Migration of Legacy Systems to Web Applications – A Novel Approach

Khansa Aatif, Saima Farhan

Deapartment of Computer Science, Lahore College for Women University, Lahore, Pakistan
khansakamran@hotmail.com, saifar79@hotmail.com

Abstract*:* Migration of legacy systems to the web applications is the need in the future of software industry. The main idea is to develop a web application either by replacing the existing legacy system or only connecting to legacy system/components. Legacy applications are migrating towards Service-Oriented Architecture (SOA) and rich internet applications. Legacy systems are based on old technology and methods, running on old platform that continues to be used, because it performs important, day to day business functions of the company. Our research would be focused on the analysis of two widely used approaches, Wrapping and Re-engineering and a method is presented which is the integration of these two approaches. Firstly legacy systems will be decomposed into components then both reengineering and wrapping techniques are applied on these components. The scope of this paper also covers the benefits accomplished from our approach. The main objective of our proposed technique is to minimize the problems involved using any one of both methods. We have applied our technique on a real life legacy system which provides an assessment of its practicality and pragmatism.

[Khansa Aatif, Saima Farhan. **Migration of Legacy Systems to Web Applications – A Novel Approach.** *J Am Sci* 2013;9(11s):1-7]. (ISSN: 1545-1003). http://www.jofamericanscience.org. 1

Keywords: Legacy Systems; Wrapping; Reengineering; Migration; Internet migration; Web integration

1. Introduction

The rapid advancement and development in information and communication technology is leading to modernization of commercial and government sectors. Since the World Wide Web came into existence, our lives and work has become easier. It has become a universal user interface for business applications, information systems, databases and legacy systems.

A legacy system is based on old methods, technology that is still in use, typically because it still functions well according to users' needs, even though newer technology and methods are also available now. A legacy system is, "any information system that significantly resists modification and evolution" (Brodie and Stonebaker, 1995). Legacy systems are important assets for their companies. Companies crucial information and business processes logic is encoded in it. These systems perform day to day business of companies and if they stop working due to any reason they may cause to stop companies business. Mostly these type of systems are running on mini or mainframe computers, developed since 1970s. They are written in 3GL or 4GL programing languages. These systems use either simple data implementation such as indexed file systems or old database systems like Hierarchical or Network. Legacy systems are used widely. According to an estimate 60% of German business run on legacy systems. "These systems usually run on obsolete hardware which is slow and expensive to maintain" (Bisbal, et al., 1999). Cost of maintaining these systems may eventually outweigh the cost of redeveloping the whole system. System documentation is often not available or not enough to understand the system. The employees who were experts on it have retired or forgotten. To identify defects/faults in them is expensive and time consuming. Legacy systems do not fulfill current demands from IT users that are flexibility, interoperability, security, performance, communication and reusability.

Demerits of legacy systems have increased as compared to their value. As companies depend on legacy systems for running their daily operations, it is not possible to just shut these systems down for even small time period but it is necessary to evolve them. There is a current trend in the industry to migrate its legacy systems to Rich Internet Applications (RIAs) and service-oriented systems. Reengineering to the web often means to create a web application either by replacing the existing legacy system or only connecting to legacy system/components. Evolution of Web Systems, ranging from migration towards SOA (service-oriented architecture) to more classic re-engineering and maintenance tasks (e.g., improving the navigational structure). Recent studies show that 80% of the actual IT-production is carried out by legacy systems (Warren, 1999). For this purpose literature presents two approaches; i.e. Wrapping and Re-engineering.

A. Wrapping

In wrapping, a legacy software system/component is wrapped within a XML or HTML shell. So that it can run on web server. Other software components/user/system can access legacy software through that wrapper. That is, interact with existing system without changing its internal architecture. The components wrapped can be a batch process, online transaction, a program, a module or just a simple block of code. Normally automated tools are used to analyze code in wrapping. Legacy code is only wrapped in an XML shell which allows individual functions in program to be offered as web services (Sneed & Wien, 2006) or decorates the input and output of systems with HTML Markup (Zdun, 2002). It is a black box modernization technique which concentrates on the interfaces of the legacy system (Ricca & Chao, 2009).

Wrapping is a quick and cost effective solution. Wrapper also changes and improves a system's outlook. In Wrapping we use the same components of legacy system which are already tested and trusted by their company. In spite of benefits wrapping can cause host organization many problems. Such as wrapping is a short term solution. It complicates system's maintenance and management. Additional software layer is used in wrapping which increases architecture overhead and increases page overload. Wrapping do not solve problems already present in maintenance and upgrading. There is an overhead of script interpretation and database access at the page load time. Performance and structure is not as better as can be achieved in redevelopment.

B. Reengineering

Reengineering means redevelop a system completely to make it purely a web-enabled solution. It is rewriting the application to transform the existing systems to new system completely. A system is redeveloped using modern technologies, database and hardware. That redeveloped system is then re-implemented. Reengineering is also known as Big Bang and Cold Turkey. Reengineering process demands major changes to code. It delivers a complete solution to our problem. No added software is needed as in wrapping, so re-engineered software provides comparatively fast solution. No added software means, no performance degradation. But here are some drawbacks too. When we redevelop a system from scratch risk of system failure and security breaches increases too. Rewriting the application is expensive and time consuming. More testing time is required while using this approach. Reengineering is also not a long term solution as business and IT industry requirements are continuously changing. After a long process of reengineering, companies may find themselves again with an out dated technology. Legacy system may completely shut down or cut over during the process of reengineering.

In spite of all this, legacy applications have some other issues. Legacy databases do not provide security. Legacy systems contain crucial business information. Security is the primary concern of the companies who want to publish their business online. A web database allows you to access and update information from anywhere in the world. When confidential data is published on web, there are two major security issues.

1. Secure data transmission: Data should be safe on its way from web browser to web server.
2. Secure data storage and access: Data should be stored in database in such a way that only people with proper authority can access it (He, et al., 2000).

Legacy information systems are huge and have large databases. Once system is migrated to web, now many users will simultaneously use the application. A fast processing database is needed to fulfill this requirement.

Legacy databases like Hierarchical, Indexed sequential and Network are not a good data storage choice in a fast-paced and growing business environment. Most of the legacy systems can only be accessed by a single user at a time, which slows down work processes. Legacy databases store redundant data. Such database occupies more than required space. More processing time required if there is a lot of redundant data.

The issues inspire us to introduce a model which focuses on minimization of the inherited problems of these approaches and further strengthening their advantages. This paper presents the real life experience of applying an integration technique of wrapping and reengineering on a legacy system and describes the results and benefits achieved from it.

2. Related Work

In 1999, Sneed discussed the key risks of re-engineering projects and ways of minimizing these risks. For this purpose thirteen projects were examined. Main risks found were performance loss, architectural mismatch, required testing effort and unfulfilled quality goals. Importance of legacy information systems for an organization and reasons for transforming LIS to new target system are presented in (Bisbal, et al., 1999). The paper briefly describes the methods for this transformation and divides these methods into three types: (1) Redevelopment (2) Wrapping & (3) Migration. Different approaches used for moving legacy systems to Service Oriented Architecture are provided in (Almonaies, et al., 2010). The paper divides these

approaches in four types, Replacement, Wrapping, Reengineering and Migration. (Sneed & Wien, 2006) presents a method for reusing legacy software in service oriented architecture. Legacy code is wrapped with XML based shell. Reusability of legacy software in another environment depends on the programming language used. In 2008, Parsa and Ghods has presented a technique for wrapping legacy systems into web services and proposes a tool which creates web based source code for legacy functionality. An object oriented model of legacy code is developed. Visual studio .NET automation model tool helps in wrapping legacy program into web services. (Bisbal, et al., 1999) describes three solutions to legacy system issues, wrapping, reengineering and migration. According to the author, migration means change of system's platform. Paper describes migration as a better approach than wrapping and reengineering. (Aversano, et al., 2001) proposes application of Re-engineering and wrapping techniques both on the same legacy system as a better strategy. User interface is reengineered while rest of the system is wrapped.

3. Comparison of Wrapping and Reengineering

There are some scenarios shown in Table 1, where an integration technique may not provide a better solution and either wrapping or reengineering can provide a good solution.

Table 1. Wrapping vs. Reengineering

Sr.	Scenarios	Wrapping	Re-Engineering
1	Where there is no need to change business functionality.	✓	
2	No access to source code, it is lost, developed by third party, no license to use it.		✓
3	Required to integrate the interaction layer of existing application or with in a newly developed system, while it is also required to keep the original application running with no alteration. This can happen due to mergers or acquisitions.	✓	
4	It is required to open system for access via different platform e.g. web, mobile device or SOA.	✓	
5	If main weakness is only interaction layer.	✓	
6	Legacy components are well tested and trusted.	✓	
7	Core functionality is buried deep in existing software, i.e. code is of high business value.	✓	
8	Program is complex enough, re-writing them is no simple task.	✓	
9	Tight time constraints, no choice but to re-use existing software.	✓	
10	Where code quality is not good and it has to be improved.		✓
11	Changes in code are risky.	✓	
12	Changes in code are too costly to afford.	✓	
14	Legacy systems do not function properly.		✓
15	If design pattern of legacy code does not support wrapping.		✓

4. Proposed Approach

Proposed model is the integration of both techniques; wrapping and reengineering. It is presented with the aim of minimizing the problems involved in using one of these techniques.

We have focused on analysis of both approaches and their impact when applying to different legacy systems. Before discussing our proposed technique, it is important to understand the decomposition of legacy systems. Research shows that legacy systems are decomposed into 3 layers (Brodie and Stonebaker, 1995) (Goeschka and Schranz, 2001).

Figure 1 shows the decomposition of legacy systems.

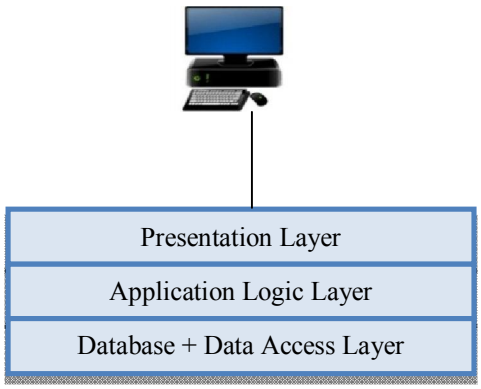

Figure 1. Decomposition of Legacy System

Presentation Layer is that part of system which interacts with other systems/users. It is the outer layer of system. Organization's business logic is encoded in application logic layer. This layer contains the application code in which flow of business Processes are programmed. The 3rd layer is of database which stores the data and methods to retrieve and store data in database; i.e. Database access methods. For example in Network, Hierarchical and Relational databases, queries are used to access data.

Now, after decomposition, if application logic is a small layer of code between data access layer and presentation layer then reengineering of whole system.

If application logic layer is not a small layer of code or system is data intensive then:

1. Reengineering of user interface layer.
2. Wrapping of application logic code.
3. Reengineering of database and data access layer.

Reengineering of user interface can be done using ASP/PHP/Html technologies. MORPH approach can be followed too (Moore and Rugaber, 1997), which can be used to reengineer graphical user interface from one platform to another. Auto tools are available for wrapping. Made Wrapper products are also available for different types of legacy systems. Popular made wrapper products are SoftWrap, ObjectStar and ObjectBroker. Companies can develop wrapper appropriate for their legacy applications.

Reengineering of legacy database is recommended into modern database, for example relational database. Auto tools are available for this purpose. Figure 2 shows some of legacy databases which can be migrated to modern relational databases. Reengineering is also possible without tools.

Legacy Databases	Relational Databases
Indexed Sequential Flat File Network Hierarchical	Oracle Microsoft access My SQL SQL server

Figure 2. Legacy Database Migration

Properties of our technique:

It involved both approaches; wrapping and Redevelopment, Partial Redevelopment/Reengineering and Partial Wrapping.

In order to avoid issues involve in Wrapping and Reengineering, we do not recommend use of one of these techniques. More over the situations where any one of these techniques can provide the better solution are discussed in previous chapter.

We have proposed partial wrapping and Re-engineering. A system will be decomposed first in components and both techniques will be applied on same legacy system but on different components.

As discussed before that reengineering is more time taking and costly solution and there are more risks involved in reengineering. These risks can lead a system to its failure. The largest part in large information and transaction oriented systems is there application logic part. According to Sneed, 1995 and 2000 cost of reengineering of logic code is equal to cost of re-implementing the whole system. Our technique proposes that application logic part should be wrapped. It saves time and money. Moreover wrapping of business code layer decreases the amount of risks involved in reengineering. Rewriting of code consume most of the testing activity time too. "Risks involved in reengineering of existing code are too high" (Aversano, et al., 2001). Automatic wrapping tool are available which can wrap code very quickly.

Legacy systems use old databases like Hierarchical, Network and Indexed Sequential, Flat file systems. These databases are not secure and efficient. Once a legacy system is made available through web, it is accessible to everyone and its security has become an important issue. As no of people using the system simultaneously increased after migrating the application to the web. A fast processing database is needed. Legacy databases decreases the performance of online Legacy systems. Placing a million records in online legacy system will slow down the opening and closing of the file, Most of the systems can only be accessed by a single user at a time, which slows down work processes. Our technique proposes database reengineering of legacy databases to modern databases e.g. relational database management system. RDBMS is secure and efficient.

Figure 3 shows the resultant web application architecture. In which redeveloped user interface and redeveloped database will communicate with application logic part through a wrapper.

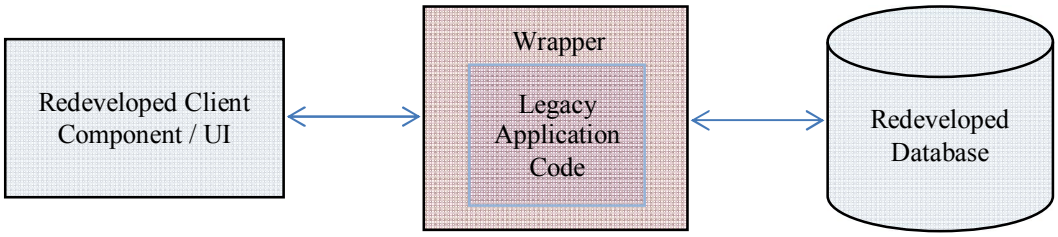

Figure 3. Web Application Architecture

Figure 4 represents our proposed mo

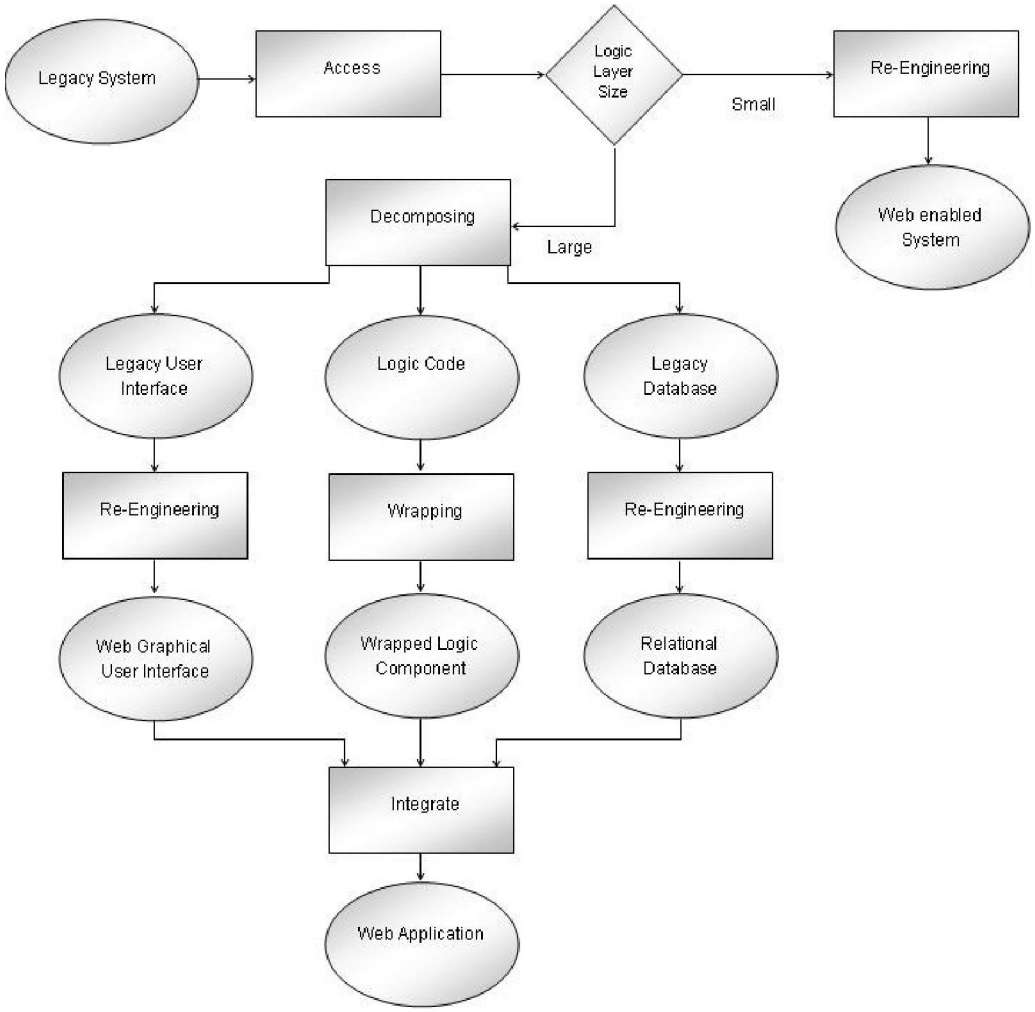

Figure 4. Flow Chart of Proposed Technique

A. Benefits of Proposed Technique

It is less costly and less time consuming as compare to reengineering approach. If a legacy system is integrated to web through complete reengineering of the existing code, the costs would be close to that of re-implementing the system from scratch using modern technologies and languages (Sneed, 1999) (Sneed, 2000). "Wrapping led to 84% lower project costs, cycle time reduction of 70%, and reduced defects" (Jalendar, et al., 2012). System performance would be better with the use of database modernization as compare to when only wrapping technique is used. When a project is reengineered

major errors which degrade its performance are (Sneed, 1999):

1. Distorted string displacements
2. Conflicting logical conditions
3. Parameter list inconsistencies
4. Diverted control flow
5. Different rounding precision

Wrapping of code + reengineering database layer only, reduces the existence of these errors.

RDBMS provide a secure environment for storing and processing huge amount of business data. A relational database management system provides the facility to create confidential tables. Companies' confidential business information will remain secure even though their legacy systems can be accessed through web. While using RDBMS only certain users have access to certain tables. For example, in an employee database, some users may be able to view salaries while others may view only work history and medical data. Important data can be made encrypted in database.

RDB is space efficient. It's processing and access time is fast. Many users can access web application simultaneously. No redundancy.
Reengineering of databases to Relational database is a long term solution of legacy migration. Legacy databases are not scalable. That's why they are not a good choice as web applications database. Future enhancement is not possible in them.

Errors per line of code are more likely to occur in case of rewriting the code so required testing effort would be less while using our approach. Our approach is less complex as compare to wrapping approach only. Because of using modern database companies can expand and integrate their database.

5. Implementation

To show the feasibility of our technique we have applied it on a legacy system.

'ROSES' is a small database program which contains details about flowers. It is made using COBOL language and Index sequential files are used to store data. Its User interface was graphical. Roses give brief details about roses flower, like their names, type, color etc. Legacy user interface shown in figure 5(a).

Legacy system provides the facility to

- Add new record
- Update existing record
- Delete record
- Find a record
- View next and previous record

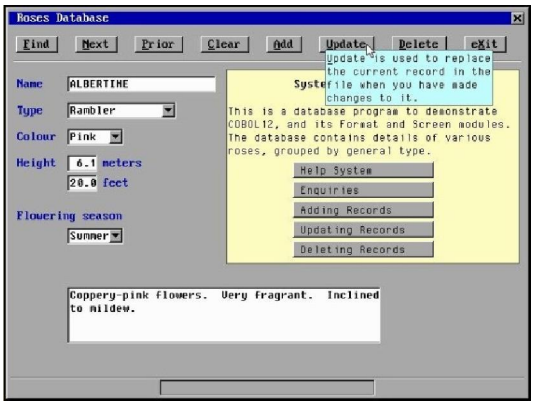

Figure 5(a). Legacy User Interface

Index Sequential database stores the whole data of system. The database was originally developed for mainframe computers. In indexed sequential file organization, the records are stored in sequence according to a primary key and an index is created to allow random access of the file.

For the purpose of migrating the application, Legacy system is decomposed in 3 layers:

1. User Interface layer
2. Logic layer
3. Database + Data access layer.

The User Interface layer and Database + Data access layer is reengineered while Logic code layer is wrapped.

User Interface is reengineered into a web browser shell using PHP, JavaScript and HTML. Cascading Style Sheets (CSS) are used to improve look and formatting. There are 3 user interface windows designed against single legacy UI to make web application more comprehensive for users. One of the user interfaces is shown in figure 5(b).

Figure 5(b). A User Interface of Developed Web Application

Application logic layer is wrapped in XML shell. Data flow analysis is necessary to identify the parameters exchanged by the user interface and the server program. Functions are extracted from COBOL code through analysis and their output is wrapped in XML tags. Java scripting code is added on user interfaces to dynamically load required information from XML file. When a particular button is pressed, java script receives that call and invokes required functionality.

Database is reengineered after analyzing legacy data structure. New database is relational database MySQL. Database access layer is integrated with user interface layer. SQL Queries to retrieve and store data in database are written within HTML code. That HTML code is used to design Presentation layer of system. And Wrapped functionality is invoked on call through JavaScript. These functions are dynamically loaded when needed.

6. Discussion

Less effort is involved as compared to redevelopment of whole application. Wrapping of code saves our time and cost. No Performance degradation is viewed in newly developed web application. It provides all services of legacy application efficiently.

Database reengineering has overcome the loss of efficiency which could be the result of additional overhead of adding XML code. Database reengineering makes it possible to expand or integrate our application in future.

Database reengineering makes the system more efficient and secure. Wrapping reduces defects per line of code, hence reduces testing efforts.

7. Conclusion/Future Work

In this paper with the help of an experimental case study we have proved that our technique is beneficial. It solves many issues. Like saves cost and time by adding partial wrapping in migration. Where complete reengineering leads to more expenses and time, wrapping of logic layer decreases that cost and time consumption prominently. Database reengineering increases the system's efficiency.

There is still a need for the development of objective metrics to evaluate approaches, Wrapping and reengineering and select suitable technique for a particular legacy system. One of the biggest challenges in migration of legacy systems is to identify candidates for web services in the functionality of legacy code. Better tools and techniques should be prepared for this purpose. These tools can extract business functionality out of legacy code for developing web services more efficiently.

References

1. Sneed H M. Risks involved in re-engineering projects. In: Proceedings of 6th Working Conference on Reverse Engineering. Atlanta. October 1999.
2. Bisbal J, Lawless D, Wu B, Grimson J. Legacy Information Systems: Issues & Directions. Journal Software, IEEE. Sep/Oct 1999;16(5):103-111.
3. Almonaies A A, Cordy J R, Dean T R. Legacy System Evolution towards Service-Oriented Architecture. 2010.
4. Sneed H M, Wien A G. Wrapping Legacy Software for Reuse in a SOA. 2006.
5. Parsa S, Ghods L. A New approach to Wrap Legacy Programs into Web Services. IN: Proceedings of 11th international conference on Computer and Information Technology. IEEE 2008: 442 – 447.
6. Bisbal J, Lawless D, Wu B, Grimson J. Legacy Information System Migration: A brief Review of Problems, Solutions, & Research Issues. 1999.
7. Aversano L, Canfora G, Cimitile A, Lucia A D. Migrating Legacy Systems to the Web: an Experience Report. In: Fifth European Conference on Software Maintenance and Reengineering. IEEE. March 2001.
8. Sneed H M. Encapsulating Legacy software for use in Client/Server systems. In: Proceedings of the Third Working Conference on Reverse Engineering. IEEE. November 1996.
9. Jalendar B, Govardhan A, Premchand P. Designing code level reusable software components. International Journal of Software Engineering & Applications (IJSEA) January 2012; 3(1).
10. Jan W. The Renaissance of Legacy Systems: Method Support for Software-System Evolution. Lancaster University. Springer Verlag Publisher, London. 1999.
11. Zdun U. Reengineering to the Web: A reference Architecture. In: Proceedings of the Sixth European Conference on Software Maintenance and Reengineering. 2002.
12. Ricca F, Chao L. Special Section on Web Systems evolution. International Journal on Software Tools for Technology Transfer (STTT) 2009; 11(6):419-425.
13. He J, Wang M. Cryptography and Relational Database Management Systems. International Symposium on Database Reengineering and Applications. 2001.
14. Brodie M L, Stonebaker M. Migrating Legacy Systems – Gateways, Interfaces & Incremental Approach. Morgan Kaufmann Publisher, San Francisco. 1995.
15. Goeschka K. M, Schranz M W. Client and Legacy Integration in Object Oriented Web Engineering. Journal of Multimedia 2001; 8(1).
16. Moore M, Rugaber S. Using Knowledge Representation to Understand Interactive Systems. In: Proceedings of 5th international Workshop on Program Comprehension, Dearborn, MI, IEEE Computer Society Press, 1997;60--67.
17. Sneed H M. Planning the reengineering of legacy systems. IEEE Software 1995;12(1):24-34.