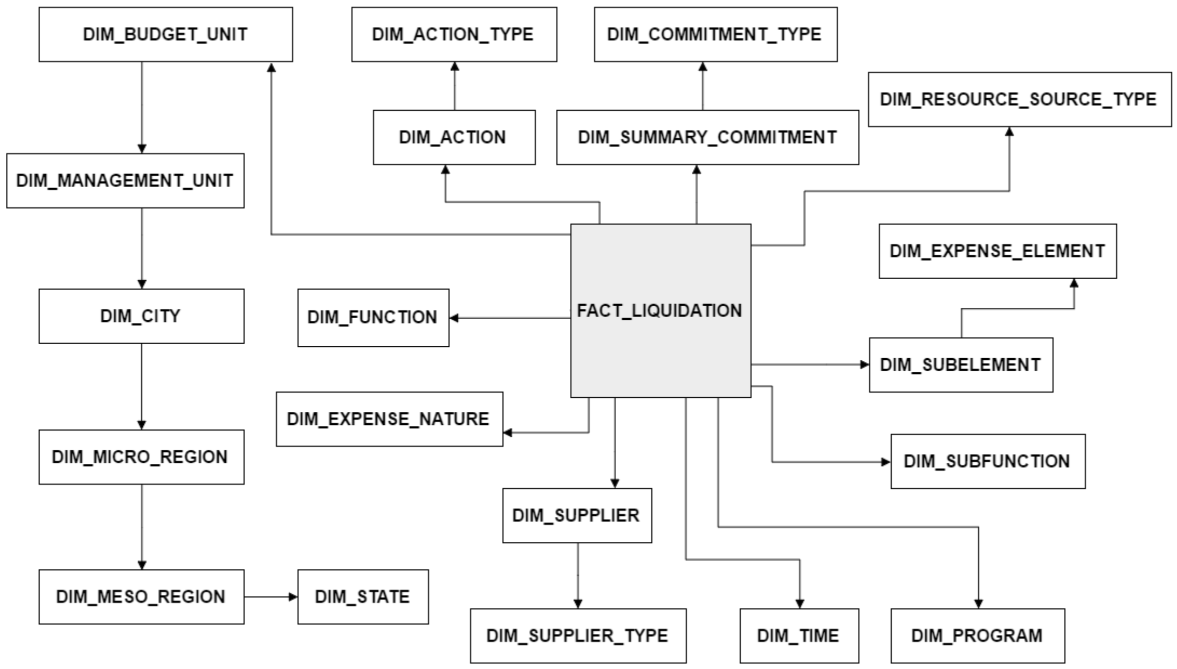


Figure 6. DW schema for the Liquidation cube

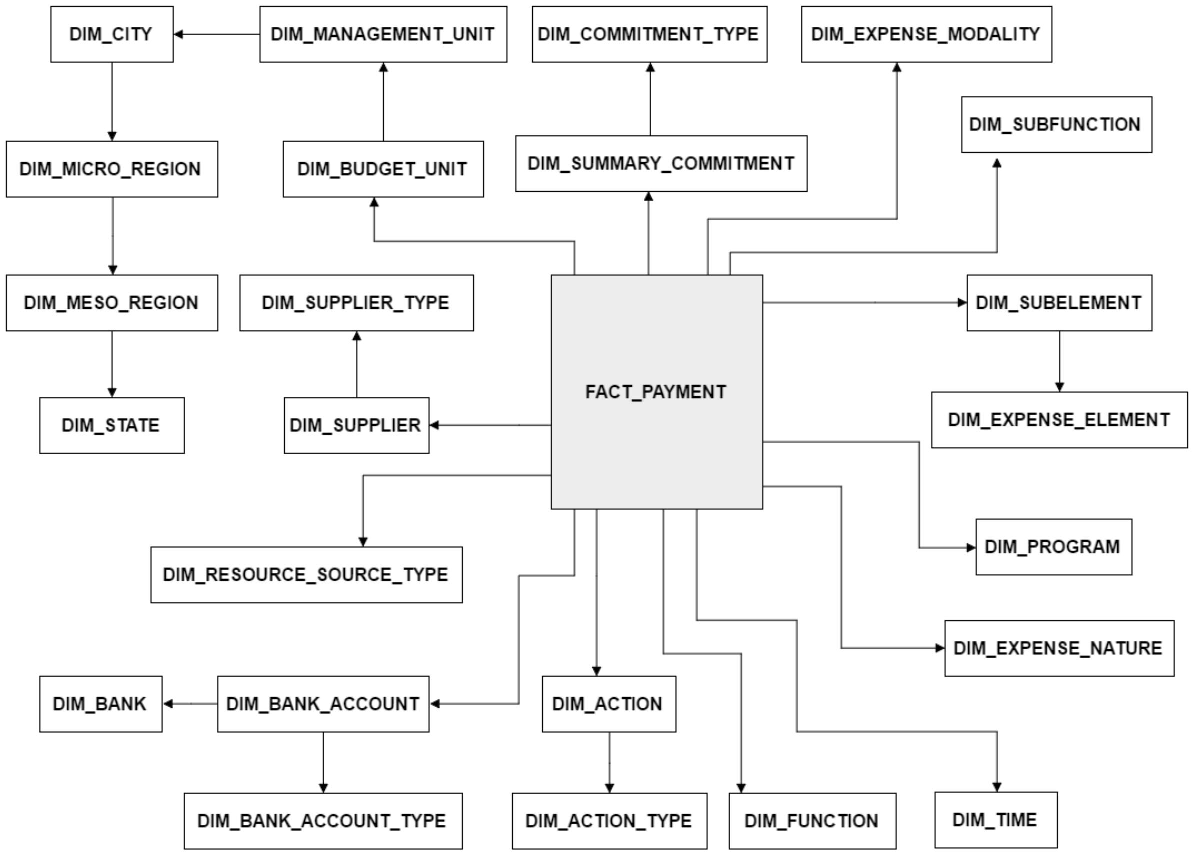
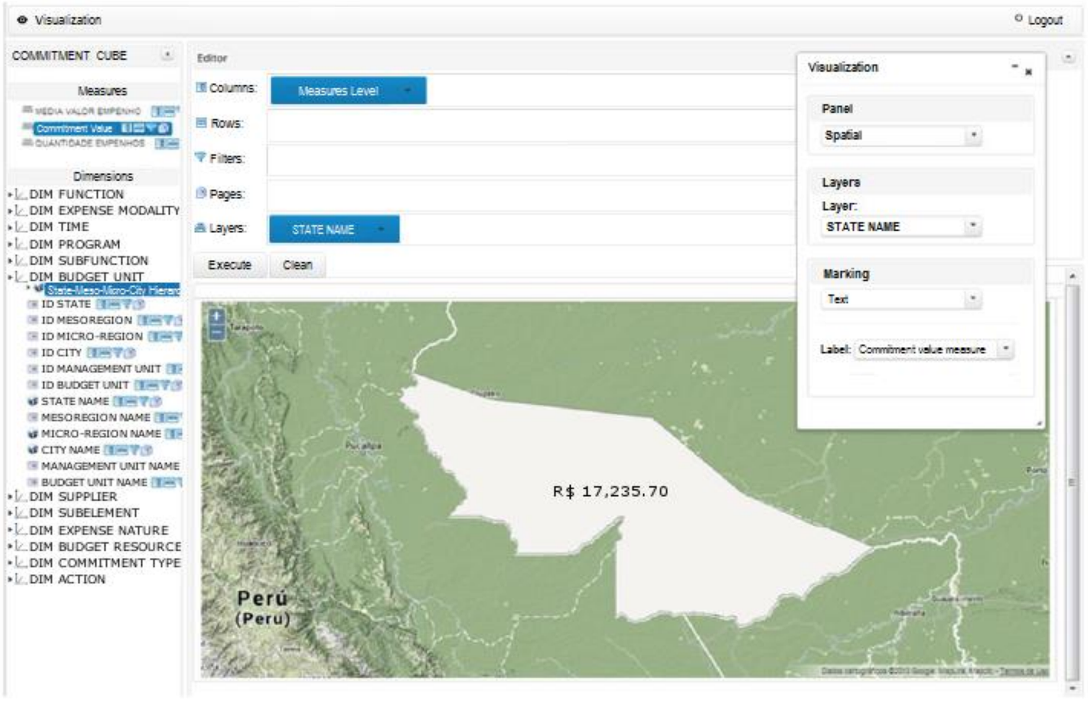


Figure 7. DW schema for the Payment cube

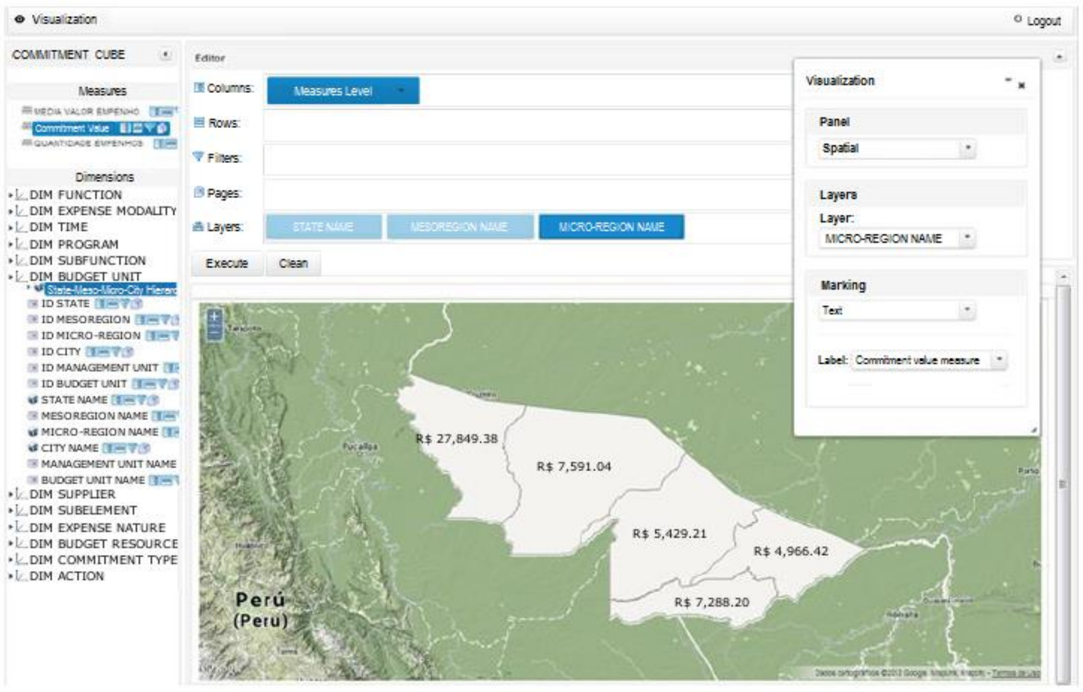
In order to present an example of pagination and overlap of layers, the Payment Value measure was added to the tab “Columns”, the Program Description hierarchy was added to the tab “Rows”, the Year – Month – Day hierarchy to the tab “Tableau” and the hierarchies City and Micro-region to the tab “Layers”. The member 2012 of the Year level was selected and, to detail the pages by months, a drill-down was made. The member Rio Branco of the Micro-region level and the member Tarauacá of the City level were selected. To visualize the data, we used a caption in spatial panels. Figure 12 presents the query result.



Columns: Measures Level
Rows:
Filters:
Pages:
Layers: STATE NAME

Panel: Spatial
Layers — Layer: STATE NAME
Marking: Text
Label: Commitment value measure

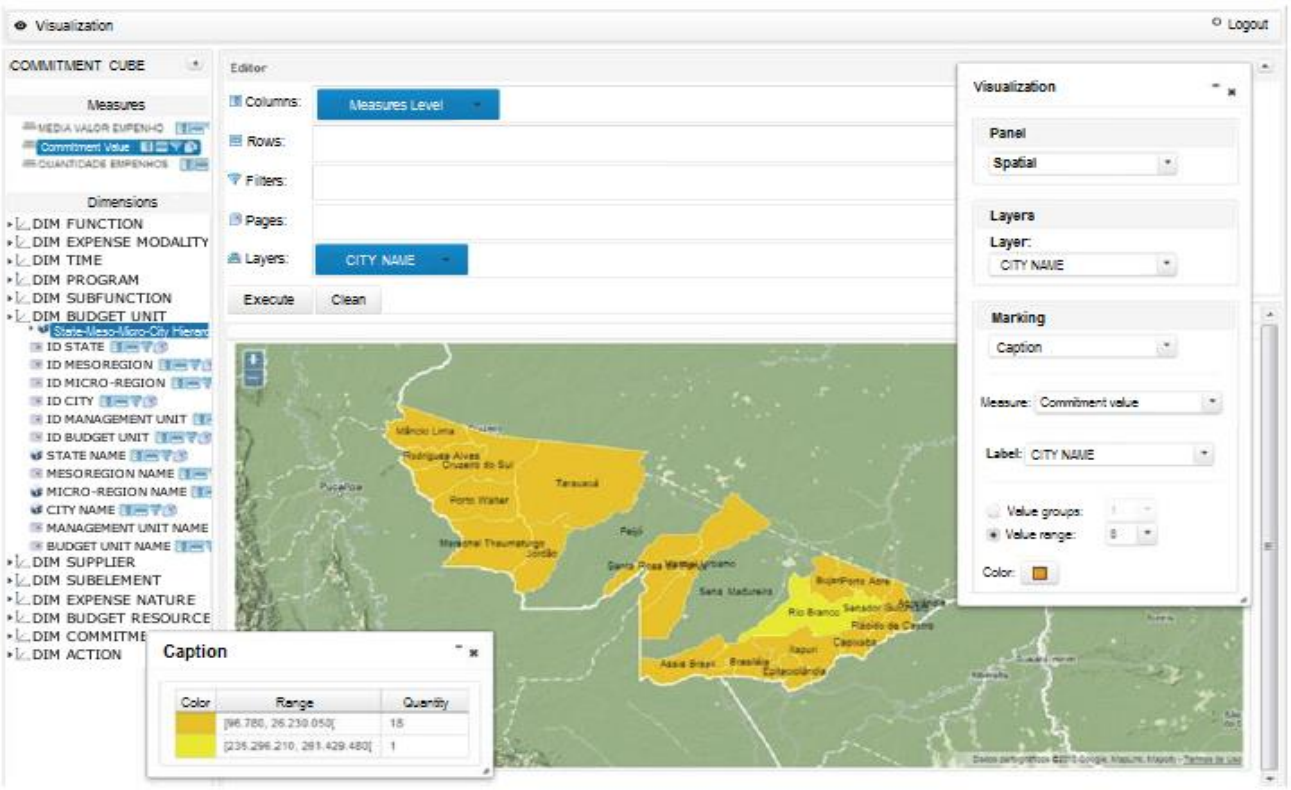
Figure 8. Query 1 result – State level.



Columns: Measures Level
Rows:
Filters:
Pages:
Layers: STATE NAME | MESOREGION NAME | MICRO-REGION NAME

Panel: Spatial
Layers — Layer: MICRO-REGION NAME
Marking: Text
Label: Commitment value measure

Figure 9. Query 1 Result – Mesoregion Level



Columns: Measures Level

Rows:

Filters:

Pages:

Layers: CITY NAME

Visualization

Panel: Spatial

Layers — Layer: CITY NAME

Marking: Caption

Measure: Commitment value

Label: CITY NAME

Value groups: 1

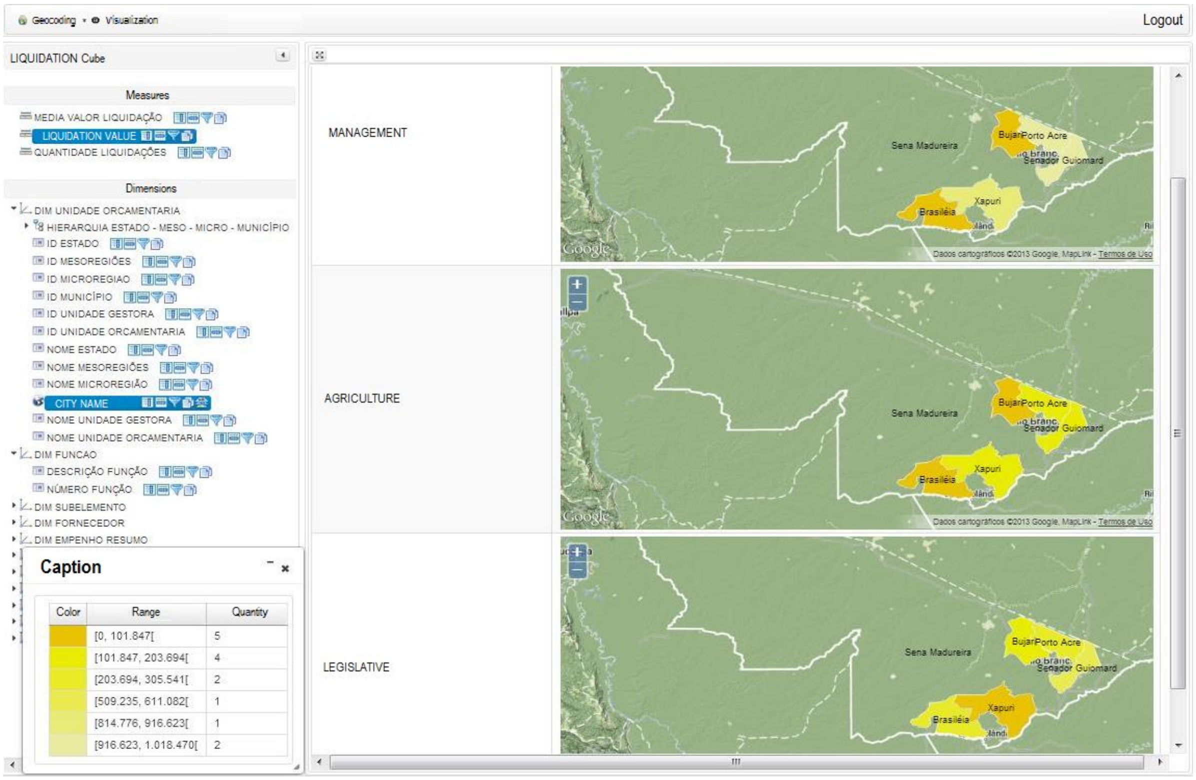
[✓ Value range: 8]

Color: 🟧

Caption

Color	Range	Quantity
🟧	[96.780, 26.230.050[	18
🟨	[235.296.210, 261.429.480[	1

Figure 10. Query 2 result.



Caption

Color	Range	Quantity
🟧	[0, 101.847[	5
🟨	[101.847, 203.694[	4
🟨	[203.694, 305.541[	2
🟨	[509.235, 611.082[	1
🟨	[814.776, 916.623[	1
🟨	[916.623, 1.018.470[	2

Figure 11. Query 3 result.

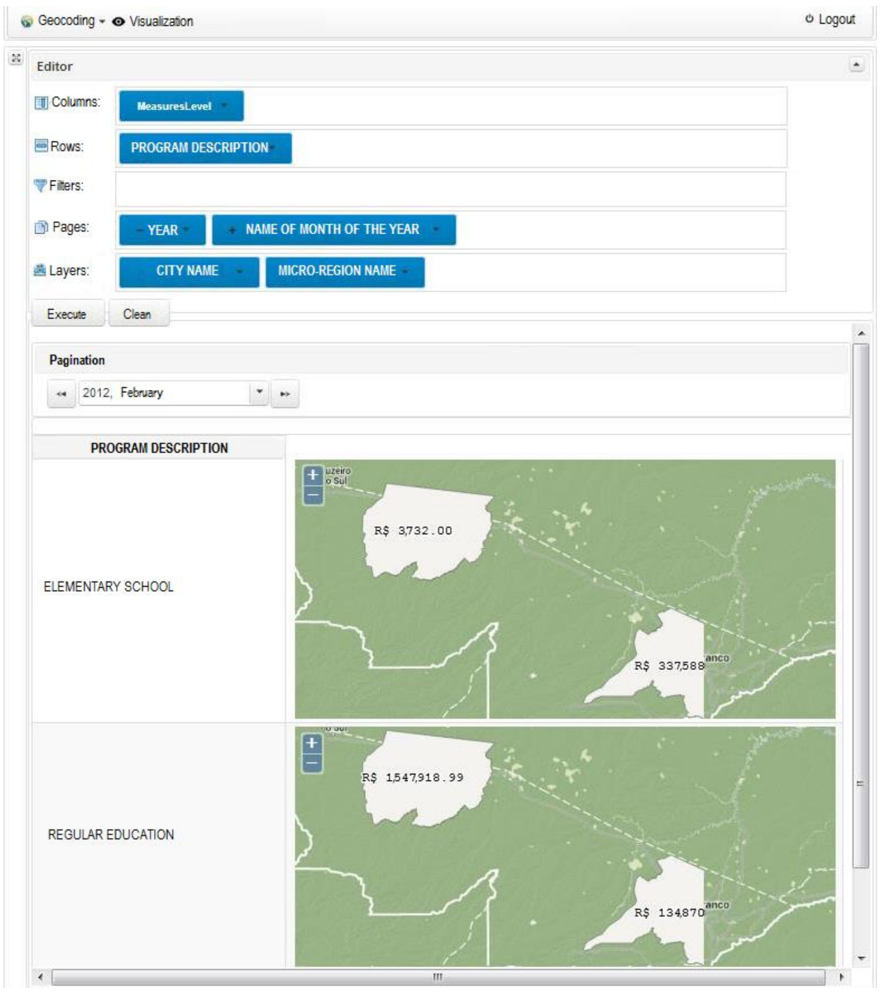


Figure 12.Query 4 result

V. Conclusions and Future Work

As presented in this work, several solutions for integration of spatial and multidimensional data have been proposed over the last few years. The main goal of these solutions was to propose improvements in the analytical process of data, providing a single environment to the multidimensional sources of spatial components analysis.

From the survey of the state-of-the-art presented, it was possible to compare the strengths and weaknesses of the main existing solutions. Because there is no standard to make spatial cubes available on the Web, we may conclude that these solutions do not provide techniques to analyze spatial cubes from heterogeneous multidimensional data sources. Based on the observed issues, a list of requirements needed for a SOLAP analysis tool was created, and the main requirements include the access to various sources of multidimensional data and data geocoding; support for topological spatial operators and a visual language for query specification and data visualization.

To fulfill these requirements, we presented a framework capable of performing spatial analytics on data provided by many sources of multidimensional spatial data. The main effort is to extend the SOLAP engine to access the data sources. Our framework has a set of graphic interfaces that enables the user to create connections to access data cubes; to perform analysis by visual query specification and data visualization; to create conventional or spatial filters to filter data; to geocode dimension members; to compare data by a tabular structure and to use maps to view the query results.

In the framework proposed in this work, our spatial cube model proved to be satisfactory for allowing spatial analysis in cubes from different sources. The extension of the visual language specification VizQL enabled the use of topological spatial operators and the display of data in maps, with overlaying. The framework also provides support for spatial analysis, through a data integration federated approach and made a geocoding service available, providing spatial analysis in pure OLAP servers.

In order to allow the user access to heterogeneous data sources, extension points were created and documented so that the framework can be extended. The Microsoft SSAS and GeoMondrian cube servers were accessed by extensions from the proposed framework, making possible to evaluate features by a practical example based on daily real problems, showing that spatial analysis in cubes from different sources can be performed efficiently. The example obtained relevant data to solve real problems related to a Brazilian Court of Audit (TCE-AC).

A comparative evaluation between the framework proposed in this work and related solutions shows that the framework addressed the necessary features to a SOLAP solution. Therefore, we conclude that this work achieved its goals, having as the main contribution an architecture in which spatial cubes from different multidimensional data sources can be analyzed, fulfilling the proposed requirements.

We also conclude that many expansion points can be explored in future works, which will cooperate for building a more robust framework.

As a future work we point out the following issues to be undertaken:

- Interoperability: although this solution enables analyzing cubes from different data sources, it does not allow the interoperability between cubes. As a future work, a suggestion is the development of a module to provide interoperability between cubes;
- SOLAP Operators**:** adding new SOLAP operators to the framework;
- Data Availability: adapting the proposed framework architecture to provide services via Web Services;
- Usability: an evaluation study of the interface concerning usability;
- Data visualization: investigate new forms of data visualization;
- Raster representation for data: adding raster data into the framework; and
- Data mining: adding data mining techniques to promote prediction.

Acknowledgment

The authors would like to thank the Court of Accounts of the State of Acre – Brazil (TCE-AC).

References

[1] T. E. Silva, D.F.B, Leite, and C.S. Baptista, "SOLAP_Frame: A Framework for SOLAP using Heterogeneous Data Sources," The Eighth International Conference on Advanced Geographic Information Systems, Applications, and Services - GEOProcessing 2016, April 24 - 28, 2016 - Venice, Italy.

[2] S. Rivest, Y. Bédard, M. Proulx, F. Hubert, and J. Pastor, "SOLAP technology: Merging business intelligence with geospatial technology for interactive spatio-temporal exploration and analysos of data," ISPRS P&RS, vol. 60, no. 1, 2005, pp. 17-33.

[3] S. Aissi, M. S. Gouider, T. Sboui, L. B. Said, "Enhancing spatial data warehouse exploitation: A SOLAP recommendation approach," SNPD 2016, pp. 457-464.

[4] J. Li, L. Meng, F. Z. Wang, W. Zhang, Y. Cai, "A Map-Reduce-enabled SOLAP cube for large-scale remotely sensed data aggregation," Computers & Geosciences, vol. 70, pp. 110-119, 2014.

[5] L. Leonardi, S. Orlando, A. Raffaetà, A. Roncato, C. Silvestri, G. L. Andrienko, N. V. Andrienko, "A general framework for trajectory data warehousing and visual OLAP," GeoInformatica, no. 18, vol. 2, pp. 273-312, 2014.

[6] B. A. A. Diallo, T. Badard, F. Hubert, S. Daniel, "Context-based mobile GeoBI: enhancing business analysis with contextual metrics/statistics and context-based reasoning," GeoInformatica no. 18, vol. 2, pp. 405-433, 2014.

[7] P. Rigaux, M. Scholl, A. Voisard, "Spatial Databases with Application to GIS," Morgan Kaufmann, 2001.

[8] E. Dubé, T. Badard, and Y. Bedard, "XML enconding and Web Services for Spatial OLAP data cube exchange: an SOA approach," CIT, vol. 17, no 4, 2009, pp. 347-358.

[9] Open Geospatial Consortium (OGC), "Open Geospatial Consortium OGC(R)," [Online]. Available: http://www.opengeospatial.org/, last access date: 11/10/2016.

[10] Open Geospatial Consortium (OGC), "Geospatial Business Intelligence (GeoBI)," OGC White Paper, 2012.

[11] M. E. Fayad and D. C. Schmidt, "Object-oriented application frameworks," Communications of the ACM, vol. 40, no. 10, 1997, pp. 32-40.

[12] I. Sommerville, Software Engineering, 10th edition, 2015, Pearson.

[13] L. Gómez, B. Kuijipers, B. Moelans, and A. Vaisman, "A Survey of Spatio-Temporal Data Warehousing," JDWM, vol. 5, no. 3, 2009, pp. 28-55.

[14] S. Bimonte, A. Tchounikine, M. Miquel, and F. Pinet, "When Spatial Analysis Meets OLAP Multidimensional Model and Operator," International Journal of Datawarehousing and Mining, vol. 6, no. 4, 2010, pp. 33-60, IGI Publishing.

[15] G. Viswanathan and M. Scheneider, "On the requirements for user-centric spatial data warehousing and SOLAP," in Proceedings of the 16th DASFAA, 2011, pp.144-155.

[16] M. Salehi, Y. Bédard, and S. Rivest, "A formal Conceptual Model and Definition Framework for Spatial Datacubes," Geomatica, vol. 64, no 3, 2010, pp. 313-326.

[17] P. Aguila, R. Fidalgo, and A. Mota, "Towards a more straightforward and more expressive metamodel for SDW modeling," in DOLAP 2011, pp. 31-36.

[18] O. Baltzer, "Computacional Methods for Spatial OLAP," Ph.D. thesis, 2011.

[19] O. Glorio and J. Trujillo, "Designing Data Warehouses for Geographic OLAP Querying by Using MDA," in Proceedings of the International Conference on Computational Science and its Applications, 2009, pp. 305-519.

[20] T. Ziouel, K. A. Derbal, and K. Boukhalfa. "SOLAP On-the-Fly Generalization Approach Based on Spatial Hierarchical Structures," CIIA 2015, pp. 279-290.

[21] S. Bimonte, A. Tchounikine, and M. Miquel, "Spatial OLAP: Open Issues and a Web Based Prototype," in M. Wachowicz & L. Bodum (Eds.), Proceedings of the 10th AGILE International Conference on Geographic Information Science, 2007.

[22] S.Bimonte, A web-based tool for spatio-temporal multidimensional analysis of geographic and complex data. In: Papajorgji, P.(ed). New technologies for constructing complex agricultural and environmental systems. Chapter 3, 2012, IGI Global.

[23] A. Escribano, L. Gomez, B. Kujipers, and A. Vaisman, "Piet: a GIS-OLAP implementation," in Proceedings of the ACM 1oth international workshop on Data Warehousing and OLAP, 2007, pp. 73-80.

[24] J. Li, L. Meng, F. Z. Wang, W. Zhang, and Y. Chai, "A Map-Reduce-enabled SOLAP cube for large-scale remotely sensed data aggregation," Computers & Geosciences, vol. 70, 2014, pp. 110-199.

[25] S. Aissi, M. S. Gouider, T. Sboui, and L. B. Said, "Personalized recommendation of SOLAP queries: theoretical framework and experimental evaluation," SAC 2015, pp. 1008-1014.

[26] M. Scotch and B. Parmanto, "SOVAT: Spatial OLAP Visualization and Analysis Tool," in Proceedings of the Proceedings of the 38th Annual Hawaii International Conference on System Sciences (HICSS'05) - Track 6 - Volume 06, Washington, DC, USA, 2005.

[27] J. Silva, V. Times, and A. Salgado, "An Open Source and Web Based Framework for Geographical and Multidimensional Processing," SAC 2006, pp. 63-67.

[28] C. Stolte and P. Hanrahan, “Polaris: A System for Query, Analysis and Visualization of Multi-Dimensional Relational Databases,” in Proceedings of the IEEE Symposium on Information Vizualization 2000, Washington, DC, USA, 2000.

[29] A, Lamas, F. Sotelo, M. Borobio, and J.I. Varela “Creación de un módulo espacial OLAP para SAIKU” VII JORNADAS DE SIG LIBRE, 2013.

[30] C. Stolte and P. Hanrahan, “Polaris: A System for Query, Analysis and Visualization of Multi-Dimensional Relational Databases,” in Proceedings of the IEEE Symposium on Information Vizualization 2000, Washington, DC, USA, 2000.

[31] P. Hanrahan, “VizQL: A Language for Query, Analysis and Visualization”, SIGMOD 2006, June 27-29, 2006, Chicago, Illinois, USA. ACM, 2006.