AN INNOVATIVE SOFTWARE REENGINEERING TOOLS WORKSHOP— A TEST OF MARKET MATURITY AND LESSONS LEARNED

Pierre Bourque[1], Alain Abran[2]

ABSTRACT

Practitioners still feel ill at ease regarding an evaluation process for reengineering tools. To address this issue, a Québec-based special interest group in software engineering reused a case study workshop format that had been previously implemented to compare CASE products. This case study format had to be substantially modified to allow both vendors and participants to properly position the reengineering tools. For example, the workshop committee had to prepare reengineering frameworks, including a taxonomy of tools as well as legacy system selection criteria for the case study. This article also reports on the challenges met during the project as well as lessons learned: within a context of an immature reengineering market, the time has not yet come for a full real-life case study at an acceptable economic cost.

1- INTRODUCTION

From a very high-level perspective, software reengineering can be viewed as the application of software engineering methods, techniques, and tools to existing software applications. Such reengineering concepts and tools are currently being promoted as the solution to many software maintenance issues for legacy applications. However, in spite of a significant number of publications in both academic and trade journals, software reengineering remains quite unfamiliar to most practitioners. Even though there is a high-level of awareness in the industry, there is not a similar degree of understanding of the concepts and underlying technology. This leads to a serious handicap for practitioners and managers who are constantly solicited by reengineering tool vendors.

To address this issue for its membership, the Québec-based special interest group in software engineering known as Groupe génie logiciel (GGL) decided to replicate a winning technology assessment and transfer concept that they successfully implemented in 1991. This concept is based on partially controlled experiments using a common case study. For example, a case study was developed in 1991 that included a series of functional specifications for an MIS application. This case study was then forwarded to potential vendors who had six weeks to prepare a prototype using their latest CASE technology. The results of the prototypes were presented individually by each vendor at a GGL workshop. It is important to note that this type of event is not competitive and that there are no awards nor designated winners.

Although all vendors solve the same case study, their approaches are usually quite different and address different needs. However, the use of a common case study gives participants a better understanding of the fundamental differences between the various vendor solutions. The 1991 CASE vendor workshop was presented in three Canadian cities (Montréal, Toronto and Québec) and participants' response and levels of satisfaction were overwhelming.

To address similar needs in the field of reengineering tools, the GGL decided to stage a follow-up event using the same approach. The objectives of this software reengineering vendors workshop were to:

- demystify the key concepts of software reengineering for participants;
- enable vendors to demonstrate their latest tools and approaches in software reengineering;
- allow participants to compare these approaches and tools since the vendors would be using the same case study as the basis for their solution.

However, due to the relative state of confusion of trade publications on the topic of reengineering, the workshop committee had to prepare additional background material. This material was developed to permit participants to better review and assess vendor solutions.

This paper presents:

- the background material prepared for both participants and vendors. (Vendors were required to position their solution as per this background material)
- the immaturity of the market and technology that led to a postponement of the workshop until further market developments.

Section 2 presents an overview of the reengineering concepts covered by the workshop, including the reengineering definition selected by the workshop committee as well as the tools taxonomy used by both vendors and participants.

Section 3 presents additional information on both the criteria set up by the workshop committee for the selection of a real-life legacy application as the case study candidate for reengineering, as well as the proposed format for the vendors' output to the case study, such as the description of the solution, approach and tools used.

1 Pierre Bourque, National Bank of Canada, 500, Place d'Armes, 11th floor, Montréal (Québec) Canada, H2Y 2W3, Telephone: (514) 394-5000 ext. 5292, Fax: (514) 394-8340, E-mail: bourque@crim.ca.

2 Alain Abran, Université du Québec à Montréal, P.O Box 8888, Branch A , Montréal, (Québec) H3C 3P8 Telephone: (514) 987-8900, Fax: (514) 982-8477, E-mail: abran.alain@uqam.ca

Section 4 presents the challenges faced by the workshop committee in obtaining a real-life legacy system for the case study as well as response, or lack of, received from reengineering vendors. It then concludes with lessons learned.

2- REENGINEERING CONCEPTS

For the benefit of both participants and vendors, it was imperative to define precisely what was meant by software reengineering. For the purpose of this workshop, the following definition was selected (ARNO92):

The reengineering of software is any activity designed:

1) *to improve one's understanding of software, or*

2) *to improve the software itself (including documentation, etc.) usually for increased maintainability, reusability, or evolvability.*

Reengineering includes within its scope the software itself as well as all related documents and graphics.

The field of software reengineering is vast. It is therefore of the utmost importance to place each vendor solution in context. The classification system defined in the table below was therefore used. Vendors were permitted to work in any of these categories but were required to classify their solution according to this taxonomy.

Table 1 Software Reengineering Tools Taxonomy

Category name	Category description
Analysis and measurement of applications	These tools measure the complexity and/or quality of an application at the physical, logical, or conceptual level.
Analysis and validation of standards	These tools make it possible to ascertain whether certain standards have been followed in the various components of an application (naming conventions, programming standards, etc.)
Application conversion	This category of tools automatically converts an application from one programming language to another within the same technological platform, without modifying the initial functionality of the application.
Application optimization	These tools modify existing components of an application to optimize computer resource usage (CPU, I/O devices, telecommunications network...)
Applications restructuring	These tools transform an application (programs, JCL ...) to make it easier to maintain (or read) and remove the dead code without modifying its functionality.
Component extraction	This group of tools extracts certain physical components from an application to migrate them to other applications (file structures and algorithms, for example).
Design recovery	Design recovery must reproduce all the information required for the user to understand the whys and wherefores of an application. These tools can recover logical information (organic and technical specifications) and/or conceptual information (application domain, objectives, constraints, business rules, etc.)
Functional enhancements	This set of tools adds functions to existing software or modifies functions already in place.
Impact analysis	These tools evaluate the extent of the modifications needed to carry out a change request. For example, it is important to identify which programs and data structures are affected by the modification of a field in a database or to identify how much effort is required to adapt an application to new standards or to a different technological platform.
Migration between software engineering tools	This category of tools facilitates the migration of information from one software engineering tool to another, either for the purpose of integrating several tools together or for converting information stored in an outdated tool to a new one.
Re-documentation of applications	This category of tools allows the generation of documentation from an existing application. The documentation may be functional or technical, for the user or the organization.

Table 1 (continued)

Software standardization	With this set of tools it is possible to identify the applications or components of an application that comply with international standards such as ISO 9000-3.
Technological migration of applications	This category allows for the automatic migration of software from one technological platform to another.

3- CASE STUDY

While vendors from the previous CASE workshop had been requested to work from a paper-based description of requirements, the workshop committee felt that for a comparative study of reengineering tools, it would be preferable to ask vendors to work on a real-life legacy application. In order to reach a consensus on the legacy application to be chosen, a set of 12 criteria was selected and is defined in Table 2.

The selected legacy application was kindly provided by the local public transit authority. It is one of three systems involved in vehicle maintenance and is known as VEENT. More specifically, the VEENT application is used to control preventive maintenance of vehicles using kilometrage data as well as inspection and maintenance data.

VEENT has been in operation since 1963; most of the programs are written in COBOL and a few are written in Assembler or PL/1. This application runs only in batch mode. Files are either accessed directly or sequentially including the master file. This application also accesses, for validation purposes, some tables that are implemented using a commercial DBMS. Table 3 provides a technical summary of the application provided in the case study.

Table 2 *Case study selection criteria*

	Criteria	Description
1	Age	If possible, the application must be more than ten years old.
2	Batch processing	Must be present
3	Completeness	The complete system must be available (JCL, programs, datalib, copybook, test or production data if possible ...)
4	DBMS used	None if possible, or a hierarchical or network DBMS
5	File organization	Sequential or indexed
6	On-line processing	Should be present
7	Original technological platform	IBM Mainframe
8	Programming language	Cobol or mainly Cobol
9	Realism	Presently in production or has already been in production
10	Size	Between 20000 and 60000 lines of code. The size may seem small but it is essential that the case be described succinctly. Moreover the participants must be able to compare one solution to another.
11	Submitter's collaboration	For such a workshop to succeed, it is imperative that that the company who supplies the application collaborate fully and be implicated in the event.
12	Technical quality	Fair to good, since the case must present the vendor with certain challenges that are, however, not insurmountable.

Table 3 Technical overview of the VEENT application

Type of component	Number of components
COBOL programs	37
ASSEMBLER programs	6
PL/1 programs	5
Unretrievable source programs	4
Copybooks	5
Procedures (including JCL jobs)	25

Vendors were allotted 40 minutes during the workshop to show how they reengineered the application and 15 minutes to talk about their products and approaches. Vendors were also required to submit a handout for the workshop proceedings as described in Exhibit 1.

Exhibit 1 Description of vendor handout

Executive Summary
This section briefly describes the solution, the approach and the products used. The advantages and disadvantages of the solution are succinctly set forth.
Section A: The solution
1. Detailed solution
This section presents the solution to the proposed case study. Please indicate which category (or categories) of reengineering tools were used in your solution and why your solution fits into it (or them). You should also explain how your solution contributes to solving the types of problems associated with the category (or categories) described. The level of detail is left to the discretion of the vendor.
2. Recommendations
This section is actually an implementation plan for the solution. It covers only the implementation and operation of the proposed solution and not the prerequisite infrastructure.
3. Resources required to prepare the solution
This section describes the vendor activities involved in preparing the solution to the case study. How much time was needed? How many resources took part? What effort was involved? What kind of expertise was needed?
Section B: Approach and products used
1. Description of the approach and the products used
This section describes the approach and the work method used. What steps did you follow?
The products used are then briefly described. What functions or subsets of functions (modules) of the product served to produce which part of the solution (or element of the deliverable)? What are the benefits associated with the use of each of the functions of the product(s)?
2. Development platform
What type of platform did you work on (microcomputer, mainframe, single workstation, multiple workstation, etc.)? Do interfaces exist to migrate to other products? What are the characteristics of the repository used?
Vendors must position their product(s) within the product infrastructure of their choice (AD/Cycle, Cohesion, PCTE ...).
3. Utilization in various types of large-scale projects
Does your "solution" (approach and software) work with large-scale projects and more complex requirements, and different types of systems (transactional, informational, real-time, complex calculations, etc.)? For each type of system give examples of projects (or clients) that have used your approach or tools.
4. Recommendations
Include here the recommendations relevant to the implementation of a support infrastructure for the development of the approach and tools used in your solution.
5. Resources required for the implementation of the platform
This section contains an implementation plan for the infrastructure. It indicates the equipment and personnel required. A typical schedule may be included. Acquisition and training costs are also indicated.

CHALLENGES AND LESSONS LEARNED

The following paragraphs present an overview of the difficulties and challenges encountered by the workshop committee:

1- Even though many corporations were contacted for legacy applications, only two candidates for reengineering were submitted to the program committee, and only one candidate met the selection criteria. Also, by selecting only one application with very specific technical characteristics, the committee obviously biased the case study towards certain vendors at the expense of others.

2- Because the company's name is inscribed throughout an application (program comments, report headings, screen headings...), it would be very difficult for a company to submit an application confidentially. Programs contain detailed business procedures and confidential information that managers were unwilling to make public.

Obviously, the transit authority that supplied the legacy application did not want its application to be partially or completely reengineered and then sold or transferred without its consent. Vendors were therefore not permitted to use the application outside this workshop. (Vendors who wished to reuse this case study in other settings were not very pleased with this condition)

3- As this project went further on, it was progressively realized that many technical questions and issues regarding the application could not be resolved independently by the vendor and would require the assistance of the transit authority's technical staff. For obvious reasons, the transit authority could not afford to spend hours with each vendor to answer their questions. It could also be anticipated that some vendors could complain that other vendors had access to privileged information.

4- When the GGL staged the special CASE workshop in 1991, it had no difficulty persuading vendors to participate in the event. On the basis of the previous year's success and with the market interest on software reengineering, the workshop committee had anticipated little difficulty in persuading vendors to participate in the reengineering workshop. However, most vendors contacted stressed that too much effort would be required to solve this relatively simple case study, and that for such an effort to be worthwhile for them, they would have to be able to reuse the case study in other settings.

Competition does not seem to be currently as fierce amongst software reengineering tool vendors as it was amongst CASE vendors in 1991. Market pressure may therefore have been less strong on vendors for them to participate in this reengineering workshop than it was for the 1991 CASE workshop. Vendors may also have shied away because they could only partially cover the reengineering tools taxonomy.

Due to the difficulties encountered, the reengineering workshop had to be postponed indefinitely. The workshop committee came to the conclusion that even though the conference format met the needs of the GGL membership, the marketplace was not yet in a position to tackle this type of innovative and challenging marketing (a limited scope real-life case study). The committee believes that this maybe due to the immaturity of the reengineering domain and not only to lack of resources.

REFERENCE

ARNO92 Arnold, R. S., Software Reengineering, Tutorial notes, Symposium 92 GIRICO, Le génie logiciel-Avantages stratégiques et réalisations, Montreal, Canada, 1992

ACKNOWLEDGMENTS

On behalf of the Groupe génie logiciel, we would like to express our thanks to the Société de Transport de la Communauté Urbaine de Montréal that kindly supplied the legacy application for this case study.

We would also like to recognize the dedication of a group of volunteers who contributed to the workshop committee. They are: Richard Basque, Francine Bélanger, Vianney Côté, Manon Dion, Gaétan Desfossés, Benoit Garceau, Christian Grou, Michael Konaté, Ettore Merlo, and Normand Rivard. The Groupe génie logiciel thanks you.