Annales UMCS Informatica AI 4 (2006) 189-197

Annales UMCS
Informatica
Lublin-Polonia
Sectio AI

http://www.annales.umcs.lublin.pl/

Comparative analysis of reporting mechanisms based on XML technology

Dariusz Król[1*], Jacek Oleksy[2], Małgorzata Podyma[2], Bogdan Trawiński[1]

[1]*Institute of Computer Science, Wrocław University of Technology,*
Wybrzeże S. Wyspiańskiego 27, 50-370 Wrocław, Poland
[2]*Computer Association of Information BOGART Ltd.,*
Rejtana 9-11, 50-015 Wrocław, Poland

Abstract

Comparative analysis of reporting mechanisms based on XML technology is presented in the paper. The analysis was carried out as the part of the process of selecting and implementing of reporting mechanisms for a cadastre information system. The reports were designed for two versions of the system, i.e. for the internet system based on PHP technology and the fat client system in two-layer client-server architecture. The reports for the internet system were prepared using XSLT for HTML output and using XML-FO for PDF output and compared with reports implemented using Free PDF library. Each solution was tested by means of the Web Application Stress Tool in order to determine what limits in scalability and efficiency could be observed. As far as the desktop system is concerned three versions of reporting mechanisms based on Crystal Reports, Microsoft Reporting Services and XML technology were accomplished and compared with the mean execution time as the main criterion.

1. Introduction

On the market there are many commercial reporting tools. They can take data from various sources such as ODBC, JDBC, EJB, OLE DB, Oracle, DB2, Microsoft SQL Server, Access, BDE, XML, txt and others. Almost all of them allow to export reports to such file formats as html, pdf, xml, rtf, xls, csv, txt and some to doc, LaTeX, tiff, jpeg, gif [1,2]. The study presented in the paper was carried out in order to choose the most suitable reporting mechanisms for a new version of a cadastre system used by many local governments in Poland. Selection of a reporting tool for an information system is a difficult task. It should be accomplished carefully and many factors should be taken into account such as user expectations, ability of report designing and modifying, possibility of presenting complex data in a readable way, costs and licensing as well as the performance and scalability required. However, just the prices usually have

[*]Corresponding author: *e-mail address*: dariusz.krol@pwr.wroc.pl

decisive influence on the selection of reporting tools for the cadastre system. Commercial tools are perceived by local governments to be too expensive and this was the main reason that we started to design and programme reporting mechanisms based on low level open source tools.

The selection process should also take into account the results of feature comparative analyses [3,2] and benchmark tests carried out on reporting tools by their producers or by independent organizations [4-7]. All benchmark tests have proved that such commercial tools as Actuate iServer, Crystal Enterprise and Cognos ReportNet significantly exceed the requirements of the cadastre system considered in the paper. Those benchmark tests were accomplished using highly efficient and expensive hardware and were designed for the developers who want to build large-scale applications. Therefore the results were not usable for the decision process of selecting the most convenient reporting tool for the cadastre system. This also led us to start testing the performance and scalability of our solution.

The most prospective seem to be the mechanisms based on XML language. Using the DOM (Document Object Model) you can create and manipulate XML documents by your application with dynamic access to the content of XML documents. XSL and XSL-FO languages enable you to create style sheet documents which control the presentation of a XML document and describe the transformation of a XML document into other formats such as HTML, SVG, CSV, plain text or PDF [8-10].

The model of generating reports using XML language is shown in Fig. 1. XSLT processor takes XML and XSLT documents and transforms them into new XML document or into a document in another format for example into HTML document. In order to obtain PDF output XSL-FO document is created and then transformed by FOP processor into final PDF document.

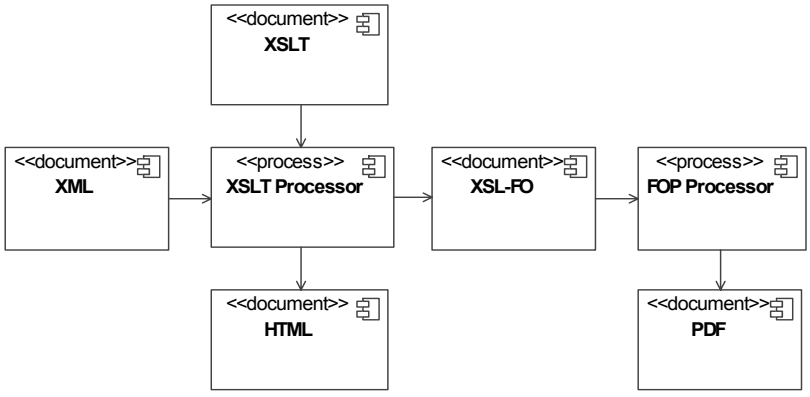

Fig. 1. Model of creating reports using XML language

2. Reports in the cadastre system

The maintenance of real estate cadastre registers is dispersed in Poland. There are above 400 information centres located by district local self-governments as well as by the municipalities of bigger towns which exploit different cadastre systems. The majority of cadastre systems in Poland are developed in two-layer client-server architecture and deployed on Microsoft SQL Server or Oracle database management systems running on Intel servers. The EGB2000 system for which we were looking for the best reporting tools is deployed in above 100 local governments throughout Poland while the EGB2000-INT system providing an internet access to cadastral databases is used in about 50 intranets and extranets. The EGB2000 system was implemented in fat client architecture using Visual C++ in Visual Studio.NET environment whereas the EGB2000-INT system was programmed using PHP script language and accommodated for the cooperation with Apache or IIS web servers.

Four years of developing and maintaining cadastre systems in many information centres in Poland experienced us that users regard reporting functions as the most important in the systems and require reporting tools of high quality. They expect reports presenting complex data clearly, allowing to select data using various criteria, enabling to change the scope of data shown in a flexible way and taking time as shortest as possible to generate.

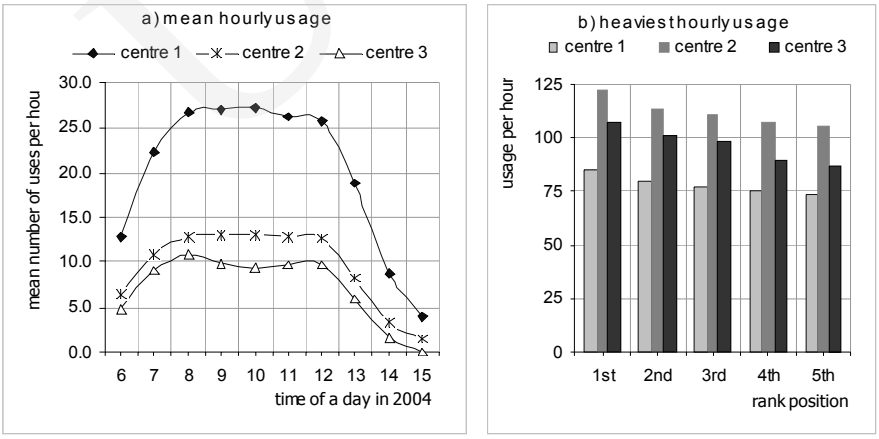

Fig. 2. Hourly usage of reports in the internet system in selected centres

There are 117 reports available in the fat client system and 15 in the internet one. In order to investigate what and how the reports are used in both systems report usage logs are maintained. The analysis of these logs in six selected centres indicated that three groups of reports were the most frequently employed by the users of the systems. These are extracts from registers, notifications of changes and various lists of objects with their attributes. Fig. 2a presents mean

hourly usage of reports in the internet system in 2004. The five best results from the top for each centre studied are shown in Fig. 2b. The data indicate that the maximum hourly usage reached 122 which equals to only 2 reports per one minute. The mean hourly usage of the reports in the fat client system during June 2005 for three other centres is shown in Fig. 3a. Analogously to Fig. 2b the top five results are placed in Fig. 3b. The data show that the maximum hourly usage reached 150 which equals to only 2.5 per minute. The load of both systems was rather moderate.

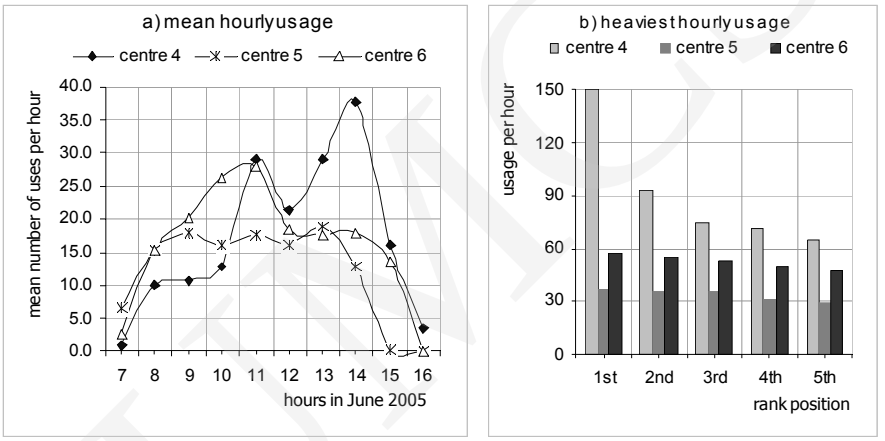

Fig. 3. Hourly usage of reports in the fat client system in selected centres

3. Testing of reporting mechanisms implemented in the internet cadastre system

Three versions of reporting mechanisms implemented in the internet system were compared, i. e. one based on XML technology with HTML output (HTML), the second employing FOP processor to generate reports in PDF format (FOP) and the third programmed using Free PDF library to produce PDF output (FPDF). Each solution was tested using Web Application Stress Tool in order to determine what limits in scalability and efficiency could be observed. The most frequently used report i.e. an extract from the land register unit was the main subject of all tests.

Architectures of the first two reporting mechanisms are presented in Figs. 4 and 5. In both cases the system applications create XML documents using DOM and selecting records from database according to the search criteria input by users. Then the required output is generated by XSLT processor or by FOP.

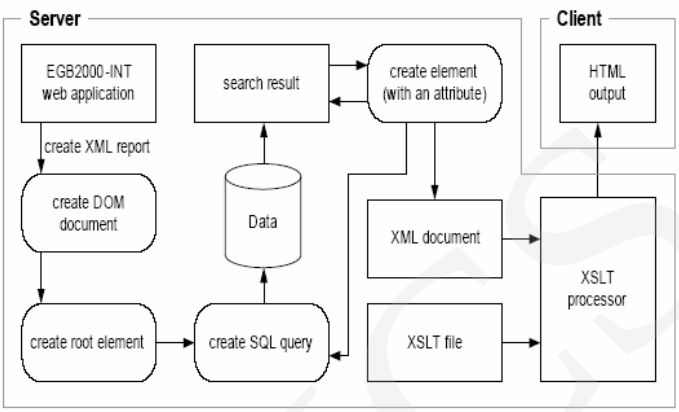

Fig. 4. Architecture of XML reporting mechanism for the internet system with HTML output

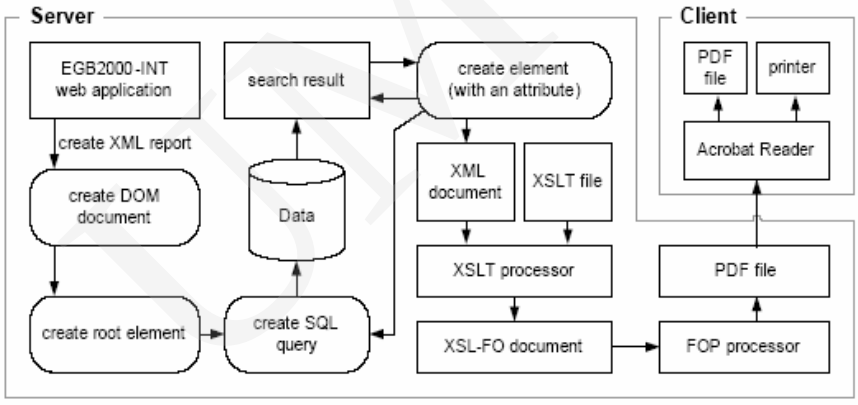

Fig. 5. Architecture of XML reporting mechanism for the internet system using FOP

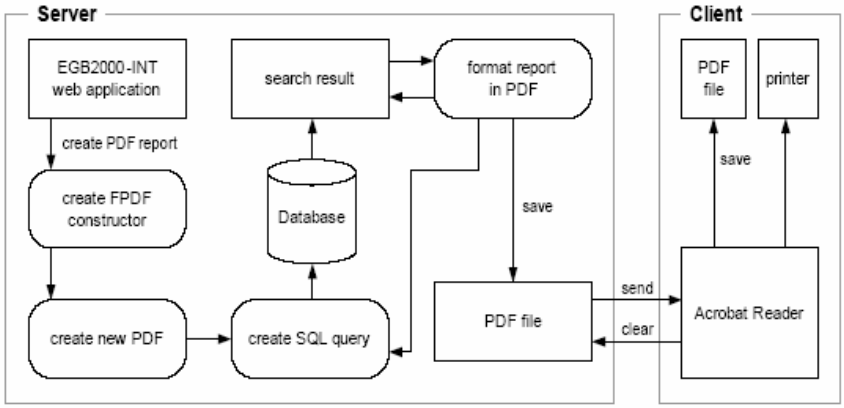

Fig. 6. Architecture of reporting mechanism using Free PDF Library

In Fig. 6 the architecture of the third reporting mechanism is shown. The Free PDF library generates a PDF file directly on the basis of the result of SQL query and then saves it on the server disk and sends it to the client's browser. When the client finishes working with the report the PDF file is deleted from the server disk.

3.1. Testing of mechanisms based on XML providing HTML and PDF output

The experiment was carried out using the server with 2.0 GHz Intel Pentium IV and 1 GB RAM, XP Windows Professional operating system, Apache web server and MS SQL Server 2000 as DBMS. The tests simulated continuous work of 1, 2, 5 or 10 concurrent users who generated reports containing 1, 10, 100 or 500 parcels. The results are presented in Fig. 7 where -10 and -100 denote the numbers of parcels comprised in respective reports. It can be easily seen that the mechanism using only XSLT processor gives better results than the mechanism having additional FOP processor. In all tests for one user the mean response time was shorter than 1 s and for 10 users smaller than 1.6 s. However 15 users or reports with 500 parcels caused the load too big for the operating system. It started to suspend or to display out of memory error.

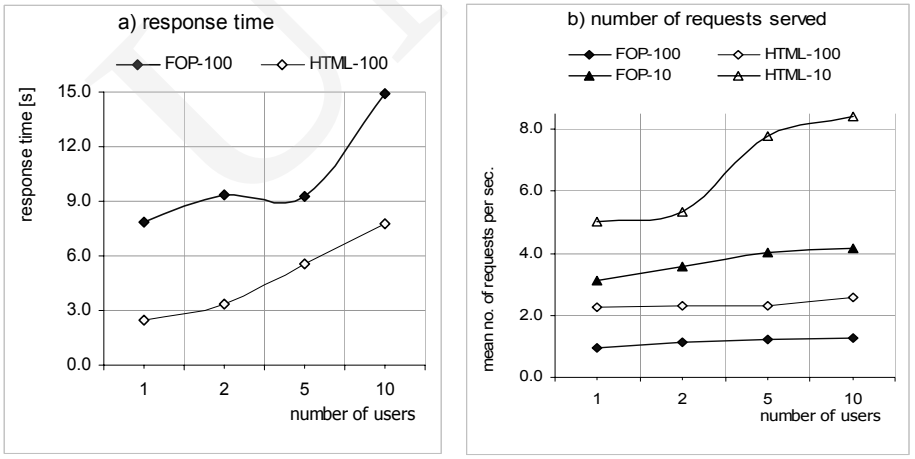

Fig. 7. Test results of XML reporting mechanism with HTML output and using FOP

3.2. Testing of mechanisms using FOP and free PDF library

In the experiment 2.0 GHz Intel Pentium IV with 1 GB RAM was used as the server. Windows 2000 Professional operating system (different from OS used in sec. 3.1), Apache web server and MS SQL Server 2000 as DBMS were installed.

Analogously the tests simulated continuous work of 1, 2, 5 or 10 concurrent users who generated reports containing 1, 10, 100 or 500 parcels. Denotation of mechanisms in Fig. 8 is similar to that used in Fig. 7. The results in Fig. 8

revealed better performance of the mechanism using Free PDF library, but the difference was not big enough to compensate longer time and inconvenience of programming. The tests for 15 users or reports with 500 parcels also caused so heavy load that the operating system did not manage to serve it.

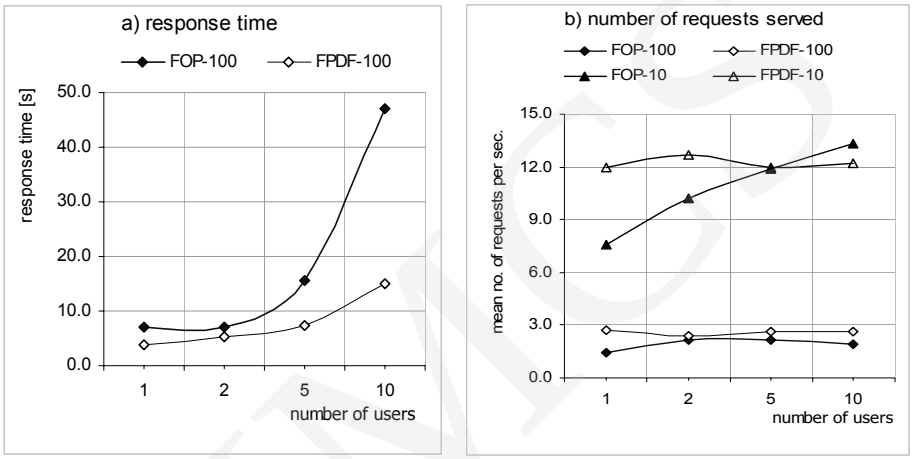

Fig. 8. Test results of reporting mechanisms based on FOP and Free PDF library

4. Testing of reporting mechanisms implemented in the fat client cadastre system

In order to study the XML technology (XML) in the fat client cadastre system two additional versions of reporting mechanisms were implemented, i.e. one using Crystal Reports 10 (CR10) and the second based on Microsoft Reporting Services (MRS). In Fig. 9 the architecture of reporting mechanism using XML technology implemented in the fat client cadastre system is presented.

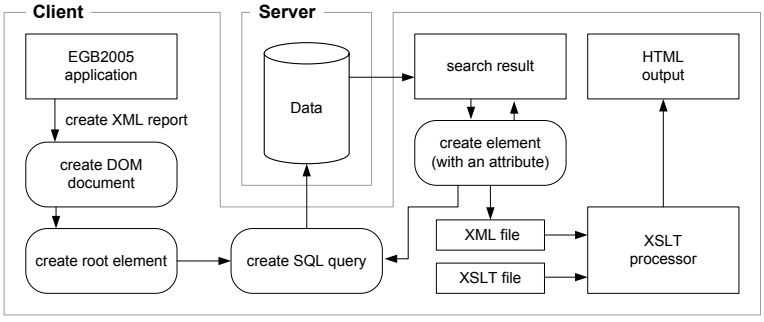

Fig. 9. Architecture of reporting mechanism in the fat client cadastre system

Each solution was tested with the most frequently used report i.e. an extract from the land register unit. The tests simulated continuous work of 1, 2, 5 or 10

concurrent users who generated 20 the same reports containing 1, 10, 100 or 500 parcels. The mean time of generating individual reports was calculated. The MRS mechanism gave the best results (see Fig. 10) and the XML mechanism showed better performance than the CR10 one. Perhaps it was caused by the method of designing CR10 reports, which contained SQL expressions in rpt files.

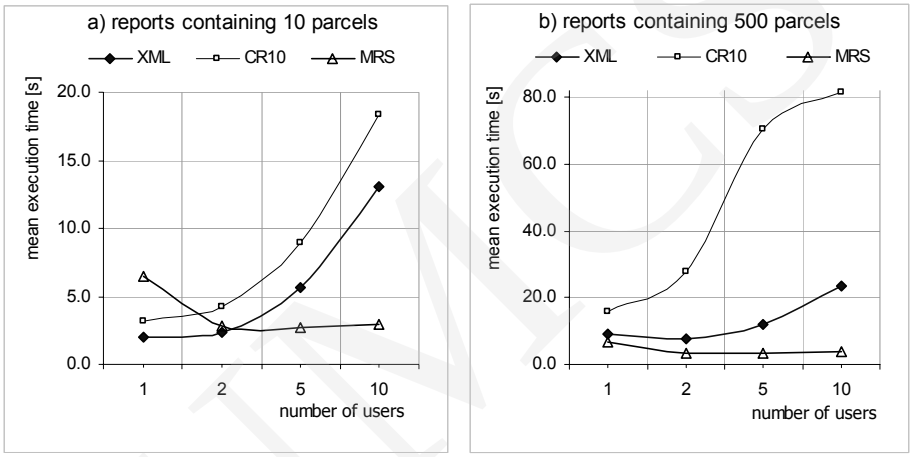

Fig. 10. Test results of reporting mechanisms in the fat client cadastre system

5. Conclusions

All the tests were carried out using the configuration of hardware and software common for majority of information centres. They proved that all reporting mechanisms tested could fulfil the present system requirements of performance. Although the mechanism based on Microsoft Reporting Services revealed the best performance, it should be treated cautiously. It is relatively new and not all features and behaviour are known yet. Moreover, we do not suppose that our clients having Oracle databases would maintain Microsoft SQL Server as their second database environment. In turn, Crystal Reports is limited to one level of sub-reports which considerably makes it more difficult to design reports selecting complex data. The most prospective seems to be XML technology which has no drawbacks of other mechanisms tested taking into account all present aspects of exploiting the cadastre information system in local governments. We appreciated a wide scope of XML language functions and especially the ability to separate the content processing form presentation layer and the dynamic access to the content of a XML document. The XML technology turned out to be especially useful for the internet version of the cadastre system as far as both technical and economical factors are concerned. In fact, the tests revealed the limits of XML based reporting tools, but those limits are at present far beyond the real usage of the system.

References

[1] Król D., Podyma M., Trawiński B., S*election and testing of reporting tools for an Internet Cadastre Information System*. Software Engineering: Evolution and Emerging Technologies Ed. K. Zielinski and T. Szmuc. IOS Press, (2005).
[2] Podyma M., *Review and comparative analysis of reporting tools for internet information systems*, M. Sc. Thesis (in Polish). Wrocław University of Technology, (2005).
[3] Feature comparison. Microsoft SQL Server 2000 Reporting Services vs. Business Objects Crystal Reports, Crystal Enterprise. Certia Business Intelligence Team, (2004).
[4] Actuate 7 iServer vs. Crystal Enterprise 9. An Interactive Reporting Performance Benchmark Project Final Report, National Technical Systems, (2003).
[5] Actuate 7SP1 Performance and Scalability on Windows 2003, Actuate Corporation, (2004).
[6] Cognos ReportnNet Scalability Benchmarks – Microsoft Windows, Cognos, (2004).
[7] Crystal Enterprise 9.0 Baseline Benchmark: Windows Server 2003 on IBM@server xSeries 16-way, Crystal Decisions, (2003).
[8] Holzer S.: *Inside XML,* New Riders Press, (2000).
[9] Holzer S.: *Inside XSLT.* Que, (2001).
[10] Pawson D.: *XSL-FO,* O'Reilly, (2002).