EVALUATION OF BEST TARGET PLATFORM FOR APPLICATION MIGRATION

MAHDI MALEKI

UNIVERSITI TEKNOLOGI MALAYSIA

EVALUATION OF BEST TARGET PLATFORM FOR APPLICATION MIGRATION

MAHDI MALEKI

A dissertation submitted in partial fulfilment of the
requirements for the award of the degree of
Master of Science (Computer Science)

Faculty of Computer Science and Information Systems
Universiti Teknologi Malaysia

JANUARY 2011

This thesis is dedicated to my parents and my love for their endless support and encouragement.

ACKNOWLEDGEMENT

I am heartily thankful to my supervisor, Dr. Shukor Abd Razak , whose encouragement, guidance and support from the initial to the final level enabled me to develop an understanding of the subject. It is a pleasure to thank those who made this thesis possible, specially to all my ex-colleagues for their endless supports and encouragments. Lastly, I offer my regards and blessings to all of those who supported me in any respect during the completion of the project.

ABSTRACT

This study is focus on finding common key factors of different platforms that should be considered in cross-platform application migration; moreover it aims to give weights to each individual factor. These weights are determined through two separate surveys. The result of this study is a prototype of a tool for compatibility ratio measurement based on the source and destination platforms to help system administrators and IT managers at choosing the best and most compatible platform in migration projects. This research also investigates issues related to application porting at different layer of IS application. Results from this investigation could be very useful to system administrator in having a better understanding and analyzing application migration issues before starting a project.

ABSTRAK

Fokus penyelidikan ini adalah untuk mencari faktor kekunci yang mempunyai persamaan dengan pelbagai platform yang berbeza untuk dipertimbangkan dalam proses perpindahan aplikasi (cross-platform). Selain itu, ia juga bertujuan untuk memberi pemberat kepada setiap faktor tersebut. Pemberat ini akan ditentukan melalui dua tinjauan yang berasingan. Hasil daripada keputusan kajian ini adalah sebuah prototaip untuk mengukur nisbah keserasian berdasarkan sumber dan tujuan platform tersebut. Ini bertujuan untuk membantu pentadbir sistem dan pengurus teknologi maklumat dalam memilih platform yang terbaik dan serasi dalam proses perpindahan aplikasi. Kajian ini juga memberi fokus kepada penyelidikan berkaitan isu-isu yang terlibat dalam perpindahan aplikasi di pelbagai peringkat yang berbeza untuk aplikasi sistem maklumat. Hasil daripada kajian tersebut amat berguna kepada pentadbir sistem dalam menganalisis permasalahan perpindahan aplikasi sebelum memulakan sesuatu projek perpindahan.

TABLE OF CONTENTS

CHAPTER	TITLE	PAGE
	DECLARATION	**II**
	DEDICATION	**III**
	ACKNOWLEDGEMENT	**IV**
	ABSTRACT	**V**
	ABSTRAK	**VI**
	TABLE OF CONTENTS	**VII**
	LIST OF TABLES	**XIII**
	LIST OF FIGURES	**XVI**
	LIST OF ABBREVIATIONS	**XVII**
1	**INTRODUCTION**	1
	1.1 Introduction	1
	1.2 Problem Background	2
	1.3 Problem Statement	6
	1.4 Project Aim	7
	1.5 Project Objectives	8
	1.6 Project Scope	8
	1.7 Significance of Project	10
	1.8 Organization of Report	10

2 LITERATURE REVIEW 11

2.1 Introduction 11

2.2 Cross-Platform Programming Languages 12

2.3 Application Migration/Porting 13

2.3.1 First Modernize then Port 14

2.3.2 First Port then Modernize 15

2.4 Legacy Application 16

2.5 Incompatibilities and Consequences 16

2.6 Existence Application Migration/Porting Case-Studies 18

2.6.1 Application Porting Issues for C++ Applications 18

2.6.2 Issues 19

2.6.2.1 Presentation Layer (PL) 20

2.6.2.2 Application Service Layer (ASL) 22

2.6.2.3 Lending Message Bus (LMB) 23

2.6.2.4 Business Layer (BL) 24

2.6.2.5 Data Service Layer (DSL) 26

2.6.2.6 Platform Service Layer (PSL) 27

2.6.3 Current Solution 30

2.6.3.1 Presentation Layer 31

2.6.3.2 Application Service Layer 34

2.6.3.3 Lending Message Bus 37

2.6.3.4 Business Layer 38

2.6.3.5 Data Service Layer 39

2.6.3.6 Platform Service Layer 39

2.6.3.7 Discussions 40

2.6.4 Migrate a Windows Application onto UNIX/Linux 43

2.6.4.1 Using Wind/U or MainWin as Source Code Emulator 44

2.6.4.2 Using Wine(a binary emulator) 45

2.6.4.3 Using some Cross-Platform Developing Toolkit such as Qt and Rewrite the Application Partially 46
2.6.4.4 Using a Absolutely Different Technology Like Java to Completely Rewriting the Application 47
2.6.5 Application Migration and Business Challenges 47
2.6.6 The Best Trusted Cross-Platform for Information Sharing 48
2.6.6.1 The Importance of Vendor Cooperation for Secure Solutions 50
2.7 Platform Virtualization 52
2.7.1 Cross-Platform Virtualization 54
2.7.2 SUN VirtualBox 55
2.7.2.1 VirtualBox's Features 56
2.8 Conclusion 57

3 RESEARCH METHODOLOGY 59
3.1 Introduction 59
3.2 Research Framework 60
3.2.1 Phase 1: Reviewing Literatures 60
3.2.2 Phase 2: Unstructured Interviews 62
3.2.3 Phase 3: Finding Common Key-factors 63
3.2.3.1 Questionnaire Method 64
3.2.3.2 Questionnaire Design 65
3.2.3.3 Questions 65
3.2.3.4 Questionnaire Distribution 66
3.2.3.5 First Survey 67
3.2.3.6 First Survey Results Analysis 69
3.2.4 Phase 4: Giving Weight to Common Key-factors 70

3.2.4.1 Second Survey 70
3.2.4.2 Second Survey Results Analysis 71
3.2.5 Phase 5: Defining Compatibility Ratio 72
3.2.6 Phase 6: Developing CR Measurement Tool 72
3.2.7 Phase 7: Evaluation 74
3.3 Conclusion 75

4 EXPERIMENT AND RESULTS 76
4.1 Introduction 76
4.2 Reviewing Literatures 77
4.3 Unstructured Interviews 78
4.4 Finding Common Key-factors 79
4.4.1 Pilot of First Survey 81
4.4.2 Experimental Result of First Survey 82
4.4.2.1 Common Key-Factors 84
4.5 Giving Weight to Common Key-Factors 85
4.5.1 Pilot of Second Survey 86
4.5.2 Experimental Result of Second Survey 86
4.6 Defining Compatibility Ratio 87
4.6.1 Weighting Key-Factor 90
4.6.2 Calculating Compatibility Ratio 93
4.7 Developing Compatibility Ratio Measurement Tool 94
4.7.0.1 Encased Application Migration Manifesto 96
4.7.1 Development Methodology 97
4.7.2 Implementation 98
4.7.2.1 Initial Analysis 99
4.7.2.2 Application Development 100
4.7.2.3 Security Analyzer 101
4.7.2.4 C++ Code Analyzer 103
4.7.2.5 Application front-end module 104
4.7.2.6 Web Analyzer 104

4.7.2.7 Header Analyzer 105
4.8 Evaluation 105
4.8.1 Phase 1: Migrate Terminal-Based Applications 107
4.8.1.1 Result of First Phase of Experiment 108
4.8.2 Phase 2: Migrate Standalone Applications 109
4.8.2.1 Result of Second Phase of Experiment 110
4.8.3 Phase 3: Migrate Web-Based Applications 111
4.8.3.1 Result of Third Phase of Experimnet 111
4.8.4 Phase 4: Migrate Mid-tier Applications 112
4.8.4.1 Result of Fourth Phase of Experimnet 113
4.9 Conclusion 114

5 ANALYSIS AND DISCUSSION 115
5.1 Introduction 115
5.2 Discussion on Findings of Phase 1: Reviewing Literatures 116
5.3 Discussion on Findings of Phase 2: Unstructured Interviews 118
5.3.1 Finding Direction of Study 118
5.3.2 Importance of Application Migration 118
5.3.3 Comparison between Different Operating Systems 119
5.3.4 Issues and Consequences of Application Migration 120
5.4 Discussion on Findings of Phase 3: Finding Common Key-factors 121
5.5 Discussion on Findings of Phase 4: Giving Weight to Common Key-Factors 125
5.6 Discussion on Findings of Phase 5: Defining Compatibility Ratio 135

	5.7	Discussion on Findings of Phase 6: Developing Compatibility Ratio Measurement Tool	137
	5.8	Discussion on Findings of Phase 7: Evaluation	138
	5.9	Conclusion	141
6	**CONCLUSION**		142
	6.1	Introduction	142
	6.2	Achievements	143
	6.3	Limitation	145
	6.4	Future Works	146
	REFERENCES		**147**
	APPENDIX A		**151**
	APPENDIX B		**157**

LIST OF TABLES

TABLE NO.	TITLE	PAGE
2.1	Programming Language Popularity Index (TIOBE, 2010).	13
2.2	Historic Programming Language Popularity Ranking (TIOBE, 2010).	19
2.3	Comparative Dependability of Fonts for Web-use: Ordered from Most to Least Frequently Found in Windows (codestyle.org, 2008).	21
2.4	Browser Compatibility Chart	22
2.5	Shows Size of Each Type in Different Models	23
2.6	File System Comparison Chart	25
2.7	Operating Systems Differences (Part I)	28
2.8	Operating Systems Differences (Part II)	29
2.9	Filename Restrictions among Different Operating Systems	33
2.10	Difference between 32-bit and 64-bit Type Enumeration	36
4.1	List of different manifestos involves in application migration planning and assessment phase	78
4.2	Relationship Between Factors Asked in The First Survey and Defined Manifesto	80
4.3	Relation Between Question of First Survey and Objected Statements	81
4.4	First Survey Participants Distribution	83
4.5	The Final Result of First Survey	83

4.6	List of Key-Factors founded through literatures and result of first survey.(Note: order of expressing key-factors does not implies any prioritization)	84
4.7	First Survey Participants Distribution	87
4.8	Findings from Second Survey	88
4.9	List of Common Key-Factors in Application Migration Process.(Note: order of listing key-factors does not implies any prioritization)	89
4.10	List of Dependent Key-Factors in Application Migration Process. (Note: ”*” mean key-factor has correlation with specific type of user interface)	90
4.11	Scaling Second Survey Results to Coefficient Values	91
4.12	List of Key-factor Scaled Coefficients Followed by Migration Direction. (Note: WtU instanced for Windows-to-UNIX Migration and UtW is for UNIX-to-Windows Migration)	92
4.13	List of Application Migration Manifestos Covered by Application Migration Tool(AMT)	96
4.14	Chosen Application as Case-Studies (source: ”www.planet-source-code.com”)	107
4.15	Statistics of Applications-First Phase: Migrating Terminal-Based Applications	108
4.16	Summery of Observation on Migrating Terminal-Based Applications	109
4.17	Statistics of application-Second Phase: Standalone Applications	109
4.18	Summery of Observation on Migrating Standalone Applications	110
4.19	Statistics of application-Third Phase: Web-Based Applications	111
4.20	Summery of Observation on Migrating Web-Based Applications	112
4.21	Statistics of application-Fourth Phase: Mid-Tier Applications	113
4.22	Summery of Observation on Migrating Mid-Tier Applications	114
5.1	Relation Between Question of First Survey and Objected Statements	124

5.2	Relationship Between Key-Factors and Application Migration Manifesto	125
5.3	Relationship between founded key-factors and questions in second survey	130
5.4	Analytical Result of Second Survey	131
5.5	One-way ANOVA: Participant Groups	132
5.6	One-way ANOVA: Key-Factors	133
5.7	Compatibility Ratio (CR) for Window to UNIX Migration Based on Different Type of Application and User Interfaces	136
5.8	Compatibility Ratio (CR) for UNIX to Windows Migration Based on Different Type of Application and User Interfaces	137
5.9	Summery of Observation on Application Migrating Simulation	140

LIST OF FIGURES

FIGURE NO.	TITLE	PAGE
2.1	Cost Distribution for Two Well-known Migration Scenarios: Port and Modernize Vs. Modernize and Port(Mercer, 2006).	15
2.2	Windows Server 2003 architecture (MicrosoftTech, 2006).	41
2.3	UNIX architecture (MicrosoftTech, 2006).	42
2.4	Schematic of Wind/U (Rajagopal, 1998).	44
2.5	Virtualization can be implemented in every layer of the computing platform (Ramanathan and Bruening, 2004).	53
3.1	Research Framework	61
4.1	Security Analyzer Output in XML Format	101
4.2	C++ Code Analyzer Output in XML Format	102
4.3	Application Migration Tool	103
4.4	Web Analyzer Sample Report	105
4.5	Header Analyzer Sample Report	106
5.1	One-way ANOVA: Participant Groups(Individual 99% CIs For Mean Based on Pooled StDev)	133
5.2	One-way ANOVA: Key-Factors(Individual 99% CIs For Mean Based on Pooled StDev)	134

LIST OF ABBREVIATIONS

ACL	-	Access Control List
AMT	-	Application Migration Tool
API	-	Application Programming Interface
ASL	-	Application Service Layer
BL	-	Business Layer
CLI	-	Command Line Interface
CR	-	Compatibility Ratio
CRM	-	Compatibility Ratio Measurement
DSL	-	Data Service Layer
GUI	-	Graphical User Interface
HTML	-	Hypertext Transfer Protocol
LMB	-	Lending Message Bus
PL	-	Presentation Layer
POSIX	-	Portable Operating System Interface for Unix
PSL	-	Platform Service Layer
RAS	-	Reliability, Availability, Serviceability

RIO	-	Return of Investment
TCO	-	Total Cost of Ownership
UI	-	User Interface
UtW	-	UNIX-to-WINDOWS
WtU	-	WINDOWS-to-UNIX
WUI	-	Web-based User Interface
XHTML	-	Extensible Hypertext Transfer Protocol
XML	-	Extensible Markup Language
	-	

LIST OF APPENDICES

APPENDIX NO.	TITLE	PAGE
A	FIRST SURVEY	150
B	SECOND SURVEY	152

CHAPTER 1

INTRODUCTION

1.1 Introduction

According to the latest surveys about programming language popularity, most companies deciding to develop an enterprise application preferred to use one of the cross-platform programming languages for development (TIOBE, 2010).

Most of the reasons given for using cross-platform programming language is related to their prominent features such as flexibility and portability. Furthermore, the future dictates a need to cater to a growing number of application users, the changing of security policies and so forth that maybe compel IT managers to choose another platform to achieve more performance, extra capabilities, security enhancement, decreasing TCO (Total Cost of Ownership) and increasing RAS (Reliability, Availability, Serviceability) (McCarthy, 2008).

There are many existing ways to migrate an operating Information System (IS) enterprise application from one development environment to another. Choosing the most suitable and compatible target platform is one of big challenges of every project migration process (Richter *et al.*, 2006).

As stated by Economides and Katsamakas (2006), there is a need for research in understanding switching costs and adoption strategies; hence this study aims to focus on finding common key factors of different platforms that must be considered in cross-platform application migration. Subsequently giving weight to each individual factors by conducting two separate surveys and finally proposing a tool to estimate compatibility ratio of the base and destination platforms .The proposed tool will help system administrators and IT Managers to choose the best and most compatible platform as destination platform to decrease the potential problems in application migration.

1.2 Problem Background

These days, the IT environment is continually changing, with changes including the size of application, capabilities, applicability, functionality and limitations of legacy applications. Most IT managers may decide to migrate their legacy application from the pre-existing production environment to a different environment that would prove more beneficial as proposed by newer fashionable platforms and technologies (Heymans *et al.*, 2007).

Migration can be defined as the process of porting from one operating environment to another heterogeneous or homogeneous operating environment. Usually, it is considered as moving to better environment. such as, migrating from Windows NT Server to the its newer version Windows 2003 Server may be considered as a migration

project because there are some new features are exploited through this migration, old configurations do not need to be changed, and it involves some steps to make sure that current applications will be operational in the new environment. Moreover, it also can be defined as porting data from one type of database to another type of database (Bitpipe, 2010).

As Glass' Law states, requirement deficiencies are the prime reason of project failure, according to Poniatowski (2003), every single cross-platform application migration procedure can be divided into following steps:

- **Planning and detailed assessment :** It includes hardware inventory, readiness reporting and a compatibility analysis.
- **Tools for development and customization :** These tools are use for modernizing and making application ready before migration and debugging and modifying after migration.
- **Test migration :** Testing ensures that all issues regarding the migrated application from the source to the target platform are identified and any problems that might occur mitigated.
- **Application migration :** It is the process of moving application from source to destination platform.
- **Acceptance :** This process is started after migration has been finished successfully to get top level manager's acceptance.
- **Installation, warranty, and product support**: After getting acceptance from top level managers is the time to put migrated application in real production environment.

This study will help IT decision makers at the first step of application migration procedure which is planning and detailed assessment.

Migration usually happens between UNIX and Linux based platforms. For an instance an application which had been developed base on C++ in past two decades has some performance issues. The security department nags about low-level security

considerations while the management level complains about applications' availability and reliability as a result of frequent crashes on the current platform.

Regarding to these consideration IS department decide to migrate current application to another platform which is most stable; secure and efficient to increase stability, functionality, performance and in the other hand decreasing TCO and risks of their legacy application.

According to the all documentation and release notes which are issued by different companies or development teams they found more than one candidate for choosing as destination platform. Now, this question comes into their minds that which platform is the most congruous and suitable destination in subjected application migration process.

During each phase of application migration process system administrator might face with some issues at different layer of application, for going in depth and find coexistence incompatibilities, it is better to categorize these issues by layers of application; Application can break into five layers and to make it more comprehensive, platform service layer is added as sixth layer:

- Presentation layer
- Application service layer
- Lending message bus layer (optional)
- Business layer
- Data service layer
- Platform service layer

Moreover, based on Bierhoff *et al.* (2007), every single IS application can be defined as a series of components plus communication between components. Hence this study has to study these two major categories separately. By focusing on the

nature of components, they can be categorized into four sub-categories: Functionality supply, Infrastructure expectations, Control model and Data manipulation. Based on communication between components, these components can be divided into two sub-categories: Asynchronous communication and Message data model (Bierhoff *et al.*, 2007). According to Inglenet-Business-Solution (2001), there are two well-known scenarios for migration (porting) which are:"Port and modernize", and "Modernize and port".

Another example is web applications which are deployed base on Java technology. It is well-known that Java technology is one of the most famous cross-platforms developing language, therefore the application migration process from legacy environment to another heterogeneous environment should be seamless.

Contrary to popular believes incompatibilities certain functions such as supported fonts, page layouts, table layouts and browser dependency can affect visibility and reliability of web application. Fonts used in ported application may not be installed at the destination, compatible browser might not be supported in the new environment, and many other possible issues that are related to third-party packages such as security modules and so on (Xu *et al.*, 2003).

Different systems in interaction with each other sometimes reveal some Unknown Errors. These incompatibilities and malfunctioning behavior became more obvious after an unsuccessful platform migration especially when target platform is chosen wrongly. It is commenced with overwhelming challenges for system administrator to make it ready for running ported application and end up with instability in application, increasing of total cost ownership (TCO) and decreasing in accessibility, reliability and serviceability (RAS) of IT environment.

According to Inglenet-Business-Solution (2001) around more than 50 percent of migration project after two to four years are abandoned as failures which cost lots of

money and wasting resources and time.

The problem is that, there is no comparison reference or tool which can help IT Managers and system specialists to choose the best target platform to prevent or reduce the possibility for migration project failure.

1.3 Problem Statement

According to Economides and Katsamakas (2006) there is needs to understand switching costs and adoption strategies for enterprise applications but until now there is no reference neither any tool can be used by system administrator or IT Manager to choose the best target in cross-platform application migration.

As (Richter *et al.*, 2006) states, choosing the most suitable and compatible target platform are one of the big challenges of every project migration process. Accordingly, this study will propose a standard method to evaluating competitive ratio between the source and destination platform to decrease the risk of application migration between two heterogeneous environments.

This study focuses upon finding the common key-factors and give weights to each of individual key-factors. Furthermore, this study defines a routine to find the best platform to be used in cross-platform application migration project.

The main research question which will be answered in this study is:

- What are key operators for evaluating compatibility ratio in application migration process?

Supporting research questions are:

1. How to find common key factors?
2. How to give weight to different key-factor?
3. How to measure Compatibility Ratio (CR) in application migration?

1.4 Project Aim

This study aims on finding common key factors between different platforms that must be considered in cross-platform application migration and the assigning of weight to each of individual factors. These are achieved through two surveys. This study will propose a tool for compatibility ratio measurement regarding the origin/source and destination platforms to help system administrators and IT Managers to choose the best platform in migration projects.

1.5 Project Objectives

This study will initially identify the common key-factors between different platforms, followed by the development of a compatibility ratio measurement tool. The proposed tool will be evaluated through emulation. Accordingly objectives of this ongoing study are:

1. To identify common key-factors of different hardware/software platforms that related to cross-platform application migration.
2. To develop a compatibility ratio measurement tool for cross-platform application migration.
3. To evaluate the developed compatibility ratio measurement tool for cross-platform application migration.

1.6 Project Scope

In this research, C++ programming language is chosen as the basis of the research. C++ is one of the most popular cross-platform programming languages, and it is also a completely structural programming language. Surveys are going to be based on this programming language that has different paradigm on the basis of distinctive application types:

1. Terminal-Based Applications (CLI).
2. Web-Based Applications.
3. Graphical User Interface (GUI) Standalone Applications.
4. Mid-tier Application.

Three heterogeneous operating systems are going be used as original and target platforms in this study. Each of these three operating systems has their own specific architecture, different model of supports and finally from three different vendors. Intended operating systems are as following:

1. Red Hat Linux AS 4.5
2. Sun Solaris 10
3. Microsoft Windows Server 2003

Windows Server 2003 is one the most well-known operating system which is developed by Microsoft, this operating system is inspired from Windows NT by making it much stable and featuring with new technologies and many innovative support services.

Solaris 10 is a UNIX-based operating system that is developed by Sun Microsystems; it is well-known due to its scalability and also supporting SPARC processor technology. There is two different version of Solaris 10 which one of them is SPARC based and another is working on x86-based server, in this study x86-based operating system is chosen.

Red Hat Linux is also one of the different distributions of UNIX-based operating system which is developed and supported by Red Hat Company. This is not a free operating system and originally knows as Red Hat commercial Linux.

1.7 Significance of Project

While the programming languages C++ and C are both categorized as cross-platform application development programming languages by practicing the American National Standards Institute (ANSI), the International Standards Organization (ISO)(ISO, 2003; American National Standards Institute and Computer and Business Equipment Manufacturers Association and Secretariat, 1989) and C library and the Standard Template Library(STL)(Nelson, 1995) standards to have platform independent application ,there are many existence issues when one legacy application which is developed based on C/C++ is going to be ported from original platform to another heterogeneous platform. So there is a need to understand switching costs and adoption strategies for enterprise applications which is developed based-on C++ programming language in every application migration process.

1.8 Organization of Report

This report contains 6 chapters which categorized as the following parts: The first chapter is introduction. In this chapter, aim, objectives, scopes and problem background of the study are being identified. In Chapter 2, researcher get into cross-platform programming languages concepts which are defined in the scope of this project then continue with migration (porting) routines to follow up minor and major porting issues. Chapter 3 discusses about research methodology: In this chapter, focus is given more on the research methodology to be carried out based on the objectives. Moreover, the framework of the study will be described together with the proposed standards. Chapter 4 presents findings of this study, chapter 5 presents discussion about results of each pahse of study and finally Chapter 6 concludes the report.

REFERENCES

American National Standards Institute and Computer and Business Equipment Manufacturers Association and Secretariat (”1989”). *American National Standard for information systems: programming language: C: ANSI X3.159-1989*. pub-ANSI:adr: pub-ANSI.

Bierhoff, K., Grechanik, M. and Liongosari, E. (2007). Architectural Mismatch in Service-Oriented Architectures. In *Systems Development in SOA Environments*. may. 4 –4. doi:10.1109/SDSOA.2007.2.

Bishop, J. and Horspool, N. (2003). Cross-Platform Development: Software that Lasts. *Annual IEEE/NASA Software Engineering Workshop SEW-30*. 39(2006), 9.

Bitpipe (2010). *Software Migration:Definistion*. doi:http://www.bitpipe.com/tlist/Software-Migration.html.

Burgess, D. T. F. (2001). A General Introduction to the design of questionnaires for survey research. *Information Systems Services*. doi:http://www.leeds.ac.uk/iss/documentation/top/top2.pdf.

codestyle.org (2008). *Web Font Survey*. Technical report. doi:http://www.codestyle.org/css/font-family/sampler-CombinedResults.shtml.

Economides, N. and Katsamakas, E. (2006). Linux vs. Windows: A Comparison of Application and Platform Innovation Incentives for Open Source and Proprietary Software Platforms. In Bitzer, J. and Schroder, P. J. (Eds.) *The Economics of Open Source Software Development*. (pp. 207 – 218). Amsterdam: Elsevier. ISBN 978-0-44-452769-1. doi:DOI:10.1016/B978-044452769-1/50010-X. Retrievable at http://www.sciencedirect.com/science/article/B86TN-4PB7H82-F/2/21b1f996676f70a7d5fe58d0ee70cc87.

Fricker, R. D. and Elliott, M. N. (2002). *Conducting Research Surveys Via E-Mail and the Web*. Rand Corporation, The. ISBN 0833031104.

Gauch, R. R. (2009). Measurements They’ever Exact. In *It’s Great! Oops, No It Isn’t*. (pp. 65–71). Springer Netherlands. ISBN 978-1-4020-8907-7. Retrievable at http://dx.doi.org/10.1007/978-1-4020-8907-7_8.

Heymans, L., der Beken, T. V. and Wilson, B. (2007). Testing Techniques for the Cross-platform Migration of Very Large Interactive Applications. *Software Maintenance and Reengineering, European Conference on*. 0, 323–324. ISSN 1534-5351. doi: http://doi.ieeecomputersociety.org/10.1109/CSMR.2007.46.

Inglenet-Business-Solution (2001). *A CASE STUDY OF PLATFORM MIGRATION FROM UNISYS 2200 TO UNIX ALBERTA BLUE CROSS PORT PROJECT*. Technical report. Inglenet Business Solutions. doi:http://www.inglenet.com/downloads/Blue_Cross_Case_Study_-_Detailed.pdf.

ISO (2003). *ISO/IEC 14882:2003: Programming languages: C++*. International Organization for Standardazation. Retrievable at http://www.iso.org/iso/en/CatalogueDetailPage.CatalogueDetail?CSNUMBER=38110.

Karpov, A. and Ryzhkov, E. (2007). *20 issues of porting C++ code on the 64-bit platform*. doi:http://www.viva64.com/content/articles/64-bit-development/?f=20_issues_of_porting_C++_code_on_the_64-bit_platform.html&lang=en&content=64-bit-development.

Kharitonov, E. V. (2000). A Method of Making Subjective Measurements Compatible with Hierarchical Matrices of Preference Ratios. *Measurement Techniques*. 43, 747–751. ISSN 0543-1972. Retrievable at http://dx.doi.org/10.1023/A:1026689404649.

Kuhn, M. (2009). *UTF-8 and Unicode FAQ for Unix and Linux*. doi:http://www.cl.cam.ac.uk/~mgk25/unicode.html.

McCarthy, S. P. (2008). *Choosing the Right Platform for Trusted Cross-Platform Information Sharing*. Technical report. doi:http://www.linux.com/learn/whitepapers/doc/11/raw.

Mercer (2006). *Migration Decision-Maker Interviews*. Technical report. doi:http://download.microsoft.com/download/E/A/0/EA0F6F0B-BAA2-46B1-9EBC-7F28EFA7C508/MercerWhitePaper%20.pdf.

MicrosoftTech (2006). *UNIX Custom Application Migration Guide*. (2nd ed.). Microsoft Tech Net.

Nelson, M. (1995). *C++ Program Guide to Standard Template Library*. Foster City, CA, USA: IDG Books Worldwide, Inc. ISBN 1568843143.

Oblitz, T. R. and Mueller, F. (2000). Combining Multi-Threading with Asynchronous Communication. In *In Myrinet User Group Conference*. 100–121.

Poniatowski, M. (2003). *Linux on HP Integrity Servers*. Upper Saddle River, NJ, USA: Prentice Hall PTR. ISBN 0131400002.

Rajagopal, R. (1998). *Windows NT, UNIX, NetWare migration and coexistence: a professional's guide*. vol. 1. CRC Press.

Ramanathan, R. and Bruening, F. (2004). *Virtualization: Bringing Flexibility and New Capabilities to Computing Platforms*. Technical report. doi:ftp://download.intel.

com/technology/computing/archinnov/teraera/download/Virtualization_0604.pdf.

Richter, K., Nichols, J., Gajos, K. and Seffah, A. (2006). The many faces of consistency in cross-platform design. In *CHI '06: CHI '06 extended abstracts on Human factors in computing systems*. New York, NY, USA: ACM. ISBN 1-59593-298-4, 1639–1642. doi:http://doi.acm.org/10.1145/1125451.1125751.

Roth, M. (2009). *Method for developing platform independent launchable applications*. Technical Report 10/978,517. doi:www.freepatentsonline.com/7610577.html.

Springer (2001). compatibility analysis. In *Computer Science and Communications Dictionary*. (pp. 259–259). Springer US. ISBN 978-1-4020-0613-5. Retrievable at http://dx.doi.org/10.1007/1-4020-0613-6_3222.

SunMicosystem (2005). *Converting 32bit Applications Into 64bit Applications Things to Consider*. doi:http://developers.sun.com/solaris/articles/ILP32toLP64Issues.html.

Taleb, M., Seffah, A. and Abran, A. (2007). Patterns-Oriented Design Applied to Cross-Platform Web-based Interactive Systems. In *Information Reuse and Integration*. aug. 122 –127. doi:10.1109/IRI.2007.4296608.

TechNet, M. (2006). Functional Comparison of UNIX and Windows. In *Functional Comparison of UNIX and Windows*. (pp. 23–79). Microsoft Press.

TIOBE (2010). *TIOBE Index for January 2010*. doi:http://www.tiobe.com/content/paperinfo/tpci/index.html.

Weiss, G. J. (2009). *The Great Virualization Delimma of Next Decade: What You Need to Know*. doi:http://www.gartner.com/DisplayDocument?doc_cd=164938&ref=g_BETAnoreg.

Wilson, B. and Beken, T. V. d. (2003). *Observations on automation in cross-platform migration*. doi:http://soft.vub.ac.be/FFSE/Workshops/ELISA-submissions/08-Wilson-position.pdf.

Wojtczyk, M. and Knoll, A. (2008). *A Cross Platform Development Workflow for C/C++ Applications*.

Xu, L., Xu, B., Nie, C., Chen, H. and Yang, H. (2003). A Browser Compatibility Testing Method Based on Combinatorial Testing. In Lovelle, J., Rodrguez, B., Gayo, J., del Puerto Paule Ruiz, M. and Aguilar, L. (Eds.) *Web Engineering*. (pp. 310–313). *Lecture Notes in Computer Science*, vol. 2722. Springer Berlin / Heidelberg. Retrievable at http://dx.doi.org/10.1007/3-540-45068-8_60.