ISBN 978-981-11-0008-6
Proceedings of 2016 6th International Workshop on Computer Science and Engineering (WCSE 2016)
Tokyo, Japan, 17-19 June, 2016, pp. 407-410
doi: 10.18178/wcse.2016.06.066

DSL for Business Intelligence Visualization

Ittipol Nuntaruji and Yachai Limpiyakorn[+]

Department of Computer Engineering, Chulalongkorn University, Bangkok 10330, Thailand

Abstract. Business Intelligence (BI) encompasses a variety of tools, applications and methodologies that enable organizations to collect and prepare data for analysis, develop and run queries against the data, and create reports, dashboards, data visualizations to make the analytical results available to end users. However, most BI solutions in the market are platform-dependent and require particular software for rendering. This paper thus presents a platform- independent approach for visualizing data and information using domain specific language (DSL). The DSL syntax is defined with EBNF. The use of DSL scripts for rendering provides the benefit of platform-independent as the language simply focuses on primitive data, whereas the library for data visualizations remains intact.

Keywords: domain specific language, data visualization, metadata, semantic model, business intelligence.

1. Introduction

Business intelligence (BI) is a technology-driven process for analysing data and presenting actionable information to help corporate executives, business managers, and other end users make more informed business decisions [1]. Over the last few decades, the growth of BI has enabled companies' success in managing their information assets. New BI technologies have been facilitating mass collaboration and innovation. In the market, a lot of BI applications exist, namely Pentaho [2], Jaspersoft [3], SQL Server Reporting Services [4]. These BI solutions are powerful and rich of features. However, these software systems are platform-dependent and require particular software for rendering. This paper thus presents a platform- independent approach for visualizing data and information using domain specific language (DSL). The design of DSL is not subject to the visualization styles of those components used for decision analysis. As a result, there will be no constraints imposed on the criteria when selecting the libraries for developing BI visualization.

2. Domain Specific Language

Domain Specific Languages (DSLs) are the computer programming language of limited expressiveness focused on a particular domain [5]. DSLs can help domain experts to write, understand, and verify the problem with the less programming skill because the domain knowledge is specified at an appropriate level of abstraction [6]. The use of DSLs would improve productivity and quality [6-7]. However, the design and development of DSLs are often subject to the challenging of finding the right abstractions and appropriate syntax [8], in addition to balancing between domain-specificity and general-purpose programming language constructs [7].

3. Development of DSL

In this research, a domain specific language is developed to facilitate BI visualization. The work contributes to the visualization part of the academic project of Quality Assurance in curriculum development.

[+] Corresponding author. Tel.: + 668 2218 6959; fax: +668 2218 6955.
E-mail address: Yachai.L@chula.ac.th.

The large amount of data will be collected from internal systems, such as office of the registrar, HR database, and external sources such as questionnaires.

The process of DSL construction consists of four main steps: 1) Domain analysis, 2) Meta-model design, 3) DSL syntax design, and 4) Parser development. Initially, the primitive visual components are identified, and the relevant details are gathered during domain analysis. The collected information will then be used for the design of metadata model, which in turn, is used for defining the DSL syntax. Next, the parser is developed and it will be used for transforming a DSL script to the corresponding semantic model, which will be used for rendering step.

3.1. Domain analysis and meta-model design

During domain analysis, the visual components are obtained from the survey of BI applications and supporting API existing in the market. The domain metadata model of the visual components is then constructed as illustrated in Fig. 1.

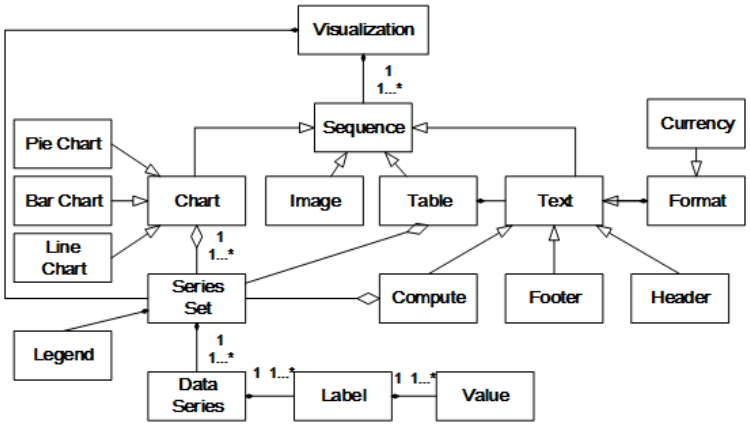

Fig. 1: Domain metadata model of visual components.

3.2. DSL syntax design

The proposed DSL grammar is built with EBNF [9]. The use of DSL scripts for rendering provides the benefit of platform-independent as the language simply focuses on primitive data. The visual data are not included because it is based on the library selected when development. Example of DSL syntax defined for visual components is shown in Fig. 2.

```
visual = 'visualization' [id] '['{visual_attrs} ']';
visual_attrs = datasource |image |table |chart |text ;
datasource = 'data' [id] '=' data_value;
image = 'image' [id] '=' data_value;
table = 'table' [id] '['{table_attrs} ']';
table_attrs = table_title |table_paging |datasource |table_select_cols[',' table_row_group];
table_title = 'title' '=' data_value;
table_paging = 'num-row' '=' data_value;
table_row_group = 'group' table_cols;
table_select_cols = 'select' '=' table_cols;
table_cols =  '('[data_field_name]data_field{','[data_field_name]data_field}')';
chart = chart_type [id] '['{chart_attrs} ']';
chart_attrs = chart_title |datasource |chart_series;
chart_type = 'barchart'|'piechart'|'linechart';
chart_title = 'title' '=' data_value;
chart_series = 'series-set' [id] '['{series_attrs} ']';
series_attrs = series_cond |datasource |series_label |series_value ;
series_cond = 'condition' '=' data_field_value '('data_cond{',' data_cond}')';
series_label = 'label' '=' data_field_value;
series_value = 'value' '=' data_field;
text = ('text'|'header' header_style) [id] '=' data_value;
header_style = 'H1'|'H2'|'H3';
data_field = aggregate_field |data_field_value |data_value;
aggregate_field = (Sum|Count|Avg|Min num|Max num)'('data_field')';
num = integer;
data_field_value = '<<'string'>>';
```

Fig. 2: Excerpt of DSL syntax defined for visual components.

4. Demonstration

An example of a DSL script (Fig. 3) is provided as input for the BI visualization as shown in Fig. 4. The script contains a header, line image, sub header, normal text, table, pie chart, and line chart. The semantic model generated from the input DSL script is illustrated in Fig. 5.

```
visualization
[
    data students = "SELECT * ,CASE Grade WHEN 'A' THEN
    4 WHEN 'B' THEN 3 WHEN 'C' THEN 2 WHEN 'D' THEN 1
    ELSE 0 END AS GradeVal FROM students"
    header H1  = "Grade Report"
    image  = "Line.jpg"
    header H2 = "2110424 Software Process Improvement"
    text = "Semester 2/2015"
    text = "Number of Students:"Count(students)
    table
    [
        num-row = "4"
        select = (<<Name>>,<<Year>>,<<Score>>,<<Grade>>)
    ]
    piechart
    [
        title = "Persons Grade"
        series-set
        [
            label = <<Grade>>
            value = Count(<<Grade>>)
        ]
    ]
    linechart
    [
        title = "Class GPA 2010 - 2015"
        series-set
        [
            label = <<Year>>
            value = Sum(<<GradeVal>>)/Count(students)
        ]
    ]
]
```

Fig. 3: Example DSL script

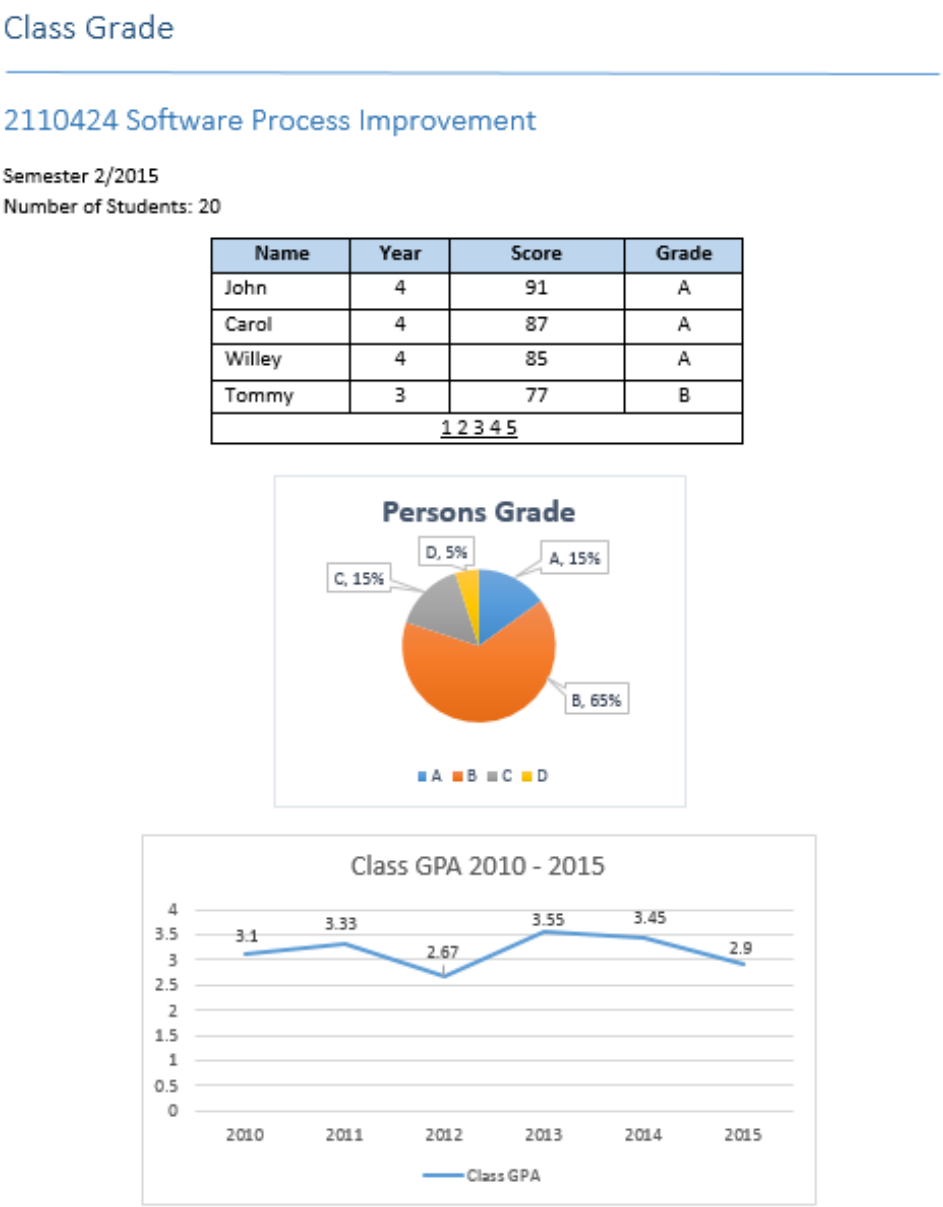

Fig. 4: Visual components rendered from input DSL script.

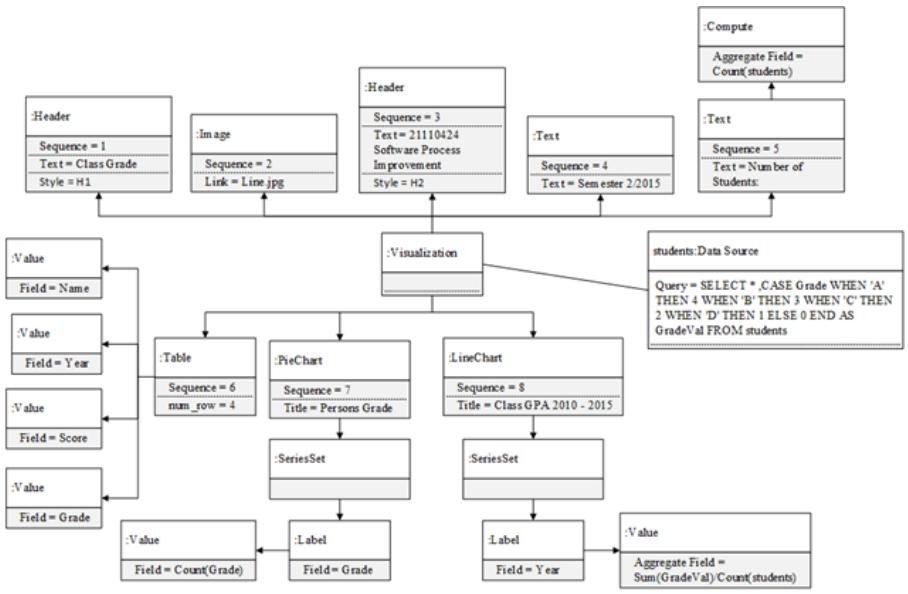

Fig. 5: Semantic model of example DSL script.

5. Conclusion

A well-designed BI solution provides a consolidated view of key business data, giving management visibility and control over measures defined by the organization. This paper presents a domain specific language developed for data visualizations in BI solutions. The parsed semantic model associated with the input DSL script will be used for rendering the visual components, providing no restriction on the visualization library selected. With the proposed approach, domain experts could themselves edit the DSL scripts to modify the data visualizations without waiting for the developer to rework. This would lead to improved project productivity.

Future directions would be further implementation of the DSL grammars to support additional visual components such as scatter chart, box plot. The enhancement of functionality to support condition will be also in consideration.

6. References

[1] M. Rouse. Definition business intelligence (BI). [Online]. Available: http://searchdatamanagement.techtarget.com/definition/business-intelligence.

[2] Pentaho. Pentaho. [Online]. Available: http://www.pentaho.com/.

[3] Jaspersoft. Jaspersoft. [Online]. Available: http://www.jaspersoft.com/.

[4] Microsoft. Microsoft. [Online]. Available: https://msdn.microsoft.com/en-us/library/ms159106.aspx.

[5] M. Fowler. *Domain-Specific Languages*. Addison-Wesley Professional, 2010.

[6] B. Bryant, J. Gray, M. Mernik. Domain-Specific Software Engineering. *Proc. of the FSE/SDP workshop on Future of software engineering research.* Santa Fe. 2010, pp. 65-68.

[7] A.V. Deursen, P. Klint, J. Visser. Domain-specific Languages: An Annotated Bibliography. *ACM SIGPLAN Notices*. June 2000, **35**(6): 26-36.

[8] T. Wegeler, F. Gutzeit, A. Destailleur, B. Dock. Evaluating the benefits of using domain-specific modeling languages: an experience report. *Proc. of the 2013 ACM workshop on Domain-specific modeling*. 2013, pp. 7-12.

[9] ISO/IEC. ISO/IEC 14977:1996(E) First edition -- Information technology -- Syntactic metalanguage -- Extended BNF. 1996.