Journal of Korea Multimedia Society Vol. 11, No. 12, DECEMBER 2008(pp. 1758-1764)

Creating Method for Multi-Size Web View Based on Query Restructuring Rules

Sang-gyu Shin†, Tai-suk Kim††

ABSTRACT

In this paper, we propose query restructure method which realizes not only to present the result from database in Web page but also to adapt user's browsing environments. Recently, wireless PDA (Personal Digital Assistant) and portable phone and terminals equipped with Web browser become very popular. Various browsers would be required to create the HTML which adapt to the user's browsing environment. We propose an adaptive data presentation system which is an extension of SQL which creates HTML pages dynamically adapting to the size of screen. In our proposed method, the three basic rules for the layout conversion are provided and applying these rules, an efficient layout conversion technique optimized to user environment is proposed. Moreover, we propose a standard for converting layouts and a layout optimization for adaptive conversion of layouts.

Key words: Multi-size web view, Creating web view, Query restructuring Rules, SQL extension

1. INTRODUCTION

Spreading use of WWW raises the importance of the application, which join the information from database and WWW technology. Furthermore, user needs for Web pages are increasing and diversifying with wider range use of WWW. It requires web provider band to provide adapted Web view optimizing the user's environment and personalizing contents for user.

Browsing WWW from Mobile devices such as wireless PDA, portable PC and mobile phone becomes popular and they are required to prepare Web pages adapted to each browsing environment. Also, Web technology helps many Web-based businesses which are required to customize the contents of Web pages for each user [1]. This leads to the idea that it is more effective to make the Web page adapted to individual than to obtain satisfaction of majority users. Therefore, to meet these requirements, the research of adaptation and personalization of Web data becomes important. We are here concerned with the situation in which corporate data in database to be delivered to workers through WWW.

In this paper, we propose an adaptive data presentation system [2], which creates HTML pages as a result of a query including the idea of adaptation. We define adaptation to convert the view layout according to the user display size. The adaptation attracts attention with small portable terminal and mobile phone widely used as information medium as well as PC. This research focuses on the use of database through the WWW to solve the adaptation problem. Many people can use information resources from various places by integrating WWW with database system. When many users share the information that is stored on database system through Web, we focus on offer-

※ Corresponding Author : Tai-suk Kim, Address : (614-714) 995 Eomgwagno, Busanjin-gu, Busan, KOREA, TEL : +82-51-890-1707, FAX : +82-51-890-2629, E-mail : tskim@deu.ac.kr

Receipt date : Oct. 5, 2008, Approval date : Nov. 27, 2008

† Dept. of Information and Computer Science Faculty of Science and Technology Keio University, Japan (E-mail : sg_shin@hotmail.com)

†† Dept. of Computer Software Engineering, Dongeui University, Korea

ing the view that adapted various user terminals such as PDA, mobile phone, desk top PC. Then we propose the adapted web view. Our research has the following features compare with other researches [3-5].

1. Based on retrieval result of the relational database.
2. Generate various layouts dynamically by transforming an original layout into an adapted layout according to the user's window size without specifying and limiting terminal environment.

Since our system use SuperSQL [6,7] processor which can publish various media such as HTML, PDF, XML and Excel using retrieval results of the relational database.

The rest of the paper is organized as follow. In Section 2, we briefly review the SuperSQL that the System which our system is based on. We present our proposed system in Section 3. Then, conclude the paper and discuss our future work in Section 4.

2. Overview of SuperSQL

2.1 Second-order headings

In this section, we will briefly review SuperSQL, which extends SQL with the TFE (Target Form Expression) [1] enerate various kinds of structured publishing / presentation documents.

TFE is an extension of target list of SQL. Unlike an ordinary target list which is a comma-separated list of attribute, TFE uses new operators (connectors and repeaters) to specify the structure of the document generated as a result of the query. Each connector and repeater is associated with dimension. When generating a Web document, the first two dimensions are associated with the columns and rows of the *<TABLE>* structure of HTML and the third dimension is associated with hyper-links. We have introduced the *GENERATE <medium> <TFE>* clause to SQL syntax to clarify the distinction with the *SELECT<target list>* clause.

2.2 Connectors and Repeaters

Binary operators represented by a comma (,), an exclamation point (!) and a percent (%) are used as the connectors of first three dimensions. Conceptually, they connect the materials generated as their operands horizontally, vertically, and in the depth direction, respectively.

A pair of square brackets ([ ]) followed by any of the above connectors is a repeater for that dimension. It will connect multiple instances in its associated dimension repeatedly. For example,

```
GENERATE HTML [c.name, c.nationality]!
FROM coach c
```

will connect name-nationality pairs vertically repeatedly until there are no such tuple in the query result; it yields the same result as an ordinary target list ensuring the backward-compatibility.

Nested repeaters introduces grouping if the inner repeaters are connected to one or more simple attributes. For example,

```
GENERATE HTML [ t.name,
                [ p.name, p.position ]!
                ]!
```

will nest player's name and position information for each division. The curly braces ({ }) are called grouper which specify the precedence of connections. Table 1 summarizes the role of operators. x in Table 1 represents attribute name or TFE sub-expression. Ornament operator (@) is used to specify decoration such as font size, font color, cell width etc.

```
t.name@{width=100, fontcolor=ff00ff}
```

Table 1. connector and repeater

connector	meaning of operator	example of use
,	connect horizontally	x1 , x2
!	connect vertically	x1 ! x2
%	connect with hyperlink	x1 % x2
repeater	**meaning of operator**	example of use
[ ],	repeat horizontally	[x],
[ ]!	repeat vertically	[x]!
[ ]%	repeat with hyperlinks	[x]%

3. OUR PROPOSED SYSTEM

Our proposed system is implemented on the basis of HTML generator of SuperSQL. The special feature of this new medium: Our system is that the result of a query changes among users and also changes among users' browsing environments. Fig. 1 shows the overview of the architecture of our proposed system.

A part rounded by the bold line is SuperSQL processing system, and a part rounded by dotted line is added part to realize our proposed system. To realize our proposed system, we added new modules, "Front End" and "Adaptation Processor". The following is the process after SuperSQL query has submitted to the processor.

- Front End

When a user access web site, the "Front End" processor snatches the user's browsing environment information (User's Web browser size).

- Preparser

When a SuperSQL query is submitted to the processor, the pre-parser modifies the query according to the proposed rules. Submitted query is parsed through our proposed system, the "Preparser" processor and is separated into a plain SQL, layout information, and width information of columns defined using SuperSQL decoration function.

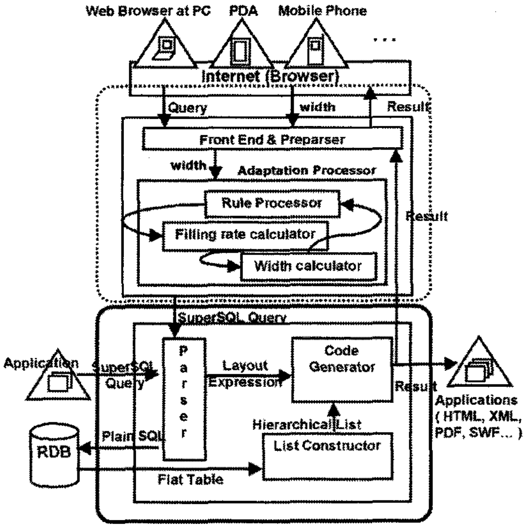


Fig. 1. Overview of architecture

- Adaptation Processor

The mainframe of the proposed system, which checks the size of resulting view created by modified query and changes the layout if it does not meet the criteria.

- SuperSQL system

Finally, the modified query in the "Adaptation Processor" according to rules is sent to SuperSQL system and it publishes HTML document that optimized user's browsing environment.

To reconstruct the view that is offered by the proposed system, it has TFE size definition using the SupreSQL's decoration description. If the proposed system has no decoration description about TFE size (column size), proposed system expects the column size using the column schema included in database. Fig. 2 shows the proposed system's query based on SuperSQL query. It has TFE descriptions and column size descriptions.

According to the query, our system parses the query to create tree structure that has construction information and layout size information. In Fig. 3, we can show the constructed tree structure based

```
GENERATE HTML
[ {"YEAR"@{width=60} , h.year@{width=70} } ,
 [{{"GROUP"@{width=80},t.group@{width=40}}
   %invoke(./teams.htmls, c.id=, c.id) } ,
    [ {"COUNTRY"@{width=120\} ,
        t.country@{width=160} } ,
      {"COACH"@{width=110} ,
        c.name@{width=260} } ,
      {"NATIONALITY"@{width=140\} ,
        c.nationality@{width=160} },
 ]! ]! ]!
FROM host h, teams t, coach c
WHERE h.id = t.id and h.id = c.entry ...
```

Fig. 2. Query

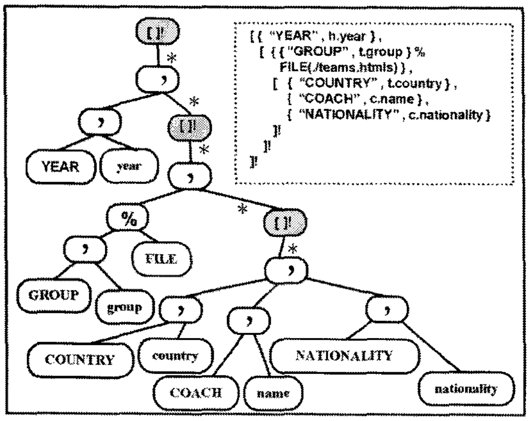


Fig. 3. Tree structure of layout information

on the proposed system's query at Fig. 2. This tree structure becomes the basic for our proposed system processing to recreate layout that adapted user browsing environments.

3.1 Adaptation

In general, when representing relation on the Web, it is inconvenient to see larger tables than window size. In the case of mobile devices like portable phone, which have tiny window, it is particularly tough to browse table which is too large. It is even more bothering to scroll both ways, horizontally and vertically.

In addition, user's browsing environment, even if it is a PC, has different screen width. First of all, small and portable terminal screen size is changed very quickly. Therefore, we focused on this problem and tend to modify the layout dynamically and adaptively to the browsing window. More accurately, modify the layout not to be larger than window width. We adopt two policies as shown below.

Two policies to modify layout

1. Change horizontal connection (,) to vertical connection (!).
2. Change horizontal connection or vertical connection (!) to connection which uses hyper-links (%).

First policy which changes the horizontal connection to vertical one is the basic way to change the layout longer rather than wider. Second policy which uses hyper-links is prepared especially for mobile devices. In general, the Web page for portable phone should not be wider than the screen width. This is because portable phone is not good for scrolling sideways compared with personal computers. Thus it is natural to represent data vertically, but we also think it is better to connect by hyper-link than to align too much data vertically. The representative information appears as a menu and details are connected by hyper-link. The introduction of hyper-link apparently reduces excessive scrolling effort.

3.2 3 basic rules

To create layout which is adapted user's browsing window, we propose 3 basic rules and discuss about applying these rules to case.

Rule 1. Horizontal connection change rule : Choose any of horizontal connection (,) and change to vertical connection (!). After applying rule, measure the width of modified layout. Apply this rule repeatedly until the modified layout of user's browsing width is satisfied.

In our experiment, this rule is applied with two plans which apply sequentially and according to patterns. The application at pattern is applied from small to many. In this research, we do not consider about "brace expression" which is SuperSQL's special expression. In SuperSQL, it is possible to define not only the data obtained from the database result as specified by query but also define the character-string content in query. It creates column name about database result. By character-string, the information became easier to understand. Character- string constant is bounded with the pair TFE by brace.

Rule 2. Character-string constant rule :
Apply the rule according to the 4 difference apply methods for a pair of character-string and an element.

Apply method 1 : Handle the character-string constant as a connector. It means character-string constant is controlled according to rule 1.

Apply method 2 : Apply rule 1 at the same time for TFE which includes the same nested group.

Apply method 3 : Apply rule 1 for all the character-string constants at first and after that, apply rule for another TFEs.

By experimental result, we recognize that it is easy for the user to fill and understand when change layout is united for character-string constants. For example, when query has 3 TFE character-string constants which are written in same nested group as follow:

[A [{C1 , B} , {C2 , D} , {C3 , E}] F, G]

It is easy for user to fill when *C1, C2, C3* are changed at the same time rather than changing separately.

Rule 3. Hyper-link rule :
When user's browsing environment is very tiny, our processor will change all the horizontal connection (,) to vertical connection (!) and then, the vertical connection to hyper-link connection (%).

3.3 Criteria for layout modification

The question arising here is to consider for the change priority of horizontal connection. There are much possibilities of layout to meet the condition not to be larger than window in width. We introduced five criteria for layout optimization.

1) Width condition

As we mentioned before, to change the layout not to exceed than window width is an absolute condition. To measure the modified layout width, we use tree structure of layout which is defined by TFE expression. Fig. 4 shows simple the proposed system's query and the tree structure of layout information which is created by our proposed system after applying rules.

Based on this tree structure, we apply the rules and measure the modified layout width. In Fig. 4, we can also show the changed size by rule. Table 2 shows how to calculate the modified layout width. Using this method we calculate the created

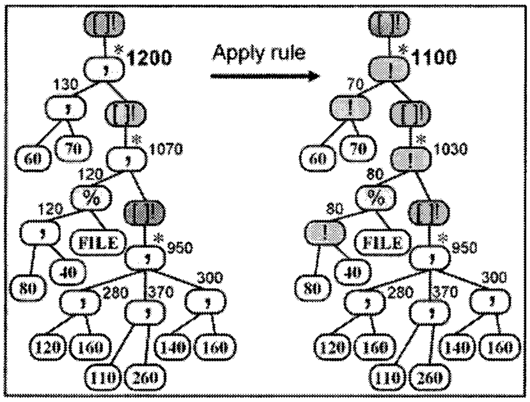


Fig. 4. Tree of layout-size expression

Table 2. Calculate each connector

Dim	Connector	Calculate method
1	E_i , E_{i+1}	(sum width(Ei) width(Ei+1))
2	E_i ! E_{i+1}	(max width(Ei) width(Ei+1))
3	E_i % *File*	*width*(E_i)

sum : () Add the values
max : () Choose max value

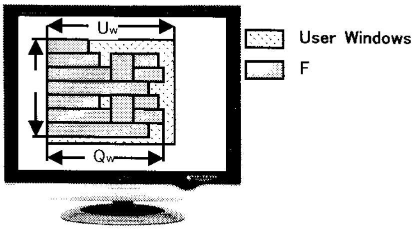


Fig. 5. Filling rate

layout (HTML view).

2) Filling rate of window

We think it is better to use area of window effectively within width restriction.

If the result layouts created by adaptation processor has same width, we choose the result layout which has highest filling rate value. We define filling rate of browser shown in Fig. 5 and also define cost evaluation function as following.

3.4 Layout modification

Fig. 6 shows the typical example of layout modification according to rules. Fig. 6 (a) is the basic structure of layout given by designer. Thus, "the order of attribute", "nesting" and "grouper" will be kept unchanged from Fig. 6 (a) during layout modification. (b) is after applied rule 1. (c) is horizontal connection between character-string constant and data from (a) has changed. (d) is the example of applying hyper-link connection.

The connection between "YEAR and h.year" pair and detail data changed into hyper-link (%) connection. Because this query is nested by year value, the menu page consists of "h.year" is made and other attributes are shown in another page associated to each year value.

When the initial layout by designer does not satisfy the width restriction, the system applied to layout modification. The algorithm to find the answer under given is the restrictions described above, changing the horizontal connection starts from top of tree structure of layout and whose

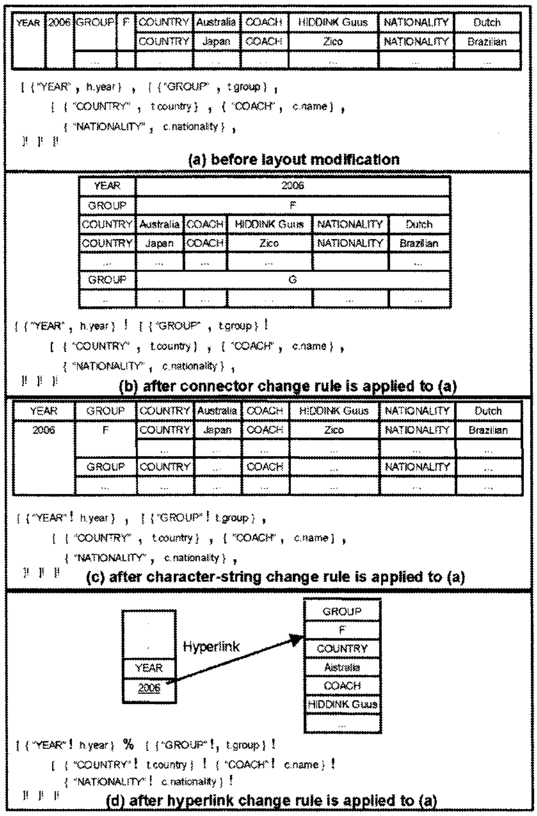


Fig. 6. Example of layout modification

filling rate of browser window is high becomes the answer of optimized layout.

4. CONCLUSION

Our proposed system enables to create HTML pages which dynamically adapt to the size of a browser. It allows browsing relations from various mobile devices and dynamically changed user browsing environments based on a given query. However, several further improvements are also needed. We summarize the future work as follows.

- Response time

During the layout modification, algorithm for searching the optimal layout remains as a matter to be discussed further.

- Optimized layout

To make sure that the layout after modification is useful and meaningful, we proposed five criteria to consider. Nevertheless, each criteria calls for further discussion and we should also think the new criteria for optimized layout.

The significance of accessing Web from various mobile devices is increasing in these days. Therefore, Web pages are required to adapt to various browsing environments. Our proposed system realizes not only to present the result from database on Web pages but also to adapt and personalize the presentation. Especially, it modify layout in order to fit automatically to screen size. To display a large relation is really challenging because it is difficult to reduce and summarize the contents. We would further improve the optimization algorithm of our proposed system so that more readable and more meaningful presentations could be obtained in a reasonable response time.

REFERENCES

[1] G. Adomavicius and A. Tuzhilin, "Personalization technologies: a process-oriented perspective," *Trans. Communication ACM*, Vol.48, No.10, pp. 83-90, 2005.

[2] Y. Maeda and M. Toyama, "ACTIVIEW: Adaptive data presentation using SuperSQL," *Proc. VLDB*, pp. 695-696, 2001.

[3] T. Bickmore, A. Girgensohn, and J. Sullivan, "Web Page Filtering and Re-Authoring for Mobile Users," *The Computer journal*, Vol.42, No.6, pp. 534-546, 1999.

[4] O. Buyukkokten, O. Kaljuvee, H. Garcia-Molina, A. Paepcke and T. Winograd, "Efficient Web Browsing on Handheld devices Using Page and Form Summarization," *ACM Trans. Inf. Syst.*, Vol.20, No.1, pp. 82-115, 2002.

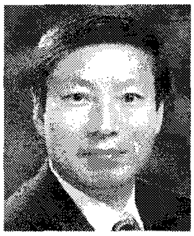
[5] O. Buyukkokten, H. Garcia-Molina and A. Paepcke, "Accordion Summarization for End-Game Browsing on PDAs and Cellular Phones," *SIGCHI '01*, Vol.3, No.1, pp. 213-220, 2001.

[6] M. Toyama, "SupserSQL:An Extended SQL for Database Publishing and Presentation," *Proc. ACM SIGMOD*, pp. 584-586, 1998.

[7] T. Seto, T. Nagafuji, M. Toyama, "Generating HTML Sources with TFE Enhanced SQL," *ACM Symposium on Applied Computing*, pp. 96-100, 1997.

Sang-gyu Shin

Sang gyu Shin received the MSc degree in computer science from the University of Keio, Japan, in 2003. Currently, he is a PhD student in the School of Science for Open and Environmental Systems, the University of Keio. His research interests include database publishing, XML query processing, data mining and information retrieval.



Tai-suk Kim

received the B.S. degree in Electronic Engineering form Kyungpook National University, Korea, in 1981 and the M.S. and ph.D. degree in Computer Science from KEIO University, Japan, in 1989 and 1993, respectively. Since 1994, he has been a faculty member of the Dongeui University, where he is now Professor in department of Computer software engineering. His reserch field has been in information system, internet business, network game and NLP.