Metadata for Creating and Displaying Dashboard in XML Schema

Rosni Lumbantoruan
Department of Information System, Del Institute of Technology

Keywords:

Metadata
Dashboard
Dashboard design
Dashboard visualization
Best practice

ABSTRACT

Dashboard is a visual graphical interface that should give complete global information to the users at a glance. The information for each dashboard can vary in content and data structure, depends on the business processes of the company or organization. Furthermore, the same business process with different target viewer may need different information displayed on its dashboard.

Information placed in a dashboard need to be organized pretty well. It consumes time, because there is a need to analyze information to be placed, data to be chosen, visualization that fits the best practice to display the data based on its characteristics.

The main objective of this research is to simplify the task of customizing dashboard, so there are some features offered: 1) suggested data to be displayed based on data source, 2) suggested chart that met the best practice for visualizing data, and 3) saving the preference of the user after doing some modification for the dashboard display, such as coloring and positioning. All of these features are fulfilled by utilizing the best usage of metadata, which grouped as metadata for data source, visualization, and user's preferences.

This research came out with an application that can extract metadata to help user choosing the relevant information to be displayed, suggesting chart for data representation, and save user's preferences display.

Copyright © 2013 Information Systems International Conference.
All rights reserved.

Corresponding Author:

Rosni Lumbantoruan, M. ISD
Department of Information System
Del Institute of Technology
Jalan Sisingamangaraja, Sitoluama, Laguboti, Toba Samosir, Sumatra Utara, Indonesia.
Email: rosni.ltoruan@gmail.com

1. INTRODUCTION

Dashboard is a graphical presentation of the current status or historical trends of an organization's key performance indicators to enable instantaneous and decisive action to be made [1]. For example, a coal mining organization might have a key performance indicator (KPI) related to productivity such as the ratio of the capacity of the coal extracted based on the regional geological conditions, overburden characteristics, coal seam continuity, thickness, structure, quality, and depth, climate, and others. Similarly, a supervisor dashboard may show KPIs related to vehicle operations, for example number fuel usage, distance operation, and number of coal carry out by the vehicle. That information will enable instantaneous and informed decisions to be made by the decisions maker, for example, a supervisor can decide when to ask the fuel staff to go to the mining field to fill the tank of the vehicles and their current locations.

Each dashboard can vary in content and data structure, for coal mining cases, a product manager might need dashboard to view productivity and the other conditions that influence the productivity itself such as wheatear condition; a supervisor might need to see the effectiveness and efficiency usage of the vehicle during the mining. The more information and the vary content to be monitored by certain people will lead to many dashboard's pages needed. It means there is a need to reorganize the information to be displayed on each dashboard based on its viewer such as the layout, the representation tools to be used (pie, line, vertical, horizontal chart, or table), and the alert at particular warning conditions.

Different with designing a website, designing dashboard is how to make the viewer have the same perceptions regarding information presented, so they can learn and make the right decisions of it [2]. Consequently, designing dashboard should follow design of best practices in layout and content representation. For example, what the spot that the viewer will directly put attention to, so we can decide where to put the most important information in a page, and the same goes to what graphical representation to be used based on data characteristics. By concerning all the stated factors, customizing dashboard, even will more consume time, because there is a need to analyze what information to be placed, the specific data should be chosen, visualization that fits the best practice to display the data based on its characteristics.

To save the dashboard designer time, in this research introduced application with dashboard metadata for creating and displaying dashboard. There are some features offered, they are: metadata for data source that will suggest data to be displayed based on given table data source; metadata for visualization that will suggest chart that met the best practice for visualizing data's characteristic, and metadata for user's preference that will save the preference of the user after doing some modification for the dashboard display, such as coloring, chart, and chart position.

2. RESEARCH METHOD

My research methodology requires gathering relevant sources and publications regarding the dashboard with design best practice and metadata concepts in order to produce an application with metadata for simplifying the creating and displaying of a dashboard. All the available sources analyzed and organized to meet the requirements of this application that built in C#. This application is able to retrieve data from database or query as data source, consequently in order to accomplish this task; data sample is taken from a data warehouse that is modeled with Star Scheme, named Hotel Voyage Star. This research using the data warehouse as the data sources to give the chance for any business needs to display their dashboard using this application, whether it is modeled with data vault, star scheme, snow flake, or others.

Current works related to dashboard metadata exist for one of the commercial dashboard, Pentaho. Pentaho covers the metadata by firstly define Pentaho domain that is separated into three layers: physical, abstract, and business layer [3]. In business layer, the user is able to define and restructure categories in order to meet the business needs. One of the differences between Pentaho with this research is data source; Pentaho use the physical database as the data source while this research using the modeled data warehouse, by this it means that the user business needs as the data source. It will open door to any type of modeled data warehouse to be displayed in dashboard regardless its modeled; data vault, snow flake, star scheme, or others.

2.1. Metadata

Metadata is structured information that describes, explains, locates, or otherwise makes it easier to retrieve, use, or manage an information resource. Metadata is often called data about data or information about information [4]. There are three main types of metadata, they are [4]:

- *Descriptive metadata* describes a resource for purposes such as discovery and identification. It can include elements such as title, abstract, author, and keywords.
- *Structural metadata* indicates how compound objects are put together, for example, how pages are ordered to form chapters.
- *Administrative metadata* provides information to help manage a resource, such as when and how it was created, file type and other technical information, and who can access it.

Based on the literature study, type of metadata that fits the needs of this research is the descriptive metadata. Metadata can be instantiated in a variety of standard forms, e.g. XML, UCD tags, or FITS keywords [5]. In this research we defined the metadata in XML Schema by concerning the clarity and reuse, extensibility, and integration with tools [6]. Metadata defined in this research should be able to define and separate between the elements and types, it's not fully supported if using object-oriented languages. However, XML schema makes a distinction between the two of them. XML types are used to represent metadata types while XML elements represent metadata, each having a name, a definition, and a type [6]. Regarding extensibility and reuse, more complex metadata can be extended by properly defined namespaces. And the last is XML schema can be generated automatically by several tools such as Java's APIs for XML Binding and Messaging (JAXB and JAXM), Apache's Axis, and Microsoft's XSD. Tools that help integrate XML with relational databases are also expected to be important [6].

2.2. Metadata of data source and data structure

Dashboard should contain information that need to be monitored and help the management to come out with the right decisions. To place the right information from the data sources might be one of the most consuming times for dashboard analyst. In order to do so, the data analyst should understand the structure of the data sources, for example database and choose the right tables (relevant fields/columns that will answer

Copyright © 2013 ISICO

the requirement). In saving the analyst time, the application will read the metadata of the chosen data sources, so once the analyst chooses the table, application will suggest field that can be defined as axis or ordinate on a chart. Furthermore, it will also give all the related information to that table (identified from the relationship between tables in data sources) that is primary and foreign key references.

2.3. Metadata for user preferences

Once the dashboard has been build, this application enables user to customize the dashboard display and refers to it when the user signed back. Commonly, chart or representation tools in share some similar elements, such each chart has font name, font style, font size, drawing style, data to display (data source), type (pie, horizontal or vertical bar chart, table, and so on), series, and title. All the similar elements will be retrieved and saved in one XML File. This XML File will be load once the application runs, so each representation tools will refer this XML File as the chart default value. Once the user customizes the chart based on her preferences, again another XML File will be created to save all the changes value.

2.4. Metadata of design best practice

In order to be able to present the information in a way that people can rapidly monitor, fully understand, and effectively respond to, dashboard design must follow visual design principles for formatting and arranging information on the screen. Visual design principles helps to display and the deliver the information well by minimizing the different perception between the users about the displayed information together with what is intended to be [7]. Each piece of information that will appear on a dashboard should be weighed relative to all others and this hierarchy of importance ought to be considered when deciding positioning what goes where on the screen. The relative prominence of screen space on a dashboard can be divided into quadrants, as shown in the figure 1. The most important information should be considered to be placed in the upper left hand region, while the least important in the lower right hand corner [7].

Emphasized	Neither emphasized Nor de-emphasized
Neither emphasized Nor de-emphasized	De-emphasized

Figure 1. Positioning content based on its importance

Ranking the importance of information still will be the task of the dashboard analyst; the application will get the rank input and then save it into metadata. So the metadata scheme will consist of the data visualization that should be displayed, the rank of the information on chart based on its importance and the identification of generated XML file itself.

3. RESULTS AND ANALYSIS

This section will give the result as well as the discussion of metadata used for creating and displaying dashboard, they are: metadata for data source and data structure, metadata for user preferences, and metadata for design best practice.

3.1. Metadata for data source and data structure

This metadata is created each time the application runs. First, the application will read the metadata from the existing database metadata (for example system catalog in Ms. SQL Server) and then create metadata that later on will be used by the application to identify the data source structure such as total number tables, table's name, number of columns, primary, and foreign key. The generated XML scheme for data source and data structures is given below.

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema id="Database"
   targetNamespace=……
   xmlns=……..
>
  <xs:element name="DATABASE"><xs:complexType> <xs:sequence>
      <xs:element name="TABLE" maxOccurs="unbounded"><xs:complexType><xs:sequence>
          <xs:element name="COLUMNS" maxOccurs="unbounded"><xs:complexType><xs:sequence>
              <xs:element name="COLUMN" maxOccurs="unbounded">
                <xs:complexType> <xs:sequence>
```

Copyright © 2013 ISICO

```
                    <xs:element name="TYPE"type="xs:string"></xs:element>
                  </xs:sequence>
                  <xs:attribute name="fieldname"></xs:attribute>
                </xs:complexType></xs:element></xs:sequence>
              <xs:attribute name="count" type="xs:string"></xs:attribute>
            </xs:complexType></xs:element>
          <xs:element name="PRIMARYKEY">
            <xs:complexType><xs:sequence>
              <xs:element name="COLUMNPK" type="xs:string" maxOccurs="unbounded"></xs:element>
            </xs:sequence>
            <xs:attribute name="countPK"></xs:attribute>
          </xs:complexType></xs:element>
          <xs:element name="FOREIGNKEYS">
            <xs:complexType><xs:sequence>
              <xs:element name="FOREIGNKEY" minOccurs="0" maxOccurs="unbounded">
                <xs:complexType><xs:sequence>
                  <xs:element name="TABLEREF"
                  type="xs:string"></xs:element>
                  <xs:element name="COLUMNREF" type="xs:string"></xs:element>
                </xs:sequence>
                <xs:attribute name="field"></xs:attribute>
              </xs:complexType></xs:element></xs:sequence>
            <xs:attribute name="countKey"></xs:attribute>
          </xs:complexType></xs:element></xs:sequence>
        <xs:attribute name="name" type="xs:string"></xs:attribute>
      </xs:complexType> </xs:element></xs:sequence>
    <xs:attribute name="tblcount" type="xs:int"></xs:attribute>
  </xs:complexType></xs:element>
</xs:schema>
```

Once the dashboard analyst determined one of the tables from the database as data source, using this metadata, the application will reveal all the related tables and displayed to the analyst. The analyst then will choose which columns or field to select in order to answer the information needed. By having this metadata, whatever the information fed to the application, it will help the analyst to identify the related tables from the given database structure.

3.2. Metadata for user preferences

This metadata is used in order to store the similar elements of representation tools in dashboard. As stated above, this metadata is used once the application loaded and the representation tools will have default value as defined in this XML File.

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema id="Database"
   targetNamespace=……
   xmlns=……..
>
  <xs:element name="ChartSchema"><xs:complexType><xs:sequence>
      <xs:element name="dashboard.chart"><xs:complexType><xs:sequence>
          <xs:element name="chartName" type="xs:string"></xs:element>
          <xs:element name="query" type="xs:string"></xs:element>
          <xs:element name="dataSoure" type="xs:string"></xs:element>
          <xs:element name="showChart" type="xs:boolean"></xs:element>
          <xs:element name="chartType" type="xs:string"></xs:element>
          <xs:element name="drawingStyle" type="xs:string"></xs:element>
          <xs:element name="fontName" type="xs:string"></xs:element>
          <xs:element name="fontStyle" type="xs:string"></xs:element>
          <xs:element name="fontSize" type="xs:float"></xs:element>
          <xs:element name="chartTitle" type="xs:string"></xs:element>
          <xs:element name="titleAligment" type="xs:string"></xs:element>
          <xs:element name="titleFont" type="xs:string"></xs:element>
```

Copyright © 2013 ISICO

```
          <xs:element name="titleSize" type="xs:float"></xs:element>
          <xs:element name="titleStyle" type="xs:string"></xs:element>
          <xs:element name="titleForeColor" type="xs:string"></xs:element>
          <xs:element name="showLegend" type="xs:boolean"></xs:element>
          <xs:element name="legendDocking" type="xs:string"></xs:element>
          <xs:element name="seriesPointWidth" type="xs:integer"></xs:element>
          <xs:element name="seriesCount" type="xs:integer"></xs:element>
          <xs:element name="axisLabel" type="xs:string"></xs:element>
          <xs:element name="ordinatLabel" type="xs:string" maxOccurs="unbounded">
<xs:element name="warOrdinatLabel" type="xs:int" maxOccurs="unbounded"/>
<xs:element name="danOrdinatLabel" type="xs:int" maxOccurs="unbounded"/>
          </xs:element>
        </xs:sequence></xs:complexType></xs:element>
    </xs:sequence></xs:complexType></xs:element>
</xs:schema>
```

On the other hand, once the dashboard user customizes the dashboard page based on her preferences, some attributes that are enabled to be customized; height, weight, border and so on will be saved in another XML File with scheme as below. This metadata will be referred when the same user log in to the dashboard.

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema id="Database"
   targetNamespace=……
   xmlns=……..
>
 <xs:element name="EditDashboardChart">  <xs:complexType> <xs:sequence>
      <xs:element name="DashboardChart"> <xs:complexType> <xs:sequence>
          <xs:element name="chartHeight" type="xs:integer"></xs:element>
          <xs:element name="chartWeight" type="xs:integer"></xs:element>
          <xs:element name="chartBackColor" type="xs:string"></xs:element>
          <xs:element name="borderSkinStyle" type="xs:string"></xs:element>
          <xs:element name="chartBorderWidth"type="xs:integer"></xs:element>
                    <xs:element  name="legendFontSize" type="xs:int"/></xs:element>
                    <xs:element  name="legendFont" type="xs:string"/></xs:element>
                    <xs:element  name="legendForeColor"type="xs:string"/></xs:element>
                    <xs:element  name="legendBackColor"type="xs:string"/></xs:element>
                   <xs:element name="legendBorderColor"type="xs:string"/></xs:element>
           <xs:element name="legendBorderWidth"  type="xs:integer"/></xs:element>
        </xs:sequence>  </xs:complexType>  </xs:element>
    </xs:sequence> </xs:complexType>  </xs:element>
</xs:schema>
```

3.3. Metadata for design best practice

As stated in 2.4 the XML scheme should consist of data visualization, rank of the information based on its importance, and identification of generated XML file.

- Position, data visualization that was ranked by the analyst based on its importance will be then saved in metadata in tag named “position”. This tag type is integer, in which the smallest the value, the most important the information will be. By this it means, when the analyst give value 1 to data visualization, then it should be displayed on the upper left hand region.
- Chart id, used to identify each chart or data visualization that will be putted on dashboard page.
- XML File, used to store the name of generated XML file. This XML file will be called back in order to display all the charts (data visualizations) into dashboard page.

The generated XML scheme for metadata for best practice design is shown below.

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema id="Database"
   targetNamespace=……
   xmlns=……..
>
  <xs:element name="dashboardChart"> <xs:complexType>
     <xs:choice maxOccurs="unbounded">
```

Copyright © 2013 ISICO

```
      <xs:element name="chart"> <xs:complexType><xs:sequence>
          <xs:element name="id" type="xs:int"></xs:element>
          <xs:element name="xmlFile" type="xs:string"></xs:element>
          <xs:element name="position" type="xs:string"></xs:element>
        </xs:sequence></xs:complexType></xs:element>
    </xs:choice>
  </xs:complexType></xs:element>
</xs:schema>
```

4. CONCLUSION

As the result of this research, an application built to help simplifying tasks for customizing dashboard. Once the dashboard analyst choose the data sources of the information, the application will guide the her to complete the dashboard by proposing the potential and relevant data to be displayed, the representation tools that met the best practice design, and a function that enable the analyst to save her preferences after customizing the dashboard view.

However, this research cannot yet fully cover the best practice aspects because the representation tools to display the information are decided by the analyst, in which her choices might not fit properly with the concept of best design. The Analyst might choose a vertical bar chart to display trends information that should be correctly represents by a line chart. So, the deepest research on analyzing data characteristics is needed; it will help to identify the best tool to represent the information instead of asking the designer to choose her preferences. Moreover, another improvement regarding the data sources is still needed, for the application data source could be from only one database and SQL syntax, it might needed to think of data source from vary databases and different DBMSs.

ACKNOWLEDGEMENTS

I would like to express my deepest thanks to my former students Januar Sirait, Marthadinata Hutagaol, and Winda Tampubolon that have been proof as the work-hardest students and worked on the application based on the given research idea and concepts. I would also thank all of my colleagues on data management cluster for valuable reviews on content and editing of this paper.

REFERENCES

[1] S. Few, "Dashboard Design for Real-time Situations Awareness," *Perceptual Edge*, Berkeley, 2007.
[2] S. Few, "With Dashboards Formatting and Layout Definitely Matter," 2008.
[3] G. Moran, "Metadata Business Model Overview," Legacy BI Server Documentation [cited: 10 September 2013]. Available from: http://wiki.pentaho.com/
[4] NISO, "Understanding Metadata," *NISO Press*, ISBN 1-880124-62-9.
[5] B. Hanisch, "Resource and Service Metadata for the Virtual Observatory Version 6," February 2002.
[6] R. Plante, "VOResource: a case study in rendering a metadata model in XML Scheme [updated: 17 April 2003. Cited: 10 September 2013]," Available from: http://wiki.ivoa.net/internal/IVOA/IVOARegWp03/MDinXML-Summary.html/
[7] S. Few, "Information Dashboard Design," *O'Reilly*, Italy, 2006.

BIBLIOGRAPHY OF AUTHORS

Rosni Lumbantoruan, a lecturer and the head of Data Management cluster of study area in Del Institute of Technology. She granted the bachelor degree of Engineering Informatics from Bandung Institute of Technology in 2007 and master degree of Information System Development from HAN University of Applied Sciences, The Netherlands in 2010.
Her research interests are mainly on business intelligence, data management, and information systems development.

Copyright © 2013 ISICO