

Reuse versus Rewrite: An Empirical Study of Alternative Software Development Methods for Web-enabling Mission-critical COBOL/CICS Legacy Applications

Howard A. Kanter, Ed. D., CPA/CITP

Director, The Laboratory for Software Metrics and

Associate Professor

The School of Accountancy and Management Information Systems

The Charles H. Kellstadt

Graduate School of Business Administration

DePaul University

Chicago, Illinois 60604

USA

312 362 8449

hkanter@depaul.edu

Thomas J. Muscarello, M.S., Ph. D.

Director, The Laboratory for Software Metrics and

Director of External Initiatives

School of Computer Science, Telecommunications and Information Systems

DePaul University

Chicago, Illinois 60604

USA

312 362 8737

muscarello@cs.depaul.edu



Fujitsu Software Corporation
1250 East Arques Avenue
Sunnyvale, CA 94085
Phone: (800) 545-6774 or (408) 428-0300
Email: cobol@netcobol.com
Web: www.netcobol.com

© 2005 Fujitsu Software Corporation. All rights reserved. May 2005

Microsoft and Windows are registered trademarks of Microsoft Corporation in the United States and other countries.

Fujitsu and NetCOBOL are registered trademarks of Fujitsu Limited in the United States and other countries.

Contents

Executive Summary	5
Experimental Design of the Study	6
Purpose of Study	6
Research Design and Methodology.....	6
Research Setting	6
Equipment/Environments	6
Study Assumptions	6
Subjects of the Study.....	7
The Application (Test) System	7
Study Methodology.....	7
Empirical Results.....	8
Data Analysis and Presentation	8
Analysis by SDLC Phase	9
Revise Mean Java Rewrite	12
Conclusions	13
Appendices	14
Issues Related to Data Conversion	14
Issues Related to n Source Code Conversion.....	14
Issues Related to the Time Required for Conversion	14
NeoKicks Knowledge Domains/Subjects.....	15

Executive Summary

Of major importance to organizations that depend on mission-critical legacy systems is the issue of how to most effectively Web-enable these systems. This study focused on several accepted methodologies for Web-enabling legacy systems. The methodologies chosen were adapted from the Legacy Transformation - Part 1 Methodology developed by Declan Good that presents a comprehensive collection of generally available approaches to the Web-enabling problem.

The methods chosen for study are:

1. Adaptation of a legacy COBOL/CICS system using COBOL and the CICS migration solution, supplied by Fujitsu Software Corporation.
2. Full rewrite of the legacy system using the Java programming language.

Based on the data collected, analyzed and presented, the following conclusions appear valid at this time:

- The time to convert the application chosen for testing using COBOL and the CICS migration solution was on average less than 3% of the time needed to rewrite the application in JAVA.
- The skill level of the programmer had little overall effect on conversion time when compared to time needed to rewrite code in JAVA. At all skill levels tested (levels 1 through 3) conversion times were less than 5% of JAVA rewrite times.
- Training times to use COBOL and the CICS migration solution for conversion work on straightforward applications is minimal.
- With appropriate training, less skilled employees or paraprofessionals, rather than skilled technicians can do most of the work required to reuse the system using the CICS migration solution in a fraction of the time needed to rewrite.

Experimental Design of the Study

Purpose of Study

Given the pressing need for organizations around the world to Web-orient their business processes, the purpose of this study is to gather and present empirical and anecdotal evidence related to the efficiency and effectiveness of two commonly considered "Webification" approaches. These approaches are:

- Revision (using NetCOBOL for .NET and the CICS migration solution.)
- Total system rewrite (using the Java language.)

To facilitate this investigation Fujitsu Software Corporation provided appropriate software tools. These included NetCOBOL for .NET, NetCOBOL for Windows, and the CICS migration solution.

Research Design and Methodology

Research Setting

Specifically, the tasks used in this research were developed by the researcher based on the research literature and professional practice. The tasks were intended to test activities classically required in the conversion of an enterprise legacy COBOL/ CICS application to a web enabled application. The subjects did not see the tasks prior to being tested.

Equipment/Environments

The testing environments were similar to those used in industry. Subjects worked at individual workstations situated in an office setting. Appropriate microcomputer software tools were used as described in the sections delineating the approaches to Webification which were tested.

Study Assumptions

The study approach was based on the situation classically found in standard IT environments when new software tools are obtained and measuring their effect is required by management:

- Programmers work with whatever tools management provides.
- Management obtains new tools that it believes will increase productivity.
- Programmers are trained in the new tools.
- The new tools are used.
- Management wishes to measure the increase in productivity that should justify the cost of obtaining and utilizing the product.
- Productivity is comprised of both speed and accuracy.
- End User satisfaction will make or break a system. The tools must support the creation of useful, usable, satisfying interfaces.
- The legacy system to be converted is not to be changed, i.e. the processes, screen display information, data captured and files maintained are to be retained.

Subjects of the Study

A sample of experienced IT professionals executed each phase of the study tasks. Expertise was varied. As such the subjects could be categorized based on years of experience, which is a standard HR method used in setting hiring qualifications and salary levels.

Subjects were representative of three levels of programming expertise:

- *Level 3* – Greater than 12 years of experience with significant experience at programming legacy systems. The average amount of experience was greater than 15 years.
- *Level 2* – These subjects had from 5 to 12 years of experience.
- *Level 1* – Programmers with little programming experience. These subjects had less than 5 years of experience.

Each subject received a condensed version of the ordinary training course in each of the software tools used and had access to full documentation and escalated support. The tools involved in the study were:

- NetCOBOL for .NET
- NetCOBOL for Windows
- Fujitsu Software's CICS migration solution
- A JAVA environment with all necessary features and tools.

The subject professionals worked as paired programmer teams. Where skills needed for task completion were multidisciplinary (e.g. COBOL/business logic expertise and Java programming skills) the teams were comprised of skill experts in each skill. This pairing of professionals represents the type of situation extant in industry, i.e. it is hard if not, in some cases, impossible to find IT personnel with the necessary depth of experience in COBOL/mainframe/business logic and new high level languages such as Microsoft packages, C++ or Java, which are common tools for building Web front ends.

The Application (Test) System

The application system used for testing purposes in this study was originally a payroll system designed for teaching purposes. As such, the system incorporated many of the standard features and programming techniques of such a system. However, larger systems, or systems using less common features, may involve attention to details not covered in this study.

Study Methodology

The specific research question of this study may be phrased so as to ask about the relationship between two variables.

- One variable to be measured is the time taken, by activity, to convert a legacy application to a web-enabled version.
- The other variable to be measured is a group membership variable: Webification approach used and the conversion tool environment.

Empirical Results

The following tables and discussions describe the methods studied and the results obtained:

Method 1

Revise Application Processing by using Net COBOL for .NET and the CICS migration solution. (Translates CICS related functions to .Net compatible COBOL code based on ASP .NET)

Method 2 - Java rewrite

The Java rewrite took the longest time to accomplish for a variety of reasons. The Java programmer, although very competent in Java, was unfamiliar with COBOL making the communication of requirements more difficult. As would be true with writing new code in any programming language, the time for writing code, testing and debugging was significant. It is true that a mainframe COBOL system rewritten in Java may provide ancillary benefits, such as ease of future modification and reuse. However, as shown in Table 2, the rewrite in Java is done at a significant incremental cost and time when compared to the revision approach.

In addition, when following the Java rewrite approach, an additional requirement is the need to convert data files that have been maintained by COBOL programs. These files may use data-types that have no equivalent in Java (such as packed decimal.) There are a number of software tools available to convert COBOL data files to files accessible by "modern" languages such as JAVA. File conversion may take a substantial amount of time and increase the risk of corrupting the converted data.

Data Analysis and Presentation

Table 1: Time to complete revise/rewrite tasks for all groups and methods (Minutes)

SDLC Step***	Revise				Rewrite
	Level 3	Level 2	Level 1	Mean	Java
1. Requirements Analysis	1.0	1.0	1.0	1.0	240
2. Specification	0.0	0.0	0.0	0.0	150
3. Design	3.6	3.5	3.5	3.6	105
4. Development	20.0	20.0	19.9	20.0	600
5. Testing	4.9	5.1	5.1	5.0	90
6. Performance/Value Issues	0.0	0.0	0.0	0.0	30
7. Packed Decimal Conversion	0.0	0.0	0.0	0.0	60
TOTAL Time to Revise	29.5	29.6	29.5	29.5	1275
Total Time to Revise as % of Java Rewrite				2.3%	100%
Java Rewrite					

*** SDLC - Software Development Life Cycle

Analysis by SDLC Phase

Note: The CICS conversion tool's processes are very straightforward and the subjects of the revision method used default names and folder locations. Consequently there was very little variation in the timings between experience levels, with most of the time being taken by computer processing (on a relatively low-powered machine).

1. **Requirements Analysis** - the mean time to revise is 0.42% of the time to rewrite. Rewriting takes so much more time because, even though the requirements do not change, it is still necessary to define requirements for the Java programmer.
2. **Specification** - the mean time to revise is 0% of the time to rewrite. Rewriting takes so much more time because, even though the specifications do not change, it is still necessary to define specification for the Java programmer. When following a revision approach, the legacy system to be revised contains the specifications within the programs which make up the legacy system.
3. **Design** - the mean time to revise is 3.43% of the time to rewrite. Rewriting takes so much more time because, even though the design does not change, it is still necessary to provide a design for the Java programmer. When revising, using a software tool, the design is implicit in the programs themselves.
4. **Development** - the mean time to revise is 3.33% of the time to rewrite. Revising takes so little time because the existing programs are converted by the revision software. Rewriting is a laborious undertaking, which introduces a high level of risk.
5. **Testing** - the mean time to revise is 5.66% of the time to rewrite. Rewriting takes so much more time because more new code has been written which requires extended testing.
6. **Assessing Performance/Value Issues** - Revising takes so much less time because the operating characteristics are not changed. Rewriting requires a production environment to be planned and created.
7. Packed Decimal Conversion of data for Java is required because this data form is not acceptable in Java. The conversion tool is able to process this data type.

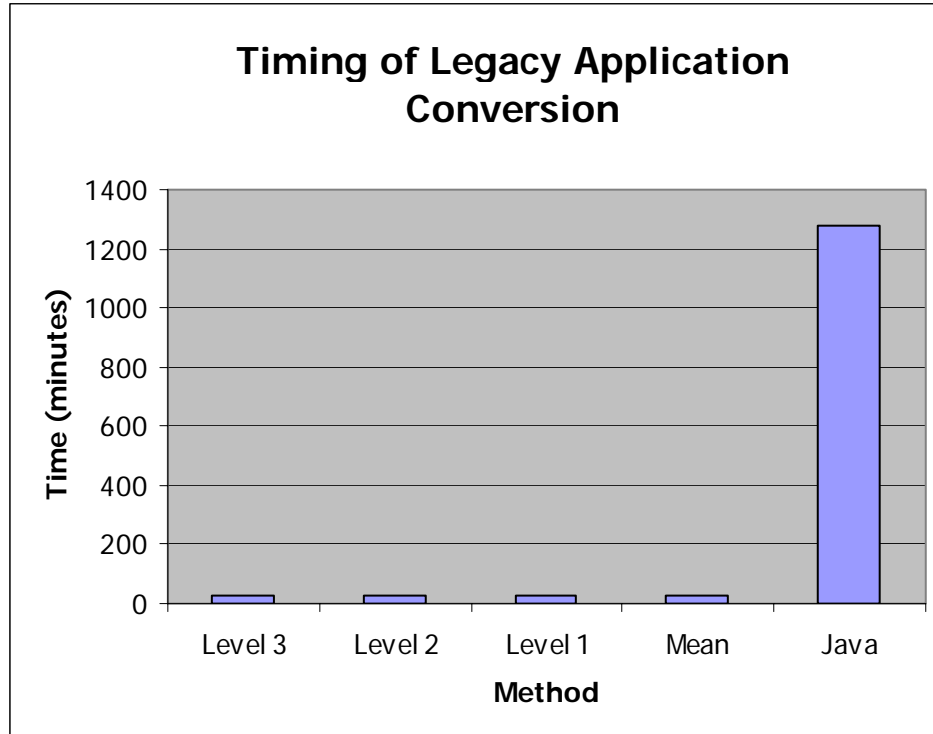


Chart 1. *Timing of Legacy Application Conversion*

Chart 1 above presents the total time to convert a legacy application using the CICS conversion tool, for each experience level, the mean time to convert for all levels and the time to rewrite the same application system using the Java language. The mean time for each experience level varied from the mean time taken by all levels by only 3 percent thus indicating that the subjects' difference in years of experience did not affect the time to revise the legacy system using the CICS conversion tool.

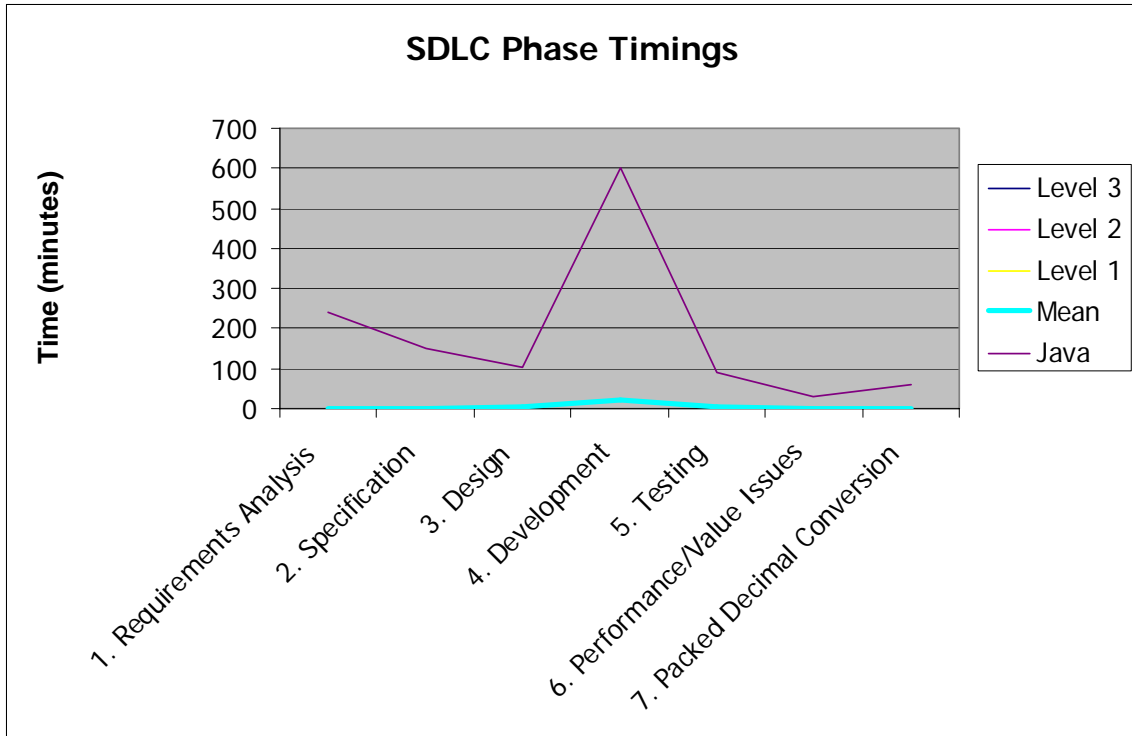


Chart 2. SDLC Phase Timings for All Methods (Minutes)

Chart 2 above depicts a comparison of times required to complete each SDLC phase of the revision and rewrite processes for all skill levels and methods. Note that there are significant variances in the time required for all phases of the Java rewrite method compared to the time required by the revision method (which utilizes the CICS conversion tool) for all phases of the conversion. The variances are greatest, however, in the requirements analysis, specification, and especially in development.

Revise Mean Java Rewrite

Table 2. Cost to Revise vs. Cost to Rewrite Salary average of \$54 per hour + 35% benefits: (Datamasters Salary Survey 2003).

SDLC Step	Revise Means		Rewrite Means		Cost Variance Rewrite-Revise
	Time	Cost	Time	Cost	
1. Requirements Analysis	1.0	0.9	240	213.6	212.7
2. Specification	0.0	0.0	150	133.5	133.5
3. Design	3.6	3.2	105	93.5	90.3
4. Development	20.0	17.8	600	534.0	516.2
5. Testing	5.0	4.5	90	80.1	76.6
6. Performance/Value Issues	0.0	0.0	30	26.7	26.7
7. Packed Decimal Conversion	0.0	0.0	60	53.4	53.4
TOTAL	29.5	26.3	1275	1134.8	1108.5
Total Time to Revise as % of Java Rewrite	2.3%		100%		

Table 2 above demonstrates that the cost, for each step in the SDLC, is substantially greater for the rewrite in Java method than for the same step in the revision method. It should be noted that the salary used to estimate the revision method was based on a relatively senior technical person (\$54 per hour + 35% benefits). However, the CICS migration tool used in the revision method is largely mechanical and revision could be done, in approximately 80% of the conversion, by a trained Para-professional at a substantially lower rate.

Conclusions

Based on the data collected, analyzed and presented above, the following conclusions appear valid at this time:

- The time to convert the application chosen for testing using COBOL and the CICS migration solution was on average less than 3% of the time needed to rewrite the application in JAVA.
- The skill level of the programmer had little overall effect on conversion time when compared to time needed to rewrite code in JAVA. At all skill levels (Level 3, Level 2, and Level 1) conversion times were less than 5% of JAVA rewrite times.
- Training times to use COBOL and the CICS migration solution for conversion work on straightforward applications is minimal.
- With appropriate training, paraprofessionals, rather than skilled technicians can do most of the work required to reuse the system using the CICS migration solution.
- In order to take advantage of the speed of conversion which comes with using the CICS migration tool, a complete inventory of source programs, copylibs, bms and data files must be accurately completed prior to the actual conversion process.

Appendices

Issues Related to Data Conversion

We will limit our discussion of data to non-relational data files. The migration of relational data files is generally accomplished by a Database Administrator (DBA). The DBA should be familiar with both the current and target platforms to facilitate the migration of the data. Several utilities are available to the DBA to assist with the migration.

Non-relational (QSAM and VSAM) data files are migrated in their current EBCDIC encoding to the new platform. This is generally accomplished via an FTP process using the binary format for transfer. Once on the target platform the data can then be converted to ASCII encoding. It should be noted the Fujitsu environment utilizes ASCII data only and can neither read nor process EBCDIC files.

Once the data is on the Wintel platform the Fujitsu Software Data File converter can be used to translate the EBCDIC characters to ASCII. A file layout, generally in the form of a copybook, will be required to inform the Data File Converter which fields to convert and which not to convert. Only the text fields (PIC X) require conversion.

Issues Related to n Source Code Conversion

Obviously if one is migrating an application you must have the source code in order to accomplish this. While this seems like an obvious observation it is none-the-less an often overlooked area that creates considerable confusion and frustration later in the migration process. A complete source code inventory should be accomplished prior to beginning any migration effort. Identify the application, or portion of an application, that you are going to migrate and identify all the source modules required to execute the application. The inventory is to include not only the COBOL source modules, but the JCL, PROCs and copybooks.

To migrate the source code from the mainframe to the Wintel environment, utilize FTP translating the EBCDIC characters to ASCII. It is recommended the file extensions of .COB, CBL, JCL and PRC be used for the COBOL source module, copybooks, JCL members and PROCs, respectively.

Issues Related to the Time Required for Conversion

"The race is not to the swift, but to the organized."

The amount of time required to accomplish moving the application from the mainframe depends on many factors. The most important factor is whether or not you have a complete source code inventory and whether or not it is up to date. In more than one instance clients assurances to the conversion team that the executable module in production was created from the source module in the production library, turned out to not be accurate. The actual source member was located after considerable research and time in a previous development team members' local work area. Source code synchronization will be the single most important and time consuming issue during the migration.

NeoKicks Knowledge Domains/Subjects

Knowledge/Subject	1	2	3	4	5	6	7	8
COBOL Language Programming	X (HVY)	X (LITE)	X (HVY)	X (HVY)	X (HVY)		X (HVY)	X
CICS Programs	X (HVY)		X (HVY)	X (LITE)	X (HVY)		X (HVY)	X
BMS	X		X		X		X	X
COPYBOOKS	X		X	X	X		X	X
Visual Studio®	X	X	X	X	X	X	X	X
Windows Navigation Techniques	X	X	X	X	X	X	X	X
C# Language Programming		X				X	X	X
Java Programming Language		X			X	X	X	
Testing Techniques	X	X	X	X	X	X	X	X
IBM JCL	X		X	X	X		X	X



Fujitsu Software Corporation
1250 East Arques Avenue
Sunnyvale, CA 94085
Phone: (800) 545-6774 or (408) 428-0300
Email: cobol@netcobol.com
Web: www.netcobol.com

© 2005 Fujitsu Software Corporation. All rights reserved. May 2005

Microsoft and Windows are registered trademarks of Microsoft Corporation in the United States and other countries.

Fujitsu and NetCOBOL are registered trademarks of Fujitsu Limited in the United States and other countries.