Reengineering of database intensive application

Rakesh Agarwal, Ajit Sarangi and Swati Das
Infosys Technologies Limited, India
Email: {rakesh_a}{ajitsarangi}{swati}@infosys.com

Abstract
Reengineering databases has been a challenge since ages and it requires process mapping to understand better and significantly improve the business processes and performance. In this paper we describe a generic architecture for reengineering legacy databases, which is an outcome of working on a real software project. The goal of this research is to formalize a process that is applicable to different database reengineering scenarios and requirements. We elaborate the steps that were actually done for implementing the project.

Keywords: Database, legacy system, application development, reengineering.

1. Introduction

Current trend of research [1][2][3] shows that most of the time spent by a software engineer is for making changes to mission-critical [4] software. Thus, software organization are impacted no longer by the development tool, it is the maintenance and re-engineering process [3][5].

Organizations migrate or replace existing software functions [6][7] rather than build them from scratch. Reducing the analysis of code time is key to improving software- productivity [8][9]. We need processes and tools to improve the maintenance of existing systems by reducing the change-request overhead and reducing the turnaround time [4][10]. Software maintenance [11][12] has been a historically problem and it faces new pressure to change systems. This pressure stems from business-process reengineering, acquisitions and mergers, the rise of multinational operations, increased competition, and new government regulations.

Re-engineering may be categorized in different forms[3][11]: Business-process reengineering deals with the business administration. It does not deal with computer process directly, but with the business process they support. Data reengineering involves restructuring of databases. The data contents basically remain the same but the form is altered. For example, a data from hierarchical database (IMS) may be ported into relational database(DB2). Data reengineering has two objectives: to improve the data's accessibility and to transfer data to a supposedly better container. Software reengineering involves in the renovation of programs and other software artifacts[13][14]. The process of data engineering[15] is similar to the programs or methods that process the data.

An organization to succeed in today's environment must focus on the customers and markets, rather than on products and production. The organization is committed to improving the flexibility, efficiency and cost effectiveness. To compete effectively in today's market environment, flexibility is the key. The market place is changing as per demand and supply. One has to change constantly as per business requirement to be leader in the market.

Regardless of how it is ultimately accomplished, process reengineering is always concerned with improvement the efficiency and the effectiveness of an overall process. Efficiency refers to the degree of economy with which the process consumes resources-especially time and money. Effectiveness is concerned with how well the process actually accomplishes its intended purpose, here again from the customer's point of view.

This paper gives the details of each of the phases for reengineering a database and elaborates the critical processes involved.

2. Re-engineering

Since the requirements keep on changing in the dynamic environment the application becomes obsolete to support all kinds of change required. This can be better explained by following example: The data stored in the database is accessed by number of users. As the number of people accessing the system increases the accessibility time increases for database. This may be because some of the fields are redundant and thus, we need to take out those fields. Another situation may be we want to add some more fields due to business process change. Also there may be a situation where we want to change the format of the fields. In these situations we have to re-engineer the database [16][17].

Re-engineering helps in integration of business plans, business processes and business information. Reengineering is required for better performance and reliability.

2.1 Types of re-engineering

Application code – Company strive to produce better products and provide better services to stay ahead in the competition. On the other hand, when the product quality is measured in terms of three or four defects per million. Over the last 10 to 15 years companies have been forced to improve their business processes because customers are demanding better and better products and services. If we do not receive what we want from one supplier, we have many others to choose from (hence the competitive issue for businesses). New technologies (like the Internet) are rapidly bringing new capabilities to businesses, thereby raising the competition and the need to improve business processes dramatically. Existing process must be related back to business plans. Only those processes that support plans (as explained in section 3.2, 3.3 and 3.4) relevant to the future should be considered for re-engineering. For this process we have to again go through the software life cycle. We have to change the design and in-turn change the code also.

Database – As explained previously only database change may cater to current requirement where we need to add/change few fields or index of the database. In this case whole system model and workflow as explained in figure 1may not be followed. Only specific stages in section 3 is required. Database re-engineering is changing the database design or changing from one type of database to another type that will provide high business quality. For example: from hierarchical database to a relational database, or, may be from Sql sever to Oracle.

Hybrid - Business process reengineering leads to change in database, and change in process/methods. That is known as hybrid reengineering, change in the code and the database. The stages to be covered are explained in section 3.1, 3.2, 3.3 and 3.4.

3. Stages of re-engineering

The system model comprising of various steps for reengineering the database is shown in Figure 1.

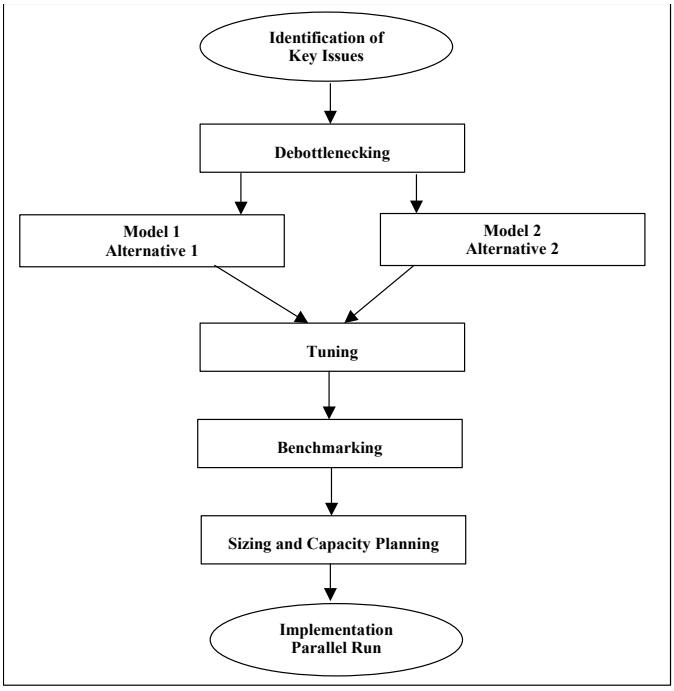

Figure 1: System flow model

3.1 Identification of key issues

Before any reengineering process we need to identify the need very specifically. If the requirement is defined meticulously the goal achievement after reengineering process will be smooth. The customer is responsible for defining what constitutes product or service value. The real winners are those who not only met the customer needs, but delights the customer.

The organization should be structured from the top down to support its value-added processes. Dramatic improvements in cycle times, process costs, and/or customer satisfaction ratings are the key indexes of the success of the most re-engineering projects.

The discussions with the people who directly support the business process were done to analyze and re-design the process. As those who own the process will likely have the best idea for improving the process, involvement is the key strategy for overcoming the resistance to change. On the other hand we need to be aware that an outside change agent may also be needed on the reengineering team to ensure objectivity. Points to be noted are:

- Senior management must be involved throughout the reengineering project.
- Reengineering seeks to optimize the performance in relation to other consideration at sub process level and task level. This will include an analysis of the distributed workload resulting from the reengineered process.

The outcome sought are:

- The current requirement are identified.
- To improve the flexibility, efficiency and cost effectiveness of current systems – when number of people increased the access time becomes very high.
- Some data is not captured in the current system – due to business process change, some more data has to be captured.
- Obsolete – There are some data/process, which are obsolete. Those things have to be taken out/changed.
- Robust – It should be capable of handling a huge number of user at a particular point of time.

3.2 Debottlenecking and Modeling

A carefully planned system of measurements is necessary to establish how well a process is performing and to compare before and after result. The process team, as well as upper management, should be involved in selecting the parameters to measure and deciding how to benchmark the performance of the process against internal and/or external standards. Care must be taken to ensure that the various indexes of process performance are internally consistent throughout.

Broadly the reengineering function can be categorized as:

- Re-engineering the business – Change in your business functionality.
- Re-engineering of database (Database migration and Database design change).

Primarily modeling is essential to verify the reengineered concept. The model should support identified parameters in previous stage (section A), for example, 500 users at a particular point of time or 300 transaction per second.

3.3 Sizing & capacity planning

Before stepping into the re-engineering of the database the infrastructure has to be ready, for example, the required hardware and software. The license of all items has to be ready. Size of reengineered database has to be estimated with current load factor along with future load. For capacity planning we need to take care of utilization factor to optimum.

3.4 Bench Marking-Tuning

To achieve the best throughput of the system we need to optimize the application. The optimization process can be better described as to set parameter at outset from standard norms of baseline data and then comparing the existing result to it. If the set point is not achieved tune the application for the desired result. Once the model is done there is a need of tuning the process.

After that is done again the test is made. Cross check for response time needs to be done with the target that has been set in the model.

3.5 Implementation

There are different phases database migrations, which are part of database re-engineering. The workflow model is shown in Figure 2.

The existing system study, extract business functionality, redesign the workflow are part of section A and B.

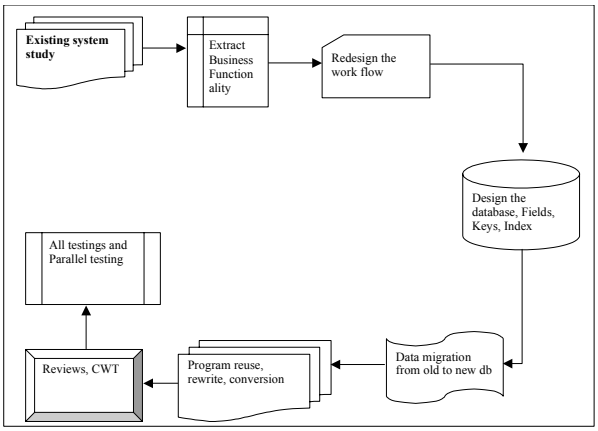

Figure 2: The workflow model

Data migration: The design of the database has to be understood properly i.e. what are the fields, indexes, data type of those fields etc. Then create the new database and map the old fields with the new fields. Then migrate from old to new. Different steps in data migration are:

- Extract – The data has been extracted from the old database by using some interface. This process may use some predefined tools to read data from the database.
- Validation- Then the data is validated with new one. For example the data type of the fields. After the data is retrieved from the database it is checked/validated.
- Input- After the data is validated that is out to the new database with the help of some interfaces. For this process also we can use some tools. For this process, we can create some new interfaces or the old interfaces can be used with some wrapper classes.

There are number of powerful, flexible and high-performance database migration and movement tools. Tool can be also used as a **data export tool**. These tools significantly facilitates the movement of **database schemas and data** between different or same database systems. Tool uses the fastest, database native tools to import or load data. Tool can export data from any database accessible through the ODBC interface.

It exports data to text files (**CSV, TAB delimited** and fixed length are supported formats) and generates native **DDL** (Data Definition Language) and import-load scripts for various databases.

There are a number of tools available for **Oracle, IBM DB2 and Microsoft SQL Server** databases but can be also very helpful for other databases such as **Sybase, Informix, MySQL, Access, FoxPro** and others.

They have a lot of features useful for DBAs:

- There is a comprehensive command prompt tool. So you can create various scripts to automate your work. The tool allows you creating scripts for regular data moving, loading into databases or Datawarehouse.
- You can provide a table name, a table template (like %.% to export all tables of a database) or any SQL SELECT statement (from the command prompt or a file) specifying the data to be exported.
- Tool generates native CREATE TABLE and CREATE INDEX scripts for Oracle, IBM DB2 and MS SQL Server.
- Tool generates native scripts for PRIMARY KEYS and REFERENTIAL CONSTRAINTS for Oracle, IBM DB2 and MS SQL Server.
- Tool generates loading scripts for specified database (control files for Oracle SQL Loader, IMPORT and LOAD scripts for IBM DB2).
- You can provide a list of columns to be excluded from converting, start row and number of rows to be exported.
- Movement of LOBs (images, binary files) is completely supported.
- Many others useful feature such setting names for output table, schema, files, file directories.

Programs change/Reuse: Due to change in business process or for better performance the existing application code needs to be changed. These changes can be categorized in three different types: (a) Rewrite; (b) Reuse; and (c) Convert.

Rewrite takes more time for developing the whole application but reuse takes the minimum time. Reuse is applicable in case of components and subroutines. In case of convert we use a part of the code, for example, embedded SQL code inside the COBOL program can be used in any 4GL. The choice of the above three types is driven by the available time and required performance. If sufficient time is available then rewrite is the best choice from performance point of view.

Reviews: Since there is change or rewrite of the program review is essential to cross verify the functionality. Once the methods have been modified or added, each module is reviewed according to the changed requirement. The code walk-through is done properly. During the code walk-through each part has to be reviewed, as the main problem will be with the data type. The values in the methods have been checked with the fields of the database.

Testing: After data migration and reengineering the most important part is testing. At each phase the system is tested according to the new requirement as well as existing functionality. Test[18] have to be carried out for verification of the systems: (a) Unit testing; (b) Module testing (c) Integration testing (d) Parallel run.
In parallel run both the new and the old system results are compared for better accuracy and updating of year to date figures of the application.

4. Problems of reengineering

Probable shortfalls while re-engineering[19] the database are:

- Testing of all the business functionality – When the new database concepts / structures are put into effect, it invariably leads to a lot of re-work in the business functionality.
- System integration testing – The penetration of the new database design may be pervasive, unless the whole system is again checked for the integrity.
- Format change- In case of data migration from hierarchical / network to relational database or from one relational database to another relational database, very fundamental difference is data type can be a big bottleneck for the smooth transition.
- Change of interfaces – Change in database design may lead to change in interfaces. The previous components may not work .
- Resources/manpower management – It is difficult to change the mindset of the manpower. In expertise regarding the new

system may lead to apprehension in adopting the technology.

- The psychological and emotional barriers to change must be accounted fro and carefully managed throughout the process.
- Communication and trust are pivotal to the the success of the reengineering project. If the people directly involved with the process feel that there is a hidden agenda or that they are being cut out of the communications loop, the project will be doomed from the start.
- Time – It is crucial factor. As re-engineering goes in phases. There is a fear of losing potential business in the mean time.
- Parallel run of both the systems- After the modification, the system tested for which both the old and new system have to run parallel. This occupies a lot of resources and manpower.

5. Conclusion

The biggest problem in the enterprises today is the question of upgrading applications based on the new process requirement. This requires the integration of application systems and databases within and across enterprises. There may be legacy databases that were developed years ago for a specific purpose and now for the changes required they need to be ported to new or different database available. The database re-engineering for process improvement involves some percent of risk factor that needs to be accounted during the design of the system. This happens because, of lack of understanding of technical and application importance of a particular field at the time of design. There should be around 5-10% of rework that needs to be accounted for.

There are some special features to be handled carefully. (a) Locking: The database-locking feature in the source and the target databases may be entirely different. This needs a detailed study. (b) Error handling: The error screens in the source system and the target system cannot be kept same in appearance, as the error data in source and the target system are entirely different. So this is a place where the user interface cannot be simulated as in the original application.

This paper has introduced the process of reengineering that provides a practical solution for effective process improvement in software maintenance. The effectiveness relates to the expressive power and re-usability [3] in context of software development perspective in an enterprise [5].

6. Disclaimer

The authors of this report gratefully acknowledge Infosys for their encouragement in the development of this research. The information contained in this document represents the views of the author(s) and the company is not liable to any party for any direct/indirect consequential damages.

7. References

[1] M. Hammer and J. Champy, “Re-engineering the Corporation: A Manifesto for Business Revolution”, Harper Business, New York, 1993.

[2] P. Aiken, A. Muntz, and R.Richards, “DOD Legacy Systems-Reverse-Engineering Data Requirements, Comm. ACM, May 1994.

[3] Harry M. Sneed, Planning the reengineering of Legacy Systems. IEEE software, pages 24-34, January 1995.

[4] R. W. Schwanke. An Intelligent Tool for Reengineering Software Modularity. In Proceedings of the 13th International Conference on Software Engineering, pages 83–92, May 1991.

[5] Bruno, G. Agarwal, R. Modeling the Enterprise Engineering Environment, IEEE Transactions on Engineering Management, 44, 1, pages 20-30, February 1997.

[6] N. Fenton, Software Metrics - A Rigorous Approach, Chapman & Hall, London, 1991.

[7] M. Brand, A. Sellink, and C. Verhoef. “ControlFlow Normalization for COBOL/CICS Legacy Systems”, In P. Nesi and F. Lehner, editors, Proceedings of the 2nd Euromicro Conference on Software Maintenance & Reengineering, pages 11–19, Los Alamitos, 1998.

[8] D. Coleman et al., “Using Metrics to Evaluate Software System Maintainability”, Computer, August 1994.

[9] G. Berns, “Assessing Software Maintainability”, Comm. ACM, pages 32-49, January 1984.

[10] E. Chikofsky et. al, “Reverse Engineering and Design Recovery - A Taxonomy”, IEEE Software, January 1990.

[11] B. Boehm, Software Engineering Economics, Prentice-Hall, Englewood Cliffs, N.J., 1983.

[12] T: Gilb, “Principles of Software Engineering Management”, Addison-Wesley, Reading, 1988.

[13] Agarwal R., Ghosh B. And Pattnaik M., "An Operational Process Modeling Approach for ERP implementation," in Proc. of the Workshop on Object-Oriented Business Process Modeling, European Conference on Object-Oriented Programming (ECOOP), Belgium, Brussels, July 1998.

[14] Agarwal R., Ghosh B., and Sarangi A., "Designing reliable software using DAF," in Proc. of the FastAbstracts and Industrial Practices, 10th IEEE International Symposium on Software Reliability Engineering, Boca Raton, Florida, November 1999.

[15] G. Verdugo, “Portfolio Analysis - Managing Software as an Asset”, Proc. Int’l Conf Software Maintenance Management, New York, pages 17-24, 1988.

[16] J. Hartman and D.J. Robson, “Revalidation During the Software Maintenance Phase”, Proc. Conf: Software Maintenance, IEEE CS Press, Los Alamitos. Calif., 1989.

[17] J.R. Horgan, S. London, and M. Lyu, “Achieving Software Quality with Testing Coverage Measures”, Computer, pages 60-69, Sept. 1994.

[18] H. Sneed, “Regression Testing of Reengineered Software”, Proc. Conf on Program Comprehension, IEEE CS Press, Los Alamitos, Calif., pages 149-155, 1993.

[19] R. Figlio, “Benefits of Reengineering”, Proc. Software Maintenance Association Conference, Software Maintenance Assoc., New York, 1989.

[20] Agarwal R., Bruno G. and Torchiano M., "Making Information Models Operational," in Proc. of the Third CAiSE/IFIP8.1 International Workshop on Evaluation of Modeling Methods in Systems Analysis and Design (EMMSAD'98), Pisa, Italy, June 1998.