Size estimation for information systems from their conceptual data models

Zhao, Yuan

2007

Zhao, Y. (2007). Size estimation for information systems from their conceptual data models.
Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/46969>

<https://doi.org/10.32657/10356/46969>

Nanyang Technological University

Downloaded on 26 Apr 2023 22:17:50 SGT

9478155

Size Estimation for Information Systems From their Conceptual Data Models

Zhao Yuan

School of Electrical & Electronic Engineering

A thesis submitted to the Nanyang Technological University
in FULFILMENT of the requirement for the degree of
Doctor of Philosophy

2007

QA
76.76
.E93
Z63
2007

Statement of Originality

I hereby certify that the work embodied in this thesis is the result of original research and has not been submitted for a higher degree to any other University or Institution.

08 - April - 2007

Date

Zhaoyuan

Zhao Yuan

Acknowledgements

I would like to thank my supervisor, Associate Professor Hee Beng Kuan Tan, for his valuable advice throughout the whole course of my research.

I would like to thank my husband, Wei Zhang, for his consistent support and my lovely daughter, Minyue Zhang, for giving me immense motivation in my PhD study. I would also like to thank my parents and parents in law, without their support, this thesis would not be possible.

I would also like to thank final year BEng students, Han Chong Tan, Pua Chyuan Koh, Tan Chye Yang, Yeow Meng Ng, Chiang Fong Lee and Teck Leong Ng, MSc student, Wang Di, for their assistance in the research project.

Summary

Effort and cost estimation is crucial in software management. Estimation of software size plays a key role in the effort and cost estimation. Source Lines of Code (SLOC or LOC) is still a commonly used software size measure. Despite the fact that software sizing is well recognized as an important problem for more than two decades, there is still much problem in existing methods.

The research in this thesis explores a software size estimation model at the early stage of development cycle based on the conceptual data model. Conceptual data model is widely used in the requirements analysis for information systems. It is also not difficult to construct conceptual data models at the early stage of developing information systems. Many characteristics of an information system are actually reflected from its conceptual data model.

Based on the measurement taken from the conceptual data model for estimation, three common techniques, Multiple Linear Regression (MLR), Support Vector Machine (SVM) and Artificial Neural Network (ANN), are

used to build the estimation models from samples collected from both the Open-Source projects and the industry projects. The evaluation associated with the model building indicates that the proposed method could obtain the good results in software size estimation. We also analyze the threats to the validation of the experiments.

The Function Point (FP) method is a well known method for software sizing including the estimation of LOC. Comparison of the proposed methods with the FP method is carried out in this thesis. The comparison results show that the proposed methods are comparable with the FP methods.

The estimation of software size plays a key role in the project effort and cost estimation. Many techniques are available for estimating project effort and cost from estimated software size. This thesis also discusses the use of the proposed method for project effort and cost estimation through these methods.

Table of Contents

Acknowledgements.....i

Summaryii

Table of Contentsiv

List of Figuresvii

List of Tablesix

1 Introduction 1

1.1 Motivation..... 1

1.2 Objective 2

1.3 Major Contribution of the Thesis 3

1.4 Organization of the Thesis 4

2 Related Work.....6

2.1 Software Estimation..... 6

2.2 Software Size Estimation 10

2.2.1 Sizing Measures..... 11

2.2.2 Current Sizing Model 13

2.2.3 Problems in the Existing Sizing Models..... 18

2.3 Software Cost Estimation Methods..... 19

2.4 Summary 21

3 Conceptual Data Model for Sizing KLOC 22

3.1 Data Modeling 22

3.2 Entity-Relationship Diagram 24

3.3 Class Diagram..... 26

3.4 Proposed Conceptual Data Model..... 29

3.4.1 Our observation 30

3.4.2 The Proposed Software Size Model..... 31

4 Estimating KLOC Using Multiple Linear Regression..... 33

Size Estimation for IS from their CDM

Table of Contents

4.1	Multiple Linear Regression.....	33
4.2	The proposed Multiple Linear Regression Model.....	35
4.3	Model Building and Validation.....	36
4.3.1	Data Analysis Tool SAS.....	36
4.3.2	Statistical Tests and Criteria for Model Building	36
4.3.3	Model Evaluation	38
4.4	Experimental Result.....	39
4.4.1	Industry VB-based Systems.....	39
4.4.2	Open-Source PHP-based Systems	43
4.4.3	Java-based Systems	48
4.5	Summary of the Results	63
4.6	Threats to Validity	63
4.7	Summary	65
5	Estimating KLOC Using Support Vector Machine	66
5.1	Support Vector Machine	66
5.1.1	Support Vector Machine.....	66
5.1.2	Support Vector Regression	70
5.2	Model Building and Evaluation	72
5.2.1	Model Building.....	72
5.2.2	Evaluation.....	73
5.3	Experimental Results	74
5.3.1	Industry VB-based Systems.....	75
5.3.2	Open-Source PHP-based Systems	75
5.3.3	Java-based Systems	76
5.4	Threats to Validity	79
5.5	Summary	83
6	Estimating KLOC Using Artificial Neural Network	84
6.1	Artificial Neural Network	84
6.1.1	Overview	84
6.1.2	Neural Model and Network Architecture	85
6.2	Model Building and Evaluation	89
6.2.1	Rationale for Selecting the Neural Network Architecture	90
6.2.2	Neural Network Construction.....	91
6.2.3	Validation Criteria	91
6.3	Experimental Results	92
6.3.1	Industry VB-based Systems.....	92
6.3.2	Open-Source PHP-based Systems	93
6.3.3	Java-based Systems	95
6.4	Threats to Validity	99
6.5	Summary	100
7	Comparison of the Proposed Method with the FP Method.....	102
7.1	Function Point Analysis.....	102
7.2	Comparison of the Proposed Method with the FP Method	105

Size Estimation for IS from their CDM

Table of Contents

7.2.1	Comparison Based on PHP Open-Source Systems	105
7.2.2	Comparison Based on Java Open-Source Systems.....	111
7.3	Comparative Discussion	114
8	Application of the Proposed Estimation Models	116
8.1	Software Cost Estimation	116
8.2	Major Software Cost Estimation Models.....	117
8.2.1	Models proposed before 1970s	117
8.2.2	Models Proposed in 1970s.....	118
8.2.3	Models Proposed in 1980s.....	119
8.2.4	Models Proposed in 1990s and Later.....	120
8.3	Application of the Proposed Estimation Models.....	121
8.3.1	Application in COCOMO II Model.....	122
8.3.2	Application in SLIM Model	124
8.3.3	Application in SEER-SEM Model.....	126
8.3.4	Application in Commercial Software Cost Estimating Tools.....	127
8.4	Summary.....	128
9	Conclusion.....	130
9.1	Summary of the proposed model	130
9.2	Limitation of Research Work.....	132
9.3	Future Work and Recommendations.....	133
	Author's Publications	134
	Bibliography	136
	Appendices	149

List of Figures

Figure 2-1 A product Life Cycle8

Figure 2-2 Typical size variation during successive re-estimates..... 10

Figure 3-1 Fundamental steps in database design.....23

Figure 3-2 A generic class diagram showing a single class.....27

Figure 3-3 An example of a class diagram.....29

Figure 4-1 The SAS printout for VB-based LOC estimation model.....40

Figure 4-2 The collinearity diagnostics for VB-based LOC estimation41

Figure 4-3 The residual test for VB-based LOC estimation model42

Figure 4-4 The SAS printout for PHP-based LOC estimation model.....44

Figure 4-5 The collinearity diagnostics for PHP-based
LOC estimation model45

Figure 4-6 The residual test for PHP-based LOC
estimation model model46

Figure 4-7 The SAS printout for Industry Java-based
LOC estimation model49

Figure 4-8 The collinearity diagnostics for Industry
Java-based LOC estimation model50

Figure 4-9 The residual test for Industry Java-based
 LOC estimation model51

Figure 4-10 The SAS printout for Open-source Java-based
 LOC estimation model53

Figure 4-11 The collinearity diagnostics for Open-source
 Java-based LOC estimation model54

Figure 4-12 The residual test for Open-source Java-based
 LOC estimation model55

Figure 4-13 The SAS printout for combined Java-based
 LOC estimation model57

Figure 4-14 The collinearity diagnostics for Combined
 Java-based LOC estimation model60

Figure 4-15 The residual test for Combined Java-based
 LOC estimation model61

Figure 5-1 (a) A shyperplane separation with small margin
 (b) A hyperplane separation with larger margin67

Figure 6-1 A single-input Neuron86

Figure 6-2 A multiple-input Neuron87

Figure 6-3 A multiple-layer Neural Network.....88

Figure 8-1 Rayleigh distribution curve125

Figure E-1 Illustration of experimental design174

List of Tables

Table 4-1 The MLR validation results for VB-based	
LOC estimation model	43
Table 4-2 The MLR validation results for Open-Source	
PHP-based LOC estimation model	47
Table 4-3 The MLR validation results for Industry Java-based	
LOC estimation model	52
Table 4-4 The MLR validation results for Open-source Java-based	
LOC estimation model	56
Table 4-5 The MLR validation results for combined Java-based	
LOC estimation model	62
Table 5-1 The SVR validation for VB-based LOC estimation model	76
Table 5-2 The SVR validation for Open-Source PHP-based	
LOC estimation model	77
Table 5-3 The SVR validation for Industry Java-based	
LOC estimation model	78
Table 5-4 The SVR validation for Open-Source Java-based	
LOC estimation model	80

Size Estimation for IS from their CDM *List of Tables*

Table 5-5 The SVR validation for Combined Java-based
LOC estimation model.....81

Table 6-1 The ANN validation for VB-based LOC estimation model93

Table 6-2 The ANN validation for Open-Source PHP-based
LOC estimation model.....94

Table 6-3 The ANN validation for Industry Java-based
LOC estimation model.....96

Table 6-4 The ANN validation for Open-Source Java-based
LOC estimation model.....97

Table 6-5 The ANN validation for Combined Java-based
LOC estimation model.....98

Table 7-1 The FP data for the model building dataset for
Open-Source PHP-based systems 106

Table 7-2 The FP data for the validation dataset for
Open-Source PHP-based systems 107

Table 7-3 The FP validation dataset result of using linear regression
for Open-Source PHP-based systems 109

Table 7-4 The FP validation dataset result of using SVR for
Open-Source PHP-based systems 110

Table 7-5 The FP validation dataset result of using NN for
Open-Source PHP-based systems 112

Table 7-6 The FP data for the validation dataset for
Open-Source Java-based systems 113

Table E-1 The dataset for building VB-based LOC estimation model.....182

Table E-2 The validation dataset for VB-based LOC estimation model182

Table E-3 The dataset for building Open-Source PHP-based
LOC estimation model.....183

Table E-4 The validation dataset for Open-Source PHP
LOC estimation model.....184

Table E-5 The dataset for building Industry Java-based

<u>Size Estimation for IS from their CDM</u>	<i>List of Tables</i>
LOC estimation model.....	185
Table E-6 The validation dataset for Industry Java-based	
LOC estimation model.....	185
Table E-7 The dataset for building Open-Source Java-based	
LOC estimation model.....	186
Table E-8 The validation dataset for Open-Source Java-based	
LOC estimation model.....	187

Chapter 1 Introduction

1.1 Motivation

Estimating the required effort and cost for a software project is crucial (Boehm, et al., 2000; Canfora, et al., 2004; Jeffery and Walkerden, 1999; Ruthe, et al., 2003). A good size estimate would help in making accurate estimation of development time. Overestimation may lead to the abortion of essential projects or loss of projects to competitors. Underestimation may result in huge financial losses. It is also likely to affect the success and quality of projects adversely (Boehm and Fairley, 2000).

Despite the existence of well established software sizing methods such as Function Point method (Albrecht and Gaffney, 1983; Garmus and Herron, 2000) and the more recent Full Function Point method (COSMIC, 1999; Abran and et al., 2003), many practitioners and project managers continue to produce estimates based on ad-hoc or so called expert approaches (Armour, 2002). This is mainly due to the fact that existing sizing methods require much implementation information that is not available at the early stages of a project.

However, the accuracy of ad-hoc and expert approaches also has many problems that result in questionable project budgets and schedules.

The Entity-Relationship (ER) model (Chen, 1976) is generally regarded as the most widely used tool for the conceptual modeling of information systems. An ER model is constructed to depict the ideal organization of data, independent of the physical organization of the data and where and how the data is used. An ER model is specified in a diagram called an ER diagram.

Class diagram (Booch, et al., 2005; Bell, 2003) is an evolution of ER diagram. The purpose of the class diagram is to show the static structure of the system being modeled. Class diagram specifically shows the entities in the system along with each entity's internal structure and relationships with other entities in the system. Simplified class diagram which excludes operations can be used for conceptual data modeling. Indeed, many of the requirements of information systems can be reflected from their conceptual data models that depict their data conceptually. Thus the size of the source code for information systems can be suitably estimated based on their conceptual data models and this leads to the motivation of this research.

1.2 Objective

Software size estimation plays an important role in software management and development. Estimating software size at the early stage of software lifecycle is a key issue in software size estimation. This project is to explore on a software size estimation model that can be used at the early stage of the software lifecycle.

This thesis will explore on the use of conceptual data model for estimating software size through the use of three common approaches in software

estimation. Empirical validation for these methods will also be carried out. All the experiments data are collected from both the industry and the Open-Source systems. The approaches used in this project are Multiple Linear Regression (MLR), Support Vector Machine (SVM) and Artificial Neural Network (ANN).

1.3 Major Contribution of the Thesis

This research proposes a novel software size estimation model based on the conceptual data model. Three approaches, Multiple Linear Regression, Support Vector Machine and Artificial Neural Network, are used in model building. The empirical validation results support the LOC estimation model built through these methods strongly. The proposed model is comparable with the well-known software sizing method, Function Point Analysis. It has the advantage over the Function Point method on using the information that are readily available at the early stage of software development. The novel software size estimation model proposed in this thesis can be used in playing role in existing methods for software cost and effort estimation.

The research comprises of the following:

- Model construction: Identify the proper variables to characterize the conceptual data model to construct software size estimation model from information that is available at the early stage of software development;
- Data collection: After model construction, experiments will be needed to test the model proposed. Thus sufficient experiment data are required to be collected for both model building and model validation;

- Model building: Each approach will be used to build a model. The model will be checked against the standard statistical criteria;
- Model validation: Empirical validation will be carried out to show whether the model built in the project is acceptable and useful;
- Comparison: It is necessary to compare the proposed model with the well-known model or method to evaluate the proposed model for estimating the software size at the early stage of software development;
- Application: Software size is an important parameter in most of the software cost estimation models. The application of the proposed software size estimation for the estimation of software cost using the existing methods will be investigated and discussed.

1.4 Organization of the Thesis

In this chapter we presented the motivation of proposing a new model for software sizing and the objective for this. Major contribution is also stated. The remaining chapters of this thesis are organized as following:

Chapter 2 reviews the related research on software size estimation models and methods. Problems in the existing models and methods are discussed. Existing methods of software cost estimation is also discussed.

Chapter 3 proposes a new estimation model to estimate software size based on the conceptual data model. The simplified class diagram that excludes operations is used for conceptual data modeling. The conceptual data model used in this research refers to such class diagram that models the entities and concepts in an information system and the relationships between them.

Chapter 4 presents the experimental design in the construction of software size estimation models. Data collection from both Open-Source and industry projects is also stated. These systems for data collection are in a variety of domains and developed in several programming languages.

Chapter 5 proposes a novel method for software size estimation from the conceptual data model through the use of Multiple Linear Regression. The statistical analysis for the resulting models is presented. The validation of the model and the threats of validity are also discussed.

Chapter 6 proposes a method for software size estimation from the conceptual data model through the use of Support Vector Machine. The common criteria are used to validate the resulting models. The threats of validity are also discussed.

Chapter 7 proposes a method for estimating LOC for the information systems from its conceptual data model through the use of Artificial Neural Network. Validations have been done on the resulting models. The threats of validity are also discussed.

Chapter 8 presents a comparison of the proposed methods with the Function Point (FP) method. FP method is a well-known method for software sizing including the estimation of LOC. The result shows there is no significant difference between the new model proposed and the FP method.

Chapter 9 discusses the application of the proposed methods in software cost estimation. Software size estimation is very important in the project effort and cost estimation. In this chapter, we discuss the use of the proposed sizing method for project effort and cost estimation through the existing software cost estimation methods.

Chapter 10 concludes the thesis. It also recommends some further research.

Chapter 2 Related Work

2.1 Software Estimation

Software has been the most labor-intensive product of the twentieth century and the most error prone. The software industry has a well-known impression as being out of control in terms of schedule accuracy, cost accuracy, and quality control (Jones, 2000). A majority of large systems run late, exceed their budgets, and may be cancelled without ever reaching completion. According to a study done by the Standish Group International, their 1995 Chaos Report on information technology (IT) project failures indicated that 40 percent of all application development projects were canceled before completion (Gray and MacDonell, 1997).

How does the project managers know what they are producing meets user requirements, schedules, and budget? While at the same time, how do the customers realize the value of their expenditures for software? System analysts and project managers have always been challenged with determining a well-defined project scope, schedule and cost. Thus, software estimation plays a very important role in the software development and the success of

Size Estimation for IS from their CDM

Related Work

software projects. Software development companies or internal information system departments can utilize their resources in a cost effective way only if they can estimate the size and effort of software required. Likewise, only with proper estimation, customers can evaluate what functionality they need at what cost they are willing to pay.

A correct estimation will inevitably contribute to the business goal by avoiding costly failures and overrun. This can be accomplished by viewing the software project in the context of its product line. Putnam describes the Product Life Cycle (Putnam and Fitzsimmons, 1979) as the life of eight sequential phases: conceptual planning, product definition, system design, subsystem design, implementation, product qualification, product maintenance and manufacturing, product termination. In Figure 2-1, Putnam describes various stages or phases of software estimations according to the Product Life Cycle. It makes the estimation at the different stages of the software development very clearly.

Software estimation comprises of two main areas: software cost and software size. An estimating process consists of a large number of tasks of undetermined complexity. It is not a mechanical process which is concrete, visible and measurable by simple means and of finite quantity. The measurement of estimating process has to base on probabilistic principle. To do a good estimation, the estimating activity must be based on well-established and tested estimation methods. Thus, a set of rules that are universally recognized is actually necessary to judge whether an estimation method is reliable.

Figure 2-1 A product Life Cycle

Putnam states the rules (Putnam and Fitzsimmons, 1979) that can measure the success of an estimating method:

Size Estimation for IS from their CDM

Related Work

- The early estimate is within $\pm 30\%$ of the actual final size and cost: this is the best accuracy currently can be obtained at the early stage of the development.
- The method allows refinement of the estimate throughout the whole software development process. A higher accuracy should be achieved by monitoring and re-estimate the development each time when more information is available.
- The method is easy to use for an estimator. This enables a quick re-estimate whenever it is necessary; for example, during a progress meeting, the evaluation of estimates for alternatives can be taken strategic decisions.
- The rules are understood by everybody concerned. Management feels more comfortable when the estimating procedures are easily understandable.
- The method is well supported by tools and documented. The availability of tools increases the effectiveness of the method, mainly because results can be obtained more quickly and in a standard fashion.
- The estimation process can be trusted by software development groups and their management. This helps in gaining the participation of everybody concerned with the estimation.

A range of size estimation at the different estimation phases is given in Figure 2-2. Since software size estimation is an iterative process, the dotted lines in Figure 2-2 represent the worst case which there is no change in the requirements. It shows there is rarely possible to establish an accurate estimation at the early stage of the software development cycle. However with

more information collected from the process of software development, the accuracy of the successive estimations is improved.

Figure 2-2 Typical size variation during successive re-estimates (Putnam and Fitzsimmons, 1979)

2.2 Software Size Estimation

The application of the principles and methods of software engineering principles requires the measurement of the essential factors of the software development process. One basic and critical factor is the software size. Software size estimation plays a key role in software cost and effort estimation. However, the estimation of software size is still a very difficult problem. Dr. Barry Boehm states, “The biggest difficulty in using today’s algorithmic software cost models is the problem of providing sound size estimates” (Boehm, B. W., 1986). Because software estimation is a major

weakness in software estimation chain, Boehm also included that software sizing in his list of seven outstanding research issues in software engineering economics (Boehm, 1984). More detailed discussion on software size estimation can be found in (Fenton, 1996).

2.2.1 Sizing Measures

Estimating the size of software is still a very difficult problem. Currently, size measures can be classified into two types: length and functionality. The two most important sizing measures are source lines of code (SLOC) and Function Point (FP).

2.2.1.1 Source Lines of Code

When the software industry began in the early 1950s, the first method developed for measuring the size of a software to be developed was the method termed Source Lines of Code (SLOC). Najberg states (Najberg and Andrew, 1984), "Although alternatives to lines of code have been proposed over the years, it appears that lines of code will remain the standard for cost estimation purposes". SLOC is still the most widely used size measure today. It is a common measure and is usually one of the critical independent variables in estimating project effort and cost (Dolado, 1997). Some practitioners sometimes prefer a larger unit, the KLOC, meaning one thousand LOC. This approach has the advantage of being simple and easy to use although it is tedious. In fact, SLOC counters for performing this can be easily written.

There are two major types of SLOC measures: physical SLOC and logical SLOC. Specific definitions of these two measures vary, but the most common definition of physical SLOC is a count of "non-blank, non-comment lines" in

the text of the program's source code. Logical SLOC measures attempt to measure the number of "statements". The definitions of statements are specific to computer languages (for example, logical SLOC measure for C-like languages measures the number of statement terminating with semicolons).

It is much easier to create tools that measure physical SLOC, and physical SLOC definitions are easier to explain. However, physical SLOC measures are sensitive to logically irrelevant formatting and style conventions, while logical SLOC is less sensitive to formatting and style conventions. Unfortunately, SLOC measures are often stated without giving their definition, and logical SLOC can often be significantly different than physical SLOC.

2.2.1.2 Function Points

Function point analysis (FPA) was developed by Allan Albrecht of IBM. It was first published in 1979 (Albrecht, 1979). In 1984, the International Function Point Users Group (IFPUG) was set up to clarify the rules, set standards, and promote their use and evolution. FPA provides a standardized methodology for measuring the size of a software application in terms of a standard set of types of functions.

Function Points (Garmus and Herron, 2000) are a unit of measure that represents the work products of software developers. They are used to quantify the deliverables of the software development process. Function Points were originally developed as a communication tool for defining functional requirements in non-technical terms.

Function Point Analysis (FPA) measures a software system's size in terms of system functionality. The measurement is from the user's point of view, that is, on the basis of what the user requests and receives in return in terms of a

standard of types of functions. The function-point method is considered an empirical estimation approach due to the observed relationship between the effort required to build a system and identifiable system features, such as external inputs, external outputs, external inquiries, internal logical files and external interface files. The advantage of FPA is that it is independent of the computer language, development methodology, technology or capability of the project team used to develop the application.

2.2.2 Current Sizing Model

From 1950's till now, software size estimation has attracted significant attentions from researches. A lot of sizing models are proposed. Many techniques are applied in building the software size estimation models. These models can be mainly divided into five categories based on the techniques used: Regression-based models, Model-based models, Expertise-based models, Machine Learning-based models and other models.

2.2.2.1 Regression-based models

Regression-based models uses regression technique to perform regression analysis on historical data and conduct size estimation models which relate size to one or more input factors.

Ikatura and Takayanagi (1981) proposed a sizing model through linear regression which is built from 38 banking programs written in the COBOL language. KLOC is used to measure the software size. Six program characteristics are selected as regression variables, the number of input files, the number of input items, the number of output files, the number of

transaction type reports, the number of horizontal items in tabular reports and the number of calculating processes. The model equation is as follows:

$$ES = \sum_{i=1}^6 A_i X_i \quad (2.1)$$

where ES is the estimated software size, X_i is the i th program characteristic mentioned above, and A_i is the coefficient of the corresponding characteristic determined through regression technique.

An increasing number of organizations are collecting data and developing their own regression equations. For example, a NASA-sponsored project is collected data and developed regression models for software projects conducted using Ada (Fairley, 1992).

Recently, Lai and Huang (2003) explored a two-stage sizing model for formal communication protocol specification and its implementation at the early stage of software development. LOC is used as the size measure. Fourteen projects data used for model building and the other fourteen projects data for model validation. The technique used in model building is least squares regression. Model selection process was carried out in the model building. The estimated size is based on four integrated parts: module header, declaration, initialization, and transition.

All regression-based models are established on historical data. They are also useful early in a software development cycle only if the input factors are available. However, it is difficult to extend the models based on regression technique to the applications that outside the scope of the datasets from which they were developed.

2.2.2.2 Model-based models

Function Point Analysis, initially published by Albrecht, appears to be a popular approach to estimate software size (Garmus and Herron, 2000). A new term, Function Points (FP), was proposed to measure software size. FP can be counted based on unadjusted FP (UFP) and the valued adjustment factor (VAF). Five elements are used to calculate UFP. They are: external input (EI), external output (EO), external inquiry (EQ), internal logical file (ILF) and external interface file (EIF). Fourteen general systems characteristics (GSC) are used to count the VAF. International Function Point Users Group (IFPUG) was founded in 1984 to clarify the rules, set standards, and promote their use and evolution (Abran and Robillard, 1996).

There are many FP variations proposed based on the IFPUG FP model. These models use different elements and rules to calculate Function Points. They extended the application of FP to different software domains and overcome some weakness of IFPUG FP model. The popular FP variations are: DeMarco function points (DeMaro, 1982), Mark II Function Points (Symons, 1988), SPR Function Points (Jones, 1986), 3D Function Points (Desharnais and Abran, 1995), Full Function Points (FFP) and COSMIC-FFP (COSMIC, 1999; Abran et al., 2003).

A software sizing model for Object-Oriented Systems has also been proposed by Luiz (1990). This model is resulted from a software project cost estimation accuracy model proposed by Boehm (Boehm, 1981). In this model, Object-Oriented functional model has been adapted for decomposing system complexity.

Although proponents of the function point method claim that the information for counting is available at the early stage of software development cycle, some researchers (Reese and Tamulevicz, 1987) have stated that those information are usually difficult to obtain at that time.

As a consequence of using use-cases to model software requirements in OO approach, Use Case Point has been proposed as a method to estimate software size and effort (Arnold, 1998; Mohagheghi, 2005). However, the information such as complexities of use-cases and details of scenarios required by this method are difficult to predict in the early stage of software development.

2.2.2.3 Expertise-based models

In these models, one or more experts are consulted for their ideas regarding the size of a program or factors which affect software size. Expertise-based models and techniques may be used alone, or are sometimes used in conjunction with other techniques. For example, expert judgment could be used to determine the inputs for a regression-based model.

Program evaluation and review technique (PERT) is one of the most basic expert judgment techniques (Ferens, 1988). The software size (E) is measured by SLOC. Experts determine the smallest possible size (a), most likely size (m) and the largest possible size (b). The equation for sizing is:

$$E = \frac{a + 4m + b}{6} \quad (2.2)$$

Another model, Software Sizing Model (SSM) was developed by Bozoki (1984). SSM requires expert judgment in four areas: PERT sizing estimates, pair wise comparison, program sorting, and program ranking. SSM also requires two reference programs of known size, plus user inputs for each of the four areas.

Lambert proposed a complex expertise-based model in (Lambert, 1986). It requires prioritized ranking of projects and uses mathematical eigenfunctions to determine ranked project size.

Size Estimation for IS from their CDM

Related Work

The expertise-based model requires little or no historical data. It can be used when such data is not available. Therefore, they are useful for new programs for which historical precedents do not exist. They can also be applied at the early stage of software development. However, expertise-based models are highly dependent on personal opinions which may be subjective and biased. Also, finding a true “expert” may be difficult; even the assessments of knowledgeable personnel are sometimes mere guesses. Thus, very few researches focus on this technique in the past decades.

2.2.2.4 Machine Learning-based models

These models use machine learning techniques to estimate software size by comparing a project with a similar project or projects of known size. Machine learning technique includes analogy method, artificial neural network, fuzzy logic and etc.

Software Size Estimator (SSE) developed by aerospace Corporation (Singhal, 1986) is one of the machine learning-based models. It contains a database of past projects, primarily from space systems. These models usually require user inputs such as software type, and a complexity assessment for the software program. The model searches the database for a similar program or programs. The basic principle applied in this model is analogy analysis. Other analogy based models are Electronic Systems Division (ESD) Software Sizing Package. Quantitative Software Management (QSM) is a software sizing model based on fuzzy logic technique.

Witting developed a software estimation model using connectionist models (synonymous with neural networks) and derived very high prediction accuracies (Wittig and Finnie, 1994) for the model.

Machine learning-based models usually use actual programs for which historical data. The models in this category are also useful early in a program because minimal input data is required. However, the project being estimated must have a true historical precedent, or the estimate will be misleading. They are not useful outside of the scope of the history data from which the model is developed. In particular, the models using Neural Network are lack of explanation because neural networks operate as black boxes and do not provide any information or reasoning about how the outputs are derived.

2.2.2.5 Other models

PRICE-S (Programming Review of Information Costing and Evaluation - Software) was originally developed by Frank Freiman and Robert Park at RCA for use internally (Freiman and Park, 1979). Although PRICE-S is a tool for software cost estimation and schedule estimation, it has a sub-model, to estimate the size of the software to be developed. It requires over 15 inputs which describe characteristics of the project being estimated.

2.2.3 Problems in the Existing Sizing Models

Estimating size is already a difficult task in organizations where a software metric activity exists, but it is made particularly hard when there is no measurement baseline of previous projects. However, as estimation technique will only be meaningful if it can estimate the size of software at the early stage.

As we have discussed in previous section, except the expertise-based methods, existing software sizing models estimate software size on a number of requirements that are usually difficult to get at the initial stage of the software

development. Expertise-based models may do some estimation at the first stage of the software development cycle. However, expertise-based models are highly dependent on personal opinions which may be subjective and biased.

However, estimating software size at the early stage of software development cycle is a very important issue in software engineering. How to obtain a good estimation at that time is a key problem both in the software size estimation and the software cost estimation.

2.3 Software Cost Estimation Methods

Software cost estimation has been an important but difficult issue since the beginning of the computer era in the 1940s. As software applications have grown in size and importance, the need for accuracy in software cost and effort estimation has also grown. Software cost estimation is a major research topic now during the software development and maintenance. It is also a complex activity that requires knowledge of a number of key attributes about the project for which the estimation is being constructed. As in the discussion of using proposed software size estimation in existing software estimation methods in Chapter 9, we will introduce existing software cost estimation methods. This section will only give an overview of software cost estimation methods.

Many techniques are used to build software cost estimation models. Classifications on these software cost estimation techniques or methods are explored by several researchers, such as Boehm (1981), Conte et al. (1986), Fairley (1992), Boehm et al. (2001), Molokken et al. (2003) and etc. They classify software cost estimation methods and techniques in different way. In

this thesis, we discuss four major classes of software cost estimation methods: expert judgment, regression, theory-based, and Analogy.

- **Expert Judgment:** Expert judgment is based on the experience of one or more people who are familiar with the development of software applications similar to that currently being sized (Hughes, 1996). Estimation based on expert judgment is useful when there is little or no historical data to support the estimating process. However, this method is highly dependent on personal opinions which may be subjective and biased.
- **Regression:** It is one of the most widely used estimation techniques. Regression-based software cost estimation models are constructed by collecting data on completed systems and conducting regression equations that characterize the relationships among the variables. The models using this technique are useful early in a project if those variables are available. The most popular software cost estimation models used regression method are COCOMO II model and Function Point model.
- **Theory-based:** Theory-based software cost estimation models are derived from underlying theoretical considerations that characterize some aspects of the software development process. Theory-based estimation models have the advantage of considering many different project factors in estimation. However, these factors may be difficult to obtain at the early stage of the software development cycle. SLIM model (Putnam and Fitzsimmons, 1979) and Price S model (Freiman and Park, 1979) are two main theory-based models.
- **Analogy:** Estimation by analogy is a type of Case Based Reasoning (CBR) where cases are defined as abstractions of events that are

limited in time and space. The main idea of analogy technique is to compare a project with a similar project or projects of known cost. ANGEL (Shepperd, et al., 1997) is the typical tool that used analogy technique. The models using this technique are useful early in a project because minimal input data is required. However, the project being estimated must have a true historical precedent, or the estimation will be misleading. They are also not useful outside of the scope of the history data from which the model is developed.

2.4 Summary

Software size estimation is an important issue in software engineering. In this chapter, software size measurement and assessment rules are presented. Many techniques are applied in software size estimation model. They can be divided into five categories: Regression-based models, Model-based models, Expertise-based models, Machine Learning-based models and other models. Advantages and disadvantages of each type of the models were discussed. The problems of existing models were also pointed out. This motivates the main objective of this thesis. As software size estimation is the key issue in software cost estimation, the techniques used for software cost estimation are also discussed in this chapter.

Chapter 3 Conceptual Data Model for Sizing KLOC

3.1 Data Modeling

A data model (utexas, 2002) is a plan for building a database. It is a conceptual representation of the data structures that are required by a database. The data structures include the data objects, the associations between data objects, and the rules which govern operations on the objects. As the name implies, the data model focuses on what data is required and how it should be organized rather than what operations will be performed on the data. To use a common analogy, the data model is equivalent to an architect's building plans.

A data model is independent of hardware or software constraints. Rather than trying to represent the data as a database would see it, the data model focuses on representing the data as the user sees it in the "real world". It serves as a bridge between the concepts that make up real-world events and processes and the physical representation of those concepts in a database.

Database design (Teory and Toby, 1994) can be defined as the design of the logical and physical structure of one or more databases to accommodate the information needs of the users in an organization for a defined set of applications. The fundamental phases in database design and analysis are depicted in Figure 3-1.

Figure 3-1 Fundamental steps in database design

The data model is one part of the conceptual model design process. The other, typically is the functional model. The data model gets its input from the real world (during the planning and analysis stage) and focuses on what data should be stored in the database while the functional model deals with how the data is processed. To put this in the context of the relational database, the data model is used to design the relational tables to form the logical model. The functional model is used to design the queries which will access and perform operations on those tables.

The next step in the database design, as indicated above, is converting this model into a logical model. The known logical models of database are hierarchical, network, relational, object oriented.

The physical model describes the management of physical data storage along with the optimal physical database layouts for any number of disks. These

layouts will be the result of understanding the ways in which various database files operate and interact (Loney, 1994).

3.2 Entity-Relationship Diagram

The Entity-Relationship (ER) model was originally proposed by Chen (1976) as a way to unify the network and relational database views. Simply stated the ER model is a conceptual data model that views the real world as entities and relationships. A basic component of the model is the Entity-Relationship diagram which is used to visually represent data objects.

The ER Diagram originally used in the database domain is commonly used in systems analysis. It is used to specify relationships between entities in the system. These relationships can lend insight into the requirements and/or design of a system. In an ER Diagram, there are three basic elements:

- Entities are the “things” about which we seek information.
- Attributes are the data we collect about the entities.
- Relationships provide the structure needed to draw information from multiple entities.

Entities are the principal data object about which information is to be collected. An entity may be concrete (such as a person or a book) or abstract (like a holiday or a concept).

An entity is represented by a set of attributes. Formally, an attribute is a function that maps an entity set into a domain. Every entity is described by a set of (attribute, data value) pairs.

A relationship is an association between two or more entities. A relationship set is a set of relationships of the same type. Formally it is a mathematical relation on $n \geq 2$ (possibly non-distinct) sets. If $E_1, E_2, \dots, E_n$ are entity sets, then a relationship set R is a subset of

$$\{(e_1, e_2, \dots, e_n) \mid e_1 \in E_1, e_2 \in E_2, \dots, e_n \in E_n\}$$

where $(e_1, e_2, \dots, e_n)$ is a relationship.

An ER Diagram may define certain constraints to which the contents of a database must conform. Mapping Cardinalities is that it can express the number of entities to which another entity can be associated via a relationship. For relationship sets between entity sets A and B, the mapping cardinality must be one of the following:

- One-to-one: An entity in A is associated with at most one entity in B, and an entity in B is associated with at most one entity in A.
- One-to-many: An entity in A is associated with any number in B. An entity in B is associated with at most one entity in A.
- Many-to-one: An entity in A is associated with at most one entity in B. An entity in B is associated with any number in A.
- Many-to-many: Entities in A and B are associated with any number from each other.

The appropriate mapping cardinality for a particular relationship set depends on the real world being modeled.

3.3 Class Diagram

Management Group (OMG) established the Unified Modeling Language (UML) in 1997. UML can provide the way to specify, visualize and document models of software systems, including their structure and design. UML brought forth a unified standard modeling notation that has been accepted by the industry as the standard modeling language.

Class diagram is one of the standard UML diagrams. It shows how the different entities (people, things, and data) relate to each other (Bell, 2003). The purpose of the class diagram is to show the static structure of the system being modeled. The class diagram specifically shows each entity's internal structure along with the relationships that the entity has to other entities in the system. Only types of entities are shown on a class diagram because class diagrams only model the static structure of a system. A class diagram can be used to represent logical classes, which are typically the kinds of things the business people in an organization talk about. Class diagrams can also be used to show implementation classes, which are the things that programmers typically deal with. An implementation class diagram will probably show some of the same classes as the logical classes diagram.

A class diagram mainly consists of the following features:

- **Class:** It represents the classes in the system and contain the information about the class name, attributes, methods and access specifies.
- **Relationship:** It models the relationships between classes in the system.

A class is depicted as a rectangle with three horizontal sections as shown in Figure 3-2. The upper section shows the name of the class; the middle section

contains the attributes of the class; and the lower section contains the operations or methods of the class.

Figure 3-2 A generic class diagram showing a single class

In a class diagram, there are mainly two types of relationships: inheritance and association. Inheritance is an important concept in object-oriented design which refers to the ability of sub-class to inherit the identical functionality of super-class and then add new functionality of its own. To model inheritance on a class diagram, a solid line is drawn from the sub-class with a closed arrowhead pointing to the super-class.

Association relationship is a linkage between two classes. There are five types of association relationships in the class diagram:

- Bi-directional (standard) association: It indicates that both classes linked by this type of association are aware of their relationship and of the other class. A bi-directional association is drawn by a solid line between the two classes. At either end of the line, a role name and a multiplicity value may be placed.
- Uni-directional association: This relationship shows that two classes are related, but only one class knows that the relationship exists. It is drawn as solid line with an open arrowhead pointing to the latter

class. The uni-directional association only contains the role name and multiplicity value for this class.

- Basic aggregation: It indicates that one class is a subordinate class (or a part) of another class. In an aggregation relationship, the child class instance can outlive its parent class. It is drawn as a solid line from the parent class to the subordinate class and an unfilled diamond shape on the parent class's association end.
- Composition aggregation: It is just another form of the aggregation relationship, but the child class's instance lifecycle is dependent on the parent class's instance lifecycle. The composition relationship is drawn like the basic aggregation relationship except the filled diamond shape.
- Reflexive association: This is an association that associates the class to itself.

Figure 3-3 shows an example of a class diagram. It includes one inheritance relationship and two association relationships. The CDSalesReport class inherits from the Report class. A CDSalesReport is associated with one CD, but the CD class doesn't know anything about the CDSalesReport class. The CD and the Band classes both know about each other, and objects of both classes can be associated to one or more objects of the other other.

Figure 3-3 An example of a class diagram

3.4 Proposed Conceptual Data Model

Software sizing is still an important problem. In the information system development field it is important that software size estimation can be made early in its development. The project is to explore in the estimation of thousand Lines of Code (KLOC) for the information systems from its conceptual data model.

Information systems constitute one of the largest software domains. These systems usually maintain large amount of structured data in a database built using a database management system (DBMS). They provide operational, control and management support to end-users through referencing and analyzing these data. The support is usually accomplished through accepting inputs from user, processing inputs, updating databases, printing of reports, and providing inquiries to help users in the management and decision making processes.

3.4.1 Our observation

The size of an information system is well characterized by its conceptual model. In fact, the derivation of program structure from the structure of the data that the program processes in Jackson structured program design method (JSP) has already implied this characteristic (Burgess, 1988).

The entity-relationship (ER) model is generally regarded as the most widely used tool for the conceptual data modeling of information systems (Teorey, 1986; Ghezzi, 2003). An ER model is constructed to depict the ideal organization of data, independent of the physical organization of the data and where and how data are used. An ER model is specified in a diagram called an ER diagram.

Today, most of the information systems are developed using UML (Blaha and Premerlani, 1998). Class diagram is an evolution of ER diagram. From the preview introduction of ER diagram and class diagram, ER diagram is equivalent to simplified class diagram that excludes operations. In this thesis, we use the simplified class diagram that excludes operations instead of ER diagram for conceptual data modeling. The term conceptual data model shall refer to such class diagram that models the entities and concepts in an information system and the relationships between them. The information based on the conceptual data model can be readily available at the early stage of software development. Although there are many types of relationships in a class diagram, we shall not further classify its relationship types into sub-types such as inheritance, associations, and aggregations in a conceptual data model.

3.4.2 The Proposed Software Size Model

The proposed software size estimating method is based on the conceptual data model for the target information system. In the proposed approach, a separate software size model should be built for each programming language or development environment used. This due to the fact that the size of source code also depends on programming language or development environment used. For example, different software size models should be built for systems written in Visual Basic with SQL, and systems written in Java with JSP, HTML and SQL. As class diagrams constructed in most projects in the industry do not classify relationship types, we propose the following three parameters to characterize the conceptual data model:

- 1) C : the total number of classes in the conceptual data model.
- 2) R : the total number of relationships in the conceptual data model.
- 3) $\bar{A}$: the average number of attributes per class, that is, $\bar{A} = A / C$, where A is the total number of attributes in the conceptual data model, and C is the total number of classes.

The model parameters should be defined according to the conceptual data model constructed during the early stage of software development. In the worst case, these parameters will at least be available at the early stage of requirements analysis.

The steps for building proposed software size models are as follows:

- 1) Data collection: This is to collect samples of information systems from which conceptual data model and their size are available or can be computed.

- 2) Model building and evaluation: This is to build the software size estimation models based on estimation methods. Statistical criteria are used for validating the resulting models. In this research, three techniques, multiple linear regression (MLR), support vector machine (SVM) and artificial neural network (ANN), are used to build software sizing models.

Chapter 4 Estimating KLOC Using Multiple Linear Regression

4.1 Multiple Linear Regression

Among the existing techniques used in software estimation, regression-based techniques are the most popular ways of building models. Through comparing estimation techniques including regression, neural networks, Case-Based Reasoning, Rule induction, Mair (Mair, et al., 1999) shows that regression and Case-Based Reasoning perform better than the other techniques. Dolado (1999) also compares the methods used in regression, neural networks and genetic programming. He concludes that although the estimations of regression can be improved by neural networks and genetic programming, the results are not very significant.

Multiple regression analysis is a statistical technique that can be used to analyze the relationship between a single response variable and multiple predictor variables. The objective of multiple regression analysis is to specify the predictor variables in a mathematical equation that will estimate the response variable. Each predictor variable is weighted. The weights denote a

Size Estimation for IS from their CDM Estimating KLOC Using MLR

variable's relative contribution to the overall prediction of the response. The multiple linear regression model assumes that there is a linear relationship between the dependent variable and each predictor. The relationship is described in the following formula:

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip} + e_i \quad (4.1)$$

where

y_i is the value of the i^{th} case of the dependent scale variable;

p is the number of predictors;

β_j is the value of the j^{th} coefficient that act as weight of the j^{th} predictor,

$j = 0, 1, \dots, p$;

x_{ij} is the value of the i^{th} case of the j^{th} predictor;

e_i is the error in the observed value for the i^{th} case.

The model is linear because increasing the value of the j^{th} predictor by 1 unit increases the value of the dependent variable by β_j units. Note that β_0 is intercept, the model-predicted value of the dependent variable when the value of every predictor is equal to 0.

The accuracy of the estimation can be assessed graphically. It is a visual representation of missed predictions. To measure the goodness of fit of the model, we use statistical measures. Coefficient of multiple determination (R^2) is a statistical indicator that compares the accuracy of the model to the accuracy of a trivial benchmark model where the prediction is just the mean of all of the samples. It can range from 0 to 1, and a perfect fit would result in an R-square value of 1, a very good fit near 1, and a very poor fit less than 0. It is calculated as follows:

Size Estimation for IS from their CDM *Estimating KLOC Using MLR*

$$R^2 = 1 - \frac{\sum (y - \hat{y})^2}{\sum (y - \bar{y})^2} \quad (4.2)$$

where y is the actual value for the dependent variable, $\hat{y}$ is the predicted value of y and $\bar{y}$ is the mean of the y value.

However, it is always true that R^2 will increase as more independent variables are added to the regression equation because adding variables to the equation causes the prediction errors to be smaller. Adjusting multiple coefficient of determination (R_a^2) is recommend to avoid overestimating the impact on the amount of explained variability. It is computed as:

$$R_a^2 = 1 - (1 - R^2) \frac{n-1}{n-p-1} \quad (4.3)$$

where n is the number of observations, and p is the number of parameters.

4.2 The proposed Multiple Linear Regression Model

Through much experimentation, for a programming language or environment, we propose the following multiple linear regression model to estimate the LOC for an information system developed using the programming language or environment respectively from its conceptual data model:

$$KLOC = \hat{\beta}_0 + \hat{\beta}_C C + \hat{\beta}_R R + \hat{\beta}_{\bar{A}} \bar{A} \quad (4.4)$$

The multiple linear regression model has the following three independent variables to characterize the conceptual data model:

- C : the total number of classes in the conceptual data model.

Size Estimation for IS from their CDM *Estimating KLOC Using MLR*

- R : the total number of relationship types in the conceptual data model.
- $\bar{A}$: the average number of attributes per class, that is, $\bar{A} = A/C$, where A is the total number of attributes in the conceptual data model, and C is the total number of classes.

4.3 Model Building and Validation

4.3.1 Data Analysis Tool SAS

Statistical Analysis System (SAS) provides extensive statistical capabilities with tools for both specialized and enterprise-wide analytical needs (SAS User Guide). Ready-to-use procedures handle a wide range of statistical analyses, including analysis of variance, regression, categorical data analysis, multivariate analysis, and nonparametric analysis. SAS offers a general regression procedure that uses least squares technique to estimate the parameters, includes nine different model selection methods, such as stepwise regression, and produces a variety of diagnostic measures. Linear regression analysis is carried out for the proposed sizing method.

4.3.2 Statistical Tests and Criteria for Model Building

We carried out the following tests that can be computed from most statistical packages are usually used in comprehensive building a model (Kennedy, 2003; McClave and Sincich, 2003; Neter, 1996; SAS/STAT User's Guide):

- Significant Test: This is usually based on 5% level of significant. An F-test should be done for the overall model. If the value of $(Pr > F)$ is less than 0.05, then it indicates that the overall model is useful. That is, there is sufficient evidence that at least one of the coefficients is

Size Estimation for IS from their CDM Estimating KLOC Using MLR

non-zero at 5% level of significant. Furthermore, a t-test is conducted on each $\hat{\beta}_j$ ($0 \leq j \leq k$). If all the values of $(Pr > |t|)$ are less than 0.05, then there is sufficient evidence of a linear relationship between y and each x_j ($1 \leq j \leq k$) at 5% level of significant.

- Fitness Test: If the adjusted multiple coefficient of determination R_a^2 for the model is higher than 0.75, then it implies that the least square model has explained more than 75% of the total sample variation of y values, after adjusting for sample size and number of independent variables.
- Collinearity Test: The independent variables should not be highly correlated. If the variance inflation factors (VIFs) and the condition indexes of all the independent variables are less than 10 (the threshold value), then there is no significant evident of the existence of collinearity between the independent variables.
- Extreme Case Test (Outlier): There should be no extreme cases (i.e., outliers). Extreme cases can distort coefficient estimation. For each datapoint, a residual analysis is conducted. If the absolute value of its studentized residual ($R_{Student}$) is below 2 (a usual threshold for identifying large influence of observation on the parameter estimates), then there is no significant evident of large influence to the parameter estimation caused by the datapoint (Glantz, 2001). That is, there is no significant evident that the datapoint is an extreme case. Otherwise, the datapoint should be investigated. If the datapoint itself is wrong, it should be recollected if possible, otherwise, appropriate action should be taken (P. Kennedy, 2003).

Size Estimation for IS from their CDM *Estimating KLOC Using MLR*

Any model built from a random sample should be validated against another random sample. As a rule of thumb, the size of the latter sample should not be less than half the size of the former sample.

4.3.3 Model Evaluation

We collected separate data sets that are usually collected for comprehensive model evaluation. We use mean magnitude of relative error (MMRE) and Prediction at level l (Pred(l)) (Conte, et al., 1986) that are the most commonly used measurements for validation. MMRE is defined as follows:

$$MMRE = \frac{1}{n} \sum_{i=1}^n \left| \frac{y_i - \hat{y}_i}{y_i} \right| \quad (4.5)$$

where y_i and $\hat{y}_i$ are the i th real and the estimate values respectively of a project. n is the number of data points.

Pred(l) is defined as the ratio of number of cases in which the estimates are within the l absolute limit of the actual values divided by the total number of cases. A common accepted prediction level is 0.25. In this project, Pred(0.25) is adopted. It is the ratio of number of cases in which the estimates are within the 25% of the actual values divided by the total number of cases.

The main criteria for validation lies in obtaining acceptable values of MMRE and Pred(0.25). A usual criteria for acceptable value of MMRE is less than 0.25. Lowest MMRE is preferred. And a standard criteria for acceptable value of Pred(0.25) is not less than 75%. Highest Pred(0.25) is preferred.

4.4 Experimental Result

Five pairs of datasets are used to carry out the multiple linear regression experiments. The details of experiment design are shown in Appendix E.

4.4.1 Industry VB-based Systems

The pair of datasets for VB-based systems is collected from a number of different organizations in the industry. It is collected from systems in a variety of domains including shipment management, auction management, finance, administration, logistic management, business management, medical system and donation management. Both datasets in this pair have similar composition. The dataset for model building is shown in Table E-1. From this dataset, we built the following model (Figure 4-1):

$$KLOC = -11.546 + 1.374 * C + 1.126 * R + 0.311 * \overline{A} \qquad (4.6)$$

The SAS output result of VB-based model shown in Figure 4-1 has two parts: the Analysis of Variance (ANOVA) table and the Parameter Estimates table.

Size Estimation for IS from their CDM Estimating KLOC Using MLR

The REG Procedure													
Model: MODEL1													
Dependent Variable: Size													
Analysis of Variance													
Source	DF	Sum of Squares	Mean Square	F Value	Pr > F								
Model	3	9137.59740	3045.86380	185.88	< .0001								
Error	12	196.63918	16.38650										
Corrected Total	15	9334.23659											
Root MSE		4.04804	R-Square	0.9789									
Dependent Mean		42.96088	Adj R-Sq	0.9737									
Coeff Var		9.42261											
Parameter Estimates													
Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > t	Tolerance	Variance Inflation						
Intercept	1	-11.54637	3.13934	-3.68	0.0032	0.33005	3.02971	0					
E	1	1.97361	0.17883	7.68	<.0001	0.32847	3.04443						
R	1	1.12572	0.16836	6.69	<.0001	0.97210	1.02870						
A	1	0.31077	0.14199	2.19	0.0491								

Figure 4-1 The SAS printout for VB-based LOC estimation model

The model was tested as discussed in Section 4.3.2. In significance test, $(Pr > F) < 0.0001$ and $(Pr > |t|)$ for intercept, C , R and $\bar{A}$ are 0.0032, <0.0001 , <0.0001 and 0.0491, respectively. In fitness test, $R_a^2 = 0.9737$. In collinearity test shown in Figure 4-2, the variance inflation factors (VIFs) and the condition indexes of all the independent variables are less than 10 (the threshold). In extreme case test shown in Figure 4-3, the absolute values of

Size Estimation for IS from their CDM *Estimating KLOC Using MLR*

studentized residuals (RStudents) for all the systems are below 2 (the threshold). Therefore, all the test results are affirmative.

Collinearity Diagnostics					
Number	Eigenvalue	Condition Index	Proportion of Variation		
			Intercept	β	λ
1	3.54238	1.00000	0.00779	0.00465	0.01567
2	0.34535	3.20272	0.00423	0.03026	0.46176
3	0.07500	6.82707	0.98226	0.03849	0.52254
4	0.03627	9.88325	0.00570	0.92659	0.0003568

Figure 4-2 The collinearity diagnostics for VB-based LOC estimation

Size Estimation for IS from their CDM *Estimating KLOC Using MLR*

The REG Procedure
Model: MODEL1
Dependent Variable: Size

Output Statistics

Obs	Residual	RStudent	Hat	Diag	H	Cov	Ratio	DPFITs	Intercept	E	K	K̂
1	-1.4615	-0.3794	0.1591	1.5389	-0.1650	-0.1524	0.0437	0.0107	0.0437	0.0107	0.1041	
2	0.5069	0.2208	0.7039	4.6397	0.3405	0.0805	0.2914	-0.3137	0.2914	-0.3137	-0.0712	
3	0.6402	0.1878	0.3478	2.1439	0.1371	0.0096	-0.0087	-0.0297	0.0096	-0.0297	0.0943	
4	3.2154	0.8465	0.1403	1.2799	0.3420	0.1420	-0.0036	-0.1162	0.1420	-0.1162	0.1150	
5	-5.7312	-1.6344	0.1451	0.6344	-0.6734	0.0797	0.3585	-0.4312	0.0797	-0.4312	-0.2349	
6	5.3297	1.6131	0.2449	0.8022	0.9186	-0.2908	0.1047	0.3490	-0.2908	0.1047	-0.1139	
7	-1.8105	-0.4818	0.1935	1.6154	-0.2361	-0.1543	0.1538	-0.1236	-0.1543	0.1538	0.1143	
8	-5.1146	-1.6267	0.3140	0.8716	-1.1006	0.5133	-0.8575	-0.4858	0.5133	-0.8575	-0.3973	
9	-5.4488	-1.5138	0.1243	0.7587	-0.5702	-0.4216	0.2128	-0.1308	-0.4216	0.2128	0.3462	
10	4.9264	1.3471	0.1284	0.8822	0.5170	-0.0611	-0.0213	0.2190	-0.0611	-0.0213	-0.0594	
11	0.8488	0.2637	0.4232	2.4019	0.2285	-0.0019	-0.0730	0.0259	-0.0019	-0.0730	0.1686	
12	1.0534	0.3345	0.4289	2.3817	0.2899	-0.2163	0.0544	0.0847	-0.2163	0.0544	0.1760	
13	-0.0433	-0.0115	0.2027	1.7163	-0.0058	-0.0006	0.0017	-0.0033	-0.0006	0.0017	0.0024	
14	0.2677	0.0679	0.1294	1.6241	0.0262	0.0218	-0.0124	0.0046	0.0218	-0.0124	-0.0122	
15	5.4660	1.5961	0.1920	0.7619	0.7780	0.7475	-0.1811	-0.1465	0.7475	-0.1811	-0.4292	
16	-2.6546	-0.6835	0.1203	1.3633	-0.2528	-0.1521	-0.0649	0.0728	-0.1521	-0.0649	0.1609	

Sum of Residuals

Sum of Squared Residuals

Predicted Residual SS (PRESS)

0

196.63918

307.56807

Figure 4-3 The residual test for VB-based LOC estimation model

The model was validated using the dataset shown in Table 4-1. MMRE and Pred(0.25) computed are 0.18 and 0.81 respectively. As discussed in Section 4.3.2, these values fall within the acceptable level (not more than 0.25 and not less than 75% respectively). Therefore, the evaluation result supports the validity of the LOC estimation model built.

Size Estimation for IS from their CDM *Estimating KLOC Using MLR*

Table 4-1 The MLR validation results for VB-based LOC estimation model

System No	C	R	$\bar{A}$	Actual Size (KLOC)	Estimated Size (KLOC)	MRE
1	16	8	6.188	27.217	21.370	0.22
2	7	6	16.286	14.53	9.893	0.32
3	28	24	13.179	65.872	58.049	0.12
4	24	21	14.417	41.435	49.560	0.20
5	36	37	15.000	72	84.245	0.17
6	16	16	27.563	29.52	37.026	0.25
7	31	24	10.645	52.76	61.383	0.16
8	29	24	6.931	46.92	57.480	0.23
9	42	44	33.524	97.88	106.132	0.08
10	18	12	11.000	38.764	30.119	0.22
11	12	14	8.000	31.665	23.194	0.27
12	30	32	7.533	77.52	68.049	0.12
13	10	7	9.500	16.81	13.031	0.23
14	32	26	14.781	59.332	66.295	0.12
15	41	45	16.561	107.8	100.608	0.07
16	24	18	9.125	53.66	44.536	0.17
MMRE = 0.18, Pred(0.25) = 0.81						

4.4.2 **Open-Source PHP-based Systems**

The pair of datasets for Open-Source PHP-based systems is collected from two open-source websites: SourceForge and Freshmeat (sourceforge; freshmeat). It is collected from systems in a variety of domains including content management, resource scheduling, inventory management, viewing and maintaining pictures and entertainment. Both datasets in this pair have similar composition. The dataset for model building is shown in Table E-3. From this dataset, we built the following model (Figure 4-4):

$$KLOC = -13.223 + 1.241 * C + 1.672 * R + 0.652 * \bar{A} \qquad (4.7)$$

Size Estimation for IS from their CDM *Estimating KLOC Using MLR*

In significance test, $(Pr > F) < 0.0001$ and $(Pr > |t|)$ for intercept, C , R and $\bar{A}$ are <0.0001 , <0.0001 , <0.0001 and 0.0467 respectively. In fitness test, $R_a^2 = 0.9438$. In collinearity test shown in Figure 4-5, the variance inflation factors (VIFs) and the condition indexes of all the independent variables are less than 10 (the threshold). In extreme case test shown in Figure 4-6, the absolute values of studentized residuals (RStudents) for all the systems are below 2 (the threshold). Therefore, all the test results are affirmative.

The REG Procedure													
Model: MODEL1													
Dependent Variable: Size													
Analysis of Variance													
Source	DF	Sum of Squares	Mean Square	P Value	Pr > F								
Model	3	16647	5549.07875	174.49	<.0001								
Error	28	890.45695	31.80203										
Corrected Total	31	17538											
Root MSE													
Dependent Mean													
Coeff Var													
R-Square													
Adj R-Sq													
Parameter Estimates													
Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > t	Tolerance	Variance Inflation						
Intercept	1	-13.22319	2.82015	-4.69	<.0001		0						
E	1	1.24098	0.17193	7.22	<.0001	0.30729	3.25426						
R	1	1.67241	0.28311	5.91	<.0001	0.30857	3.24079						
A	1	0.65200	0.31925	2.08	0.0467	0.99252	1.00753						

Figure 4-4 The SAS printout for PHP-based LOC estimation model

Size Estimation for IS from their CDM *Estimating KLOC Using MLR*

Number	Eigenvalue	Collinearity Diagnostics			
		Condition index	Intercept	Proportion of Variation Σ	$\bar{\lambda}$
1	3.49000	1.00000	0.00916	0.0073	0.0073
2	0.37395	3.03496	0.05651	0.06571	0.07515
3	0.07749	6.71100	0.90639	0.04569	0.00316
4	0.03855	7.72023	0.02793	0.88087	0.91383
					0.01075
					0.14316
					0.80988
					0.03621

Figure 4-5 The collinearity diagnostics for PHP-based LOC estimation model

Size Estimation for IS from their CDM Estimating KLOC Using MLR

The REG Procedure Model: MODEL1 Dependent Variable: Size											
Output Statistics											
Obs	Residual	RStudent	Hat	Diag	R	Cov	Ratio	DPFITs	Intercept	DPFITs	R
1	-0.1997	-0.0367	0.0997	0.0657	0.0997	1.2845	-0.0122	-0.0002	0.0003	0.0003	-0.0072
2	-0.6631	-0.1195	0.0657	0.0746	0.0657	1.2353	-0.0317	-0.0178	-0.0190	0.0185	0.0131
3	-5.2853	-0.9734	0.0746	0.0888	0.0746	1.0888	-0.2764	-0.1336	-0.0300	-0.0488	0.1685
4	9.2288	1.9439	0.2209	0.8791	0.2209	0.8791	1.0352	-0.1982	0.0242	0.5504	-0.1413
5	-3.2927	-0.6276	0.1532	0.2889	0.1532	1.2889	-0.2659	0.0770	-0.1307	-0.0013	0.0052
6	0.7893	0.1418	0.0595	0.2352	0.0595	1.2352	0.4357	0.0201	-0.0197	0.0083	-0.0011
7	8.6887	1.6423	0.0667	0.4391	0.0667	0.4391	-0.5269	0.0718	-0.5052	0.0093	0.0933
8	-2.0652	-0.5163	0.5101	2.2700	-0.5163	2.2700	-0.5269	-0.0404	-0.5052	0.4051	-0.0296
9	5.6009	1.0511	0.1038	1.0993	0.3577	1.0993	1.1157	-0.6802	-0.5511	0.7911	-0.0466
10	5.2467	1.2569	0.4407	1.6473	1.1157	1.6473	0.0127	-0.0044	-0.0002	0.0047	0.1144
11	-0.2816	-0.0506	0.0594	1.2292	-0.0594	1.2292	-0.2658	-0.0232	0.0746	-0.0033	-0.0033
12	-6.2488	-1.1439	0.0512	1.0088	-0.0512	1.0088	0.2917	-0.2778	-0.0744	-0.0001	-0.1275
13	4.8285	0.8972	0.0956	1.1370	0.0956	1.1370	0.1246	-0.0284	-0.0351	0.0061	-0.1871
14	1.9258	0.3561	0.1090	1.2740	0.1090	1.2740	-1.1533	0.6394	0.0049	0.0061	0.9942
15	-8.0651	-1.7714	0.2984	1.0619	-0.2984	1.0619	0.1662	0.1597	-0.0037	0.1067	-1.0800
16	2.7832	0.5120	0.0953	1.2239	0.0953	1.2239	0.5607	0.5396	-0.0217	-0.1611	-0.1070
17	7.7567	1.5008	0.1225	0.9566	0.5607	0.9566	-0.1100	-0.0350	0.0517	-0.0811	-0.3432
18	-1.1629	-0.2258	0.1945	1.4251	-0.1945	1.4251	0.1638	0.0458	-0.0285	-0.0380	0.0533
19	3.1235	0.5693	0.0764	1.1939	0.0764	1.1939	0.0211	-0.0060	0.0169	-0.0093	0.0036
20	0.2394	0.0459	0.1751	1.4016	0.0459	1.4016	0.1040	0.0651	0.0043	-0.0414	-0.0070
21	2.3831	0.4285	0.0556	1.1920	0.0556	1.1920	-0.4399	-0.3581	-0.2741	0.3038	0.2955
22	-7.4145	-1.4172	0.0893	0.9794	-0.0893	0.9794	0.0343	0.0060	-0.0111	0.0103	0.0074
23	0.9926	0.1762	0.0366	1.1950	0.0366	1.1950	-0.4286	-0.1985	0.1926	-0.2786	0.2425
24	-5.4142	-1.0399	0.1452	1.1563	-0.1452	1.1563	0.2714	-0.2029	0.1410	-0.1273	0.1653
25	-4.3342	-0.8059	0.1019	1.1710	-0.1019	1.1710	-0.3632	-0.2731	0.0802	-0.0830	0.2103
26	-8.4642	-1.5821	0.0516	0.8555	-0.0516	0.8555	-0.4355	-0.1161	0.2702	-0.2978	0.0198
27	-9.7562	-1.8571	0.0563	0.7577	-0.0563	0.7577	-0.4355	-0.0332	0.1474	-0.1117	-0.0890
28	-5.3951	-0.9855	0.0585	1.0665	-0.0585	1.0665	0.1206	0.1122	0.0132	-0.0321	-0.0850
29	2.2090	0.4027	0.0823	1.2305	0.0823	1.2305	0.4313	0.2281	-0.0040	-0.2052	0.0754
30	8.8849	1.6943	0.0776	0.8370	0.0776	0.8370	0.0805	-0.0145	0.0182	0.0114	-0.0364
31	-1.5847	-0.2870	0.0729	1.2224	-0.0729	1.2224	0.2701	0.1120	-0.0325	-0.0859	0.0604
32	4.9467	0.9121	0.0806	1.1142	0.0806	1.1142					
Sum of Residuals = 0											
Sum of Squared Residuals = 890.45693											
Predicted Residual SS (PRESS) = 1220.61680											

Figure 4-6 The residual test for PHP-based LOC estimation model

The model was validated using the dataset shown in Table 4-2. MMRE and Pred(0.25) computed are 0.15 and 0.81 respectively. These values fall within the acceptable level. Therefore, the evaluation result supports the validity of the LOC estimation model built.

Size Estimation for IS from their CDM *Estimating KLOC Using MLR*

Table 4-2 The MLR validation results for Open-Source PHP-based LOC estimation model

System	C	R	$\bar{A}$	Actual Size (KLOC)	Estimated Size (KLOC)	MRE
ackerTodo	8	5	5.000	9.47	8.325	0.12
Bookmark4u	10	10	5.800	19.381	19.689	0.02
ByteMonsoon	6	2	8.000	4.219	2.783	0.34
Core	8	5	8.125	17.065	10.363	0.36
e107	34	12	7.912	65.649	67.215	0.05
eFiction	9	6	6.222	12.443	12.035	0.03
Fdcl	5	2	11.600	3.856	3.889	0.01
galant	6	4	11.333	10.225	8.300	0.19
Infocentral	19	13	8.053	47.734	40.136	0.16
jamdb	4	5	8.500	4.554	5.643	0.24
openrealty	9	8	6.333	16.846	15.451	0.08
phpESP	16	15	5.438	33.645	35.258	0.05
Phpnews	6	5	6.167	6.532	6.604	0.01
phpollster	4	3	7.000	1.818	1.321	0.27
PhpScheduleIt	6	9	8.667	16.006	14.922	0.07
Phpsera	6	6	7.000	15.846	8.819	0.44
Phpwims	6	4	9.333	10.507	6.996	0.33
Phpwscookbook	5	5	10.200	7.607	7.992	0.05
plume	11	7	6.455	15.875	16.340	0.03
Quantum_Star_SE	30	23	9.533	57.643	68.679	0.19
Rasmp	10	4	6.600	9.452	10.178	0.08
Rimps	10	6	6.500	10.81	13.457	0.25
so-net	20	14	10.100	38.781	41.590	0.07
Supersurf	4	3	7.000	1.489	1.321	0.11
Usebb	6	3	13.333	6.118	7.932	0.30
videodb	14	10	4.643	24.227	23.898	0.01
Webaddressbook	3	2	20.000	8.363	6.884	0.18
wikiwig	15	9	5.800	23.457	24.222	0.03
yabbse	11	10	12.818	22.475	25.505	0.14
zebraz	6	5	7.833	8.59	7.690	0.11
Ztml	10	3	9.600	11.724	10.462	0.11
MMRE = 0.15, Pred(0.25) = 0.81						

Size Estimation for IS from their CDM *Estimating KLOC Using MLR*

4.4.3 Java-based Systems

There are three pairs of datasets collected for Java-based systems: Industry Java-based systems, Open-Source Java-based systems and Combined Industry and Open-Source Java-based systems.

4.4.3.1 Industry Java-based systems

The pair of datasets for industry Java-based systems is collected from a number of different organizations in the industry. It is collected from systems in a variety of domains including food-supply management, business management, project management, schedule management, demand chain management, freight management and booking management. Both datasets in this pair have similar composition. The dataset for model building is shown in Table E-5. From this dataset, we built the following model (Figure 4-7):

$$KLOC = -10.729 + 1.324 * C + 1.254 * R + 0.889 * \bar{A} \quad (4.8)$$

In significance test, $(Pr > F) < 0.0001$ and $(Pr > |t|)$ for intercept, C , R and $\bar{A}$ are 0.0008, <0.0001 , <0.0001 and 0.0178 respectively. In fitness test, $R_a^2 = 0.9910$. In collinearity test shown in Figure 4-8, the variance inflation factors (VIFs) and the condition indexes of all the independent variables are less than 10 (the threshold). In extreme case test shown in Figure 4-9, the absolute values of studentized residuals (RStudents) for all the systems are below 2 (the threshold). Therefore, all the test results are affirmative.

The model was validated using the dataset shown in Table 4-3. MMRE and Pred(0.25) computed are 0.15 and 0.81 respectively. These values fall within

Size Estimation for IS from their CDM *Estimating KLOC Using MLR*

the acceptable level. Therefore, the evaluation result supports the validity of the LOC estimation model built.

The REG Procedure									
Model: MODEL1									
Dependent Variable: Size									
Analysis of Variance									
Source	DF	Sum of Squares	Mean Square	F Value	Pr > F				
Model	3	11839	3946.38144	548.85	<.0001				
Error	12	86.28343	7.19029						
Corrected Total	15	11925							
Root MSE									
Dependent Mean									
Coeff Var									
R-Square									
Adj R-Sq									
F Value									
Pr > F									
Parameter Estimates									
Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > t	Tolerance	Variance Inflation		
Intercept	1	-10.72928	2.41728	-4.44	0.0008	0.43212	2.31416		
E	1	1.32448	0.07440	17.80	<.0001	0.38165	2.62023		
K	1	1.25400	0.13883	9.03	<.0001	0.79350	1.26024		
K	1	0.88946	0.32399	2.75	0.0178				

Figure 4-7 The SAS printout for Industry Java-based LOC estimation model

Size Estimation for IS from their CDM *Estimating KLOC Using MLR*

Number	Eigenvalue	Collinearity Diagnostics			
		Condition Index	Intercept	Proportion of Variation	R
1	3.74486	1.00000	0.00513	0.00709	0.00403
2	0.17158	4.67182	0.11435	0.32283	0.01937
3	0.04562	9.06022	0.61026	0.35980	0.43400
4	0.03794	9.93335	0.27027	0.31028	0.54259
					0.00441
					0.07069
					0.21837
					0.70654

Figure 4-8 The collinearity diagnostics for Industry Java-based LOC estimation model

Size Estimation for IS from their CDM *Estimating KLOC Using MLR*

The REG Procedure
Model: MODEL1
Dependent Variable: Size

Output Statistics

Obs	Residual	RStudent	Hat	Diag	H	Cov	Ratio	DFFITs	Intercept	-----DPBETAS-----			R	R̄
										E				
1	3.7416	1.6993	0.2197	0.2445	0.7144	0.9018	0.5879	-0.0132	-0.4310	-0.0884				
2	3.6471	1.6792	0.2445	0.7524	0.9552	0.6338	0.0097	-0.5305	0.0492					
3	-2.4505	-0.9812	0.1353	1.1709	-0.3881	-0.1680	0.1067	-0.1049	-0.0890					
4	-2.4301	-0.9692	0.1302	1.1732	-0.3749	-0.2445	0.1885	-0.0518	0.0875					
5	-1.2504	-0.4858	0.1373	1.5081	-0.1938	-0.1083	0.1060	-0.1039	0.1003					
6	0.4979	0.1870	0.0935	1.5426	0.0501	0.0276	-0.0276	0.0072	-0.0016					
7	2.1925	0.9118	0.2071	1.3347	0.4560	0.2792	-0.1756	0.2684	-0.3503					
8	-3.2156	-1.2847	0.0815	0.8815	-0.3826	-0.2302	-0.0285	-0.0282	0.1805					
9	-2.8589	-1.1934	0.1735	1.0531	-0.5471	-0.0059	0.3070	-0.4375	0.1488					
10	-0.4731	-0.1905	0.2112	1.7720	-0.0386	0.0277	0.0625	-0.0738	-0.0097					
11	2.4024	1.0828	0.3056	1.3601	0.7183	0.2186	0.2829	0.1433	-0.4653					
12	1.0274	0.4593	0.3500	2.0195	0.3371	-0.1910	0.1887	-0.1107	0.2030					
13	-0.2217	-0.0999	0.3719	2.2468	-0.0769	0.0205	-0.0501	0.0244	-0.0111					
14	0.4769	0.1915	0.2068	1.7620	0.0378	0.0210	0.0439	0.0128	-0.0493					
15	2.1053	1.4304	0.6725	2.1854	2.0495	-1.3763	-0.8730	0.4964	1.5956					
16	-3.1908	-1.7524	0.4593	0.9784	-1.6151	-0.3424	-1.0057	1.4936	-0.6433					

Sum of Residuals

Sum of Squared Residuals

Predicted Residual SS (PRESS)

0

86.28345

187.85800

Figure 4-9 The residual test for Industry Java-based LOC estimation model

Size Estimation for IS from their CDM *Estimating KLOC Using MLR*

Table 4-3 The MLR validation results for Industry Java-based LOC estimation model

System No	C	R	$\bar{A}$	Actual Size (KLOC)	Estimated Size (KLOC)	MRE
1	49	22	5.898	84.89	86.978	0.03
2	12	7	4.250	20.446	17.715	0.13
3	8	6	7.750	17.29	14.277	0.17
4	16	12	5.125	34.075	30.059	0.12
5	18	14	6.389	40.113	36.339	0.09
6	17	14	7.412	28.722	35.924	0.25
7	31	27	6.871	71.37	70.281	0.02
8	30	29	10.300	60.34	74.514	0.24
9	11	11	5.091	20.45	22.155	0.08
10	35	36	8.486	93.27	88.299	0.05
11	19	20	4.789	31.69	43.765	0.38
12	30	33	6.267	77.52	75.944	0.02
13	21	24	7.714	45.384	54.029	0.19
14	24	28	6.000	52.1	61.493	0.18
15	7	9	4.857	19.36	14.143	0.27
16	18	26	5.944	56.744	50.992	0.10
MMRE = 0.15, Pred(0.25) = 0.81						

4.4.3.2 Open-Source Java-based systems

The pair of datasets for Open-Source Java-based systems is collected from two open-source websites: SourceForge and Freshmeat (sourceforge; freshmeat). It is collected from systems in a variety of domains including content management, project organization, member management, job scheduler and entertainment. Both datasets in this pair has similar composition. The dataset for model building is shown in Table E-7. From this dataset, we built the following model (Figure 4-10):

Size Estimation for IS from their CDM *Estimating KLOC Using MLR*

$$KLOC = -10.121 + 1.201 * C + 1.439 * R + 0.726 * \bar{A}$$
 (4.9)

The REG Procedure													
Model: MODEL1													
Dependent Variable: Size													
Analysis of Variance													
Source	DF	Sum of Squares	Mean Square	F Value	Pr > F								
Model	3	19321	6106.88488	220.17	<.0001								
Error	26	721.15875	27.73688										
Corrected Total	29	19042											
Parameter Estimates													
Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > t	Tolerance	Variance Inflation						
Intercept	1	-10.12061	2.90126	-3.49	0.0017	0.43488	2.29948	0					
E	1	1.20073	0.16747	7.17	<.0001	0.42014	2.38013						
R	1	1.43937	0.13952	10.32	<.0001	0.93833	1.06572						
A	1	0.72552	0.35276	2.06	0.0499								

Figure 4-10 The SAS printout for Open-source Java-based LOC estimation model

In significance test, (Pr > F) < 0.0001 and (Pr > |t|) for intercept, C, R and $\bar{A}$ are 0.0017, <0.0001, <0.0001 and 0.0499 respectively. In fitness test, $R_a^2 = 0.9587$. In collinearity test shown in Figure 4-11, the variance inflation factors (VIFs) and the condition indexes of all the independent variables are less than 10 (the threshold). In extreme case test shown in Figure 4-12, except for jwp

Size Estimation for IS from their CDM *Estimating KLOC Using MLR*

and xc-act systems, the absolute values of studentized residuals (RStudents) for all the systems are below 2 (the threshold). The studentized residuals for these two systems are 2.2146 and 2.3948 respectively. We checked the data for these two systems and found them correct. Therefore, no adjustment top the dataset was made. Therefore, except these two systems (out of 30 systems) show some mild extreme case behavior, all the test results are affirmative.

Number	Eigenvalue	Collinearity Diagnostics			
		Condition Index	Intercept	Proportion of Variation	$\bar{\lambda}$
1	3.58714	1.00000	0.00776	0.00714	0.00950
2	0.26702	3.66527	0.08136	0.04519	0.14541
3	0.09102	6.27787	0.28608	0.26839	0.51526
4	0.05483	8.08832	0.62480	0.67928	0.32973

Figure 4-11 The collinearity diagnostics for Open-source Java-based LOC estimation model

Size Estimation for IS from their CDM Estimating KLOC Using MLR

The REG Procedure Model: MODEL1 Dependent Variable: Size											
Output Statistics											
Obs	Residual	RStudent	Hat	Diag	H	Cov Ratio	DFPITS	Intercept	-----DFBETAS----- E	K	K
1	1.8547	0.3662	0.1060			1.2808	0.1261	0.0629	-0.0444	-0.0301	0.0274
2	2.7136	0.5386	0.1097			1.2548	0.1891	0.1118	-0.0952	-0.0133	0.0243
3	-0.0792	-0.0156	0.1024			1.3033	-0.0053	-0.0049	0.0019	0.0001	0.0025
4	-2.0742	-0.4060	0.0890			1.2509	-0.1269	-0.1167	0.0440	0.0029	0.0586
5	-1.5157	-0.3153	0.2010			1.4409	-0.1587	0.2654	0.0110	0.0615	-0.1224
6	2.4491	0.4857	0.1104			1.2666	0.1711	0.1643	-0.0703	0.0127	-0.0936
7	5.3082	1.0763	0.1177			1.1062	0.3931	0.3757	-0.1926	0.0540	-0.2016
8	0.5121	0.0995	0.0806			1.2703	0.0294	0.0269	-0.0095	-0.0006	-0.0140
9	-0.5007	-0.1080	0.2551			1.5676	-0.0692	0.0117	0.0214	0.0045	-0.0512
10	-4.5485	-0.8788	0.0427			1.0820	-0.1856	-0.0795	-0.0119	0.0661	-0.0122
11	1.8392	0.3747	0.1045			1.2775	0.1280	0.0958	0.0307	-0.0416	-0.0863
12	-7.5236	-1.5050	0.0552			0.8752	-0.3637	0.0493	0.0266	0.0269	-0.2260
13	-7.7783	-1.6417	0.1379			0.9010	-0.6565	0.1064	-0.5520	0.5088	-0.0718
14	-0.3238	-0.0628	0.0792			1.2697	-0.0184	-0.0066	0.0135	-0.0114	-0.0023
15	-2.2723	-0.4335	0.0404			1.1832	-0.0890	-0.0474	0.0256	-0.0010	-0.0028
16	4.9384	0.9808	0.0874			1.1022	0.3035	0.1113	0.1593	-0.2109	-0.0706
17	-5.7171	-1.1822	0.1440			1.0994	-0.4848	-0.0886	0.1720	-0.3837	0.1480
18	5.3602	1.1077	0.1484			1.1342	0.4624	-0.1658	-0.1563	0.1300	0.3541
19	8.8645	2.2146	0.3356			0.8601	1.5739	0.4979	-0.8663	1.4235	-0.6424
20	6.8987	1.4202	0.1160			0.9703	0.5145	0.0190	0.0184	0.2736	-0.1648
21	4.0375	0.8004	0.0953			1.1686	0.2598	0.0112	0.1781	-0.2060	0.0233
22	-4.2638	-0.8424	0.0867			1.1452	-0.2595	0.1343	-0.1234	0.0722	-0.1506
23	-3.4880	-0.6712	0.0459			1.1428	-0.1488	-0.0772	-0.0084	-0.0248	0.0737
24	-1.7754	-0.3379	0.0387			1.1950	-0.0678	-0.0120	-0.0228	0.0102	0.0062
25	-9.0730	-1.9601	0.1431			0.7706	-0.8008	-0.1313	-0.3610	-0.0531	-0.4351
26	-2.9791	-0.6659	0.2359			1.5484	-0.4323	0.1639	-0.1387	-0.3019	-0.1366
27	1.9549	0.3746	0.0506			1.2050	0.0865	0.0193	0.0447	-0.0274	-0.0221
28	5.2025	1.2311	0.3434			1.4080	0.8904	-0.0600	0.7877	-0.4940	-0.2795
29	-7.3076	-1.4696	0.0688			0.9019	-0.3993	-0.0963	0.0439	-0.2158	0.1388
30	9.2267	2.3948	0.3674			0.8095	1.8249	-1.4309	1.0597	-0.5332	1.2216
Sum of Residuals 0											
Sum of Squared Residuals 721.15875											
Predicted Residual SS (PRESS) 1126.79075											

Figure 4-12 The residual test for Open-source Java-based LOC estimation model

The model was validated using the dataset shown in Table 4-4. MMRE and Pred(0.25) computed are 0.20 and 0.79 respectively. These values fall within the acceptable level. Therefore, the evaluation result supports the validity of the LOC estimation model built.

Size Estimation for IS from their CDM *Estimating KLOC Using MLR*

Table 4-4 The MLR validation results for Open-Source Java-based LOC estimation model

System	C	R	$\bar{A}$	Actual Size (KLOC)	Estimated Size (KLOC)	MRE
abaguibuilder	29	23	7.345	83.176	63.137	0.24
Art	14	7	6.929	17.67	21.796	0.23
Bofhms	12	4	4.333	10.507	13.193	0.26
Contineo	11	11	7.364	21.067	24.265	0.15
Dspace	32	20	4.250	47.57	60.177	0.27
Ejen	5	3	4.600	4.437	3.541	0.20
Imcms	59	30	4.712	91.424	107.329	0.17
Itracker	19	14	6.579	38.721	37.620	0.03
jdbforms	19	15	6.789	50.72	39.212	0.23
jmbase	8	8	5.250	19.723	14.811	0.25
Jwma	22	19	8.045	63.228	49.483	0.22
Jbooks	11	7	3.727	12.817	15.869	0.24
Jfolder	10	8	12.800	29.288	22.694	0.23
Jgossip	14	9	5.714	28.047	23.793	0.15
Jpo	5	5	5.800	8.54	7.290	0.15
Jwnl	7	5	3.571	6.913	8.074	0.17
mp3cattle	9	10	3.444	14.598	17.579	0.20
openjms	17	16	6.824	59.72	38.274	0.36
Openhre	9	4	3.444	9.932	8.945	0.10
Quartz	11	11	4.364	22.954	22.087	0.04
sqlunit	26	20	7.808	74.089	55.553	0.25
tau_lastest	14	16	8.000	48.039	35.525	0.26
testsuite	15	9	5.067	30.076	24.523	0.19
Tesuji	2	1	16.000	8.01	5.336	0.33
MMRE = 0.20 Pred(0.25) = 0.79						

Size Estimation for IS from their CDM *Estimating KLOC Using MLR*

4.4.3.3 Combined Industry and Open-Source Java-based systems

The pair of datasets for combined Java-based systems is the combination of java-based system collection from the industry and Open-Source. From the dataset that is formed by the combination of the two datasets (Table E-5 and Table E-7) collected from the industry and Open-Source Java-based systems for model building, we built the following model (Figure 4-13):

$$KLOC = -10.576 + 1.258 * C + 1.392 * R + 0.754 * A$$
 (4.10)

The REG Procedure									
Model: MODEL1									
Dependent Variable: Size									
Analysis of Variance									
Source	DF	Sum of Squares	Mean Square	F Value	Pr > F				
Model	3	32372	10837	556.73	< .0001				
Error	42	819.08360	19.50199						
Corrected Total	45	33391							
Root MSE									
Dependent Mean									
Coeff Var									
R-Square									
Adj R-Sq									
Parameter Estimates									
Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > t	Tolerance	Variance Inflation		
Intercept	1	-10.57626	1.95424	-5.41	< .0001				0
E	1	1.25831	0.08243	15.26	< .0001	0.48974	2.04191		
R	1	1.39164	0.09572	14.54	< .0001	0.47110	2.1269		
A	1	0.75365	0.25531	2.95	0.0051	0.89666	1.11524		

Figure 4-13 The SAS printout for combined Java-based LOC estimation model

Size Estimation for IS from their CDM *Estimating KLOC Using MLR*

In significance test, $(Pr > F) < 0.0001$ and $(Pr > |t|)$ for intercept, C , R and $\bar{A}$ are <0.0001 , <0.0001 , <0.0001 and 0.0051 respectively. In fitness test, $R_a^2 = 0.9737$. In collinearity test shown in Figure 4-14, the variance inflation factors (VIFs) and the condition indexes of all the independent variables are less than 10 (the threshold). In extreme case test shown in Figure 4-15, in the 46 systems, except for jcv, northstarbbs, Sacash and xc-ast systems, the absolute values of studentized residuals (RStudents) for all the systems are below 2 (the threshold). The studentized residuals for these four systems are -2.0230, 2.7722, -2.2772 and 2.2337 respectively. As we confirmed the correctness of the data for these four systems, no adjustment to the dataset was made. Therefore, all the test results are affirmative except that these four systems (out of 43 systems) show some extreme case behavior. However, except the behavior of the second and third system is more moderate, the other two systems are mild. We believe that this could be due to the combination of the two datasets with different accuracy in their conceptual data models – the conceptual data models of the Open-Source datasets are clearly more accurate than the conceptual data models from the industry dataset. A brief description of these extreme case behaviors and the assessment of the impacts of these behaviors are given as below.

Jcv is a powerful multilingual Web application designed for creating, searching and printing resumes. The numbers of class and relationship for this system are 26 and 30 respectively. This system is using a reporting tools (JasperReports, iReport). The reuse may affect the estimated size of this system. Currently, we haven't considered the impact of reuse on software size estimation. In the future, it should be an aspect to be explored in our research.

Northstarbbs is a Java based Bulletin Board System. Although the number of class is 15, the number of relationship is 20. It shows this system has a complicated relationship. Northstarbbs also uses a complicate architecture,

Size Estimation for IS from their CDM *Estimating KLOC Using MLR*

the MVC architecture, to implement the system. It considers a complicated algorithm in encryption of the password, and also provides many categories of chatting rooms. Thus may affect the estimation on this system.

Sacash is a Quicken-like financial account manager. It handles multiple accounts, account transfers, and reconciles information. In this system, the numbers of class and relationship are 6 and 4 respectively. However the number of attributes is only 24. And the system only has 8 web pages. The functions this system provided are not very complicated. The system just keep the accounts, process the account transfer and only provides a simple report to list the accounts. Analyses and detail reports are not provided. Thus the size of the source code is not large.

Xc-ast is a system that analyzes data about call queues maintained by an Asterisk PBX. It is designed to be used for monitoring call centers that require real-time activity reports. This system provides a lot of interface to generate different reports, such as general report of agent logon and log off, general and detail reports of taken calls and lost calls, etc. However, in this system, the number of class is 7, the number of relationship is only 2 and the number of attributes is 78. Although this system implements a lot interface codes, the relationship in this system is very simple. That because these reports don't need so many relationships of the data model, but the different or more columns in a same table. This may be the reason that this system is an extreme case.

The model was validated using the dataset that is formed by the combination of the two datasets (Table E-6 and Table E-8) collected from the industry and open-source Java-based systems for model validation. MMRE and Pred(0.25) computed are 0.18 and 0.78 respectively. These values fall within the acceptable level. Therefore, the evaluation result supports the validity of the LOC estimation model built.

Collinearity Diagnostics						
Number	Eigenvalue	Condition Index	Proportion of Variation			
			Intercept	E	R	$\bar{R}$
1	3.62927	1.00000	0.00769	0.00905	0.00308	0.00776
2	0.22713	3.99739	0.11929	0.14509	0.14705	0.12537
3	0.08143	6.67594	0.09069	0.70112	0.70661	0.09119
4	0.06217	7.64038	0.78234	0.14473	0.13726	0.77548

Figure 4-14 The collinearity diagnostics for Combined Java-based LOC estimation model

Size Estimation for IS from their CDM *Estimating KLOC Using MLR*

The REG Procedure
Model: MODEL1
Dependent Variable: Size

Output Statistics

Obs	Residual	RStudent	Mat Diag H	Cov Ratio	DFFITS	Intercept	DPBETAS E	R	A
1	1.9308	0.4509	0.0776	1.1704	0.1308	0.0594	-0.0235	-0.0623	0.0327
2	2.9045	0.6787	0.0729	1.1359	0.1903	0.1012	-0.0584	-0.0664	0.0331
3	0.1141	0.0263	0.0723	1.1869	0.0874	0.0069	-0.0014	-0.0016	-0.0035
4	-1.8995	-0.4402	0.0638	1.1542	-0.1149	-0.1059	0.0212	0.0235	0.0525
5	-1.6811	-0.4102	0.1557	1.2830	-0.1761	0.0282	0.0088	0.0881	-0.1350
6	2.6994	0.6313	0.0757	1.1462	0.1807	0.1735	-0.0417	-0.0205	-0.0996
7	5.6134	1.3352	0.0757	1.0060	0.3849	0.3666	-0.1146	-0.0271	-0.2015
8	0.6740	0.1555	0.0586	1.1670	0.0388	0.0358	-0.0069	-0.0070	-0.0183
9	-0.3954	-0.0979	0.1832	1.3469	-0.0464	0.0123	0.0130	0.0108	-0.0389
10	-4.5979	-1.0599	0.0321	1.0212	-0.1932	-0.1073	0.0007	0.0762	-0.0057
11	1.7985	0.4204	0.0798	1.1763	0.1238	0.1106	0.0251	-0.0373	-0.0858
12	-7.5146	-1.7785	0.0375	0.8498	-0.3509	0.0230	0.0798	0.0269	-0.2103
13	-8.3653	-2.0230	0.0587	0.7993	-0.5031	-0.0941	-0.3550	0.3789	-0.0259
14	0.0977	0.0224	0.0480	1.1566	0.0050	0.0014	-0.0037	0.0027	0.0006
15	-2.1700	-0.4943	0.0294	1.1079	-0.0860	-0.0497	0.0248	0.0076	-0.0002
16	4.6151	1.0790	0.0584	1.0455	0.2686	0.1723	0.1157	-0.1747	-0.0782
17	-5.2223	-1.2661	0.1151	1.0675	-0.4567	-0.0802	0.2159	-0.3975	0.1649
18	5.5905	1.3464	0.1020	1.0314	0.4538	-0.1892	-0.1997	0.1199	0.3469
19	9.7983	2.7722	0.2575	0.7459	1.6324	0.3792	-0.9137	1.5008	-0.7202
20	7.1162	1.7257	0.0870	0.9112	0.5329	0.0838	-0.1277	0.4017	-0.2224
21	3.6049	0.8364	0.0543	1.0882	0.2005	0.0710	0.1055	-0.1534	0.0082
22	-4.4946	-1.0401	0.0406	1.0342	-0.2139	0.0808	-0.0308	0.0178	-0.1319
23	-3.4073	-0.7816	0.0344	1.0749	-0.1476	-0.1006	0.0188	-0.0423	0.0856
24	-1.8351	-0.4162	0.0228	1.1080	-0.0636	-0.0296	-0.0029	-0.0018	0.0135
25	-9.1538	-2.2772	0.0889	0.7506	-0.7111	-0.3064	-0.1109	-0.2651	0.5184
26	-2.4396	-0.6204	0.2225	1.3643	-0.3318	0.1346	0.1645	-0.2658	-0.0988
27	1.8090	0.4114	0.0280	1.1144	0.0699	0.0403	0.0176	-0.0094	-0.0283
28	4.4002	1.0732	0.1348	1.1393	0.4236	0.1415	0.3215	-0.1582	-0.2303
29	-7.0856	-1.6839	0.0524	0.8894	-0.3962	-0.1327	0.1281	-0.2733	0.1715
30	8.6218	2.2337	0.1635	0.8315	0.9874	-0.7413	0.2643	-0.0550	0.7434
31	4.0019	0.9275	0.0485	1.0651	0.2094	0.1565	-0.0623	-0.0457	-0.0274
32	4.0405	0.9387	0.0527	1.0677	0.2215	0.1448	-0.0679	-0.0599	0.0024
33	-2.4372	-0.5578	0.0370	1.1094	-0.1093	-0.0405	0.0505	-0.0006	-0.0286
34	-2.8415	-0.6514	0.0377	1.0982	-0.1289	-0.0771	0.0770	-0.0379	0.0146
35	-1.6027	-0.3784	0.0988	1.2050	-0.1253	-0.0004	-0.0830	0.1075	-0.0464
36	-1.9817	-0.4520	0.0380	1.1164	-0.0835	-0.0444	0.0359	-0.0417	0.0280
37	0.1963	0.0445	0.0275	1.1321	0.0075	0.0027	-0.0033	0.0017	0.0008
38	1.3164	0.3015	0.0438	1.1415	0.0645	0.0374	-0.0095	0.0290	-0.0401
39	-3.3639	-0.7694	0.0294	1.0713	-0.1340	-0.0380	-0.0549	0.0132	0.0349
40	-3.8055	-0.8781	0.0417	1.0566	-0.1832	0.0095	0.0336	-0.1084	0.0046
41	-1.4250	-0.3285	0.0555	1.1537	-0.0796	0.0239	0.0253	-0.0468	-0.0258
42	2.2293	0.5413	0.1450	1.2518	0.2230	0.0038	0.1632	-0.0383	-0.0838
43	1.8519	0.4636	0.1972	1.3433	0.2298	-0.1329	0.1707	-0.0891	0.1007
44	0.6377	0.1658	0.2593	1.4826	0.0981	-0.0373	0.0869	-0.0416	0.0117
45	0.4364	0.1038	0.1151	1.2431	0.0374	-0.0027	0.0291	-0.0080	-0.0092
46	1.6257	0.4058	0.1936	1.3438	0.1988	-0.1221	-0.0649	0.0542	0.1647
Sum of Residuals						0			
Sum of Squared Residuals						819.08360			
Predicted Residual SS (PRESS)						1034.41622			

Figure 4-15 The residual test for Combined Java-based LOC estimation model

Size Estimation for IS from their CDM *Estimating KLOC Using MLR*

Table 4-5 The MLR validation results for combined Java-based LOC estimation model

System	C	R	$\bar{A}$	Actual Size (KLOC)	Estimated Size (KLOC)	MRE
abaguibuilder	29	23	7.345	83.176	63.137	0.24
Art	14	7	6.929	17.67	21.796	0.23
Bofhms	12	4	4.333	10.507	13.193	0.26
Contineo	11	11	7.364	21.067	24.265	0.15
Dspace	32	20	4.250	47.57	60.177	0.27
Ejen	5	3	4.600	4.437	3.541	0.20
Imcms	59	30	4.712	91.424	107.329	0.17
Itracker	19	14	6.579	38.721	37.620	0.03
jdbforms	19	15	6.789	50.72	39.212	0.23
jmbase	8	8	5.250	19.723	14.811	0.25
Jwma	22	19	8.045	63.228	49.483	0.22
Jbooks	11	7	3.727	12.817	15.869	0.24
Jfolder	10	8	12.800	29.288	22.694	0.23
Jgossip	14	9	5.714	28.047	23.793	0.15
Jpo	5	5	5.800	8.54	7.290	0.15
Jwnl	7	5	3.571	6.913	8.074	0.17
mp3cattle	9	10	3.444	14.598	17.579	0.20
openjms	17	16	6.824	59.72	38.274	0.36
Openhre	9	4	3.444	9.932	8.945	0.10
Quartz	11	11	4.364	22.954	22.087	0.04
sqlunit	26	20	7.808	74.089	55.553	0.25
tau_lastest	14	16	8.000	48.039	35.525	0.26
testsuite	15	9	5.067	30.076	24.523	0.19
Tesuji	2	1	16.000	8.01	5.336	0.33
1	49	22	5.898	84.89	63.137	0.24
2	12	7	4.250	20.446	21.796	0.23
3	8	6	7.750	17.29	13.193	0.26
4	16	12	5.125	34.075	24.265	0.15
5	18	14	6.389	40.113	60.177	0.275
6	17	14	7.412	28.722	3.541	0.20
7	31	27	6.871	71.37	107.329	0.17
8	30	29	10.300	60.34	37.620	0.03
9	11	11	5.091	20.45	39.212	0.23
10	35	36	8.486	93.27	14.811	0.25

Size Estimation for IS from their CDM *Estimating KLOC Using MLR*

11	19	20	4.789	31.69	49.483	0.22
12	30	33	6.267	77.52	15.869	0.24
13	21	24	7.714	45.384	22.694	0.23
14	24	28	6.000	52.1	23.793	0.15
15	7	9	4.857	19.36	7.290	0.15
16	18	26	5.944	56.744	8.074	0.17
MMRE = 0.18, Pred(0.25) = 0.78						

4.5 **Summary of the Results**

The results of model building and validation from all the five pairs of datasets are encouraging. Such results are reckoned as good by other researchers (Lai and Huang, 2003; Briand, et al., 1999).

We also observe that in all the models built the magnitude of the coefficient $\overline{A}$ is the lowest among the three independent variables. This is consistent with the intuitive understanding. Much complexity in information systems is caused by the maintaining of class instances and navigation between these instances. The numbers of attributes in its classes do not usually affect the complexity too much. Therefore, C and R have much greater influence than $\overline{A}$ on LOC.

4.6 **Threats to Validity**

Many organizations in the industry did not construct conceptual data models for information systems developed at the early stage of software development. Furthermore, to ensure that the data collected from a system is correct, we have to verify the data with the system documentations. Furthermore, due to the view of confidentiality, many industry organizations did not allow us to access their system documentations to verify the accuracy of the data supplied

Size Estimation for IS from their CDM *Estimating KLOC Using MLR*

by them. This makes the data collection from industry extremely difficult. As a result, the sizes of our industry datasets just managed to meet the minimum criteria (15 datapoints). This may affect the accuracy of our validation. To address this issue, in addition to the industry datasets, we have also collected some datasets from Open-Source systems to build and validate models.

The sizes of samples collected from Open-Source systems are much better. Furthermore, the datasets collected from open-source also have the benefit of allowing other people to verify their accuracy. However, the conceptual data models for these systems have to be constructed from reengineering their source codes and database schemas. Though steps have been taken to ensure that only conceptual entities but not implementation data are included in the models, they are usually more accurate than those models constructed at the early stage of software development. Therefore, models built from Open-Source samples also face some threat to validity. Fortunately, models built from Open-Source and industry samples that were developed based on the same programming language are quite consistent to a certain extent.

Furthermore, surely, it is not possible for our samples to cover the full range of possible LOC values well. So, the models that we have built are not appropriate for estimating systems with LOC values that are significantly larger or smaller than the LOC values in the respective samples. The current evaluation criteria of using MMRE and PRED(0.25) for building models is also not perfect (Stensrud, et al., 2002).

In summary, we do not claim that we have built models that are readily for use. However, in the worst case, at least, our work has shown that it is useful and promising to experiment the proposed method on much larger scale to address the critical software sizing problem in the industry. Clearly, such experiment requires organized industry and research community cooperation on a long term basis.

4.7 Summary

Regression-based technique is the most common method for estimation. In this chapter, multiple linear regression technique is adopted to build the proposed model described in chapter 3. Five pairs of datasets collected from industry and Open-Source are used to do the model building and validation. In model building, four statistical tests were carried out, they are significant test, fitness test, collinearity test and extreme case test. The results of the testing show the LOC estimation models we build by using multiple linear regression satisfied the criteria. Model validation are carried out by using MMRE and Pred(0.25) as criteria. The evaluation results support the validity of the LOC estimation models built. Thus, the encouraging results of model building and validation are got.

Threats to validity were also discussed in this chapter. Since data collection from industry is quite difficult, the sizes of the industry datasets are not very sufficient and this may affect the accuracy of the validation. Furthermore, the datasets collected from Open-Source systems contain the data from reengineering the systems source codes and database schemas. Thus the data may be more accurate than the data collected in the early stage of software development. Therefore, model built from Open-Source systems also face some threat to validity. The models we built currently may not be readily for use. However, at least, our work has shown that it is useful and promising to experiment the proposed method on much larger scale to address the critical software sizing problem in the industry.

Chapter 5 Estimating KLOC Using Support Vector Machine

5.1 Support Vector Machine

5.1.1 Support Vector Machine

Support Vector Machine (SVM) is a machine learning technique which is based on Vapnik-Chervonenkis (VC) theory (Vapnik, 1982; Vapnik, 1995). It is originally designed for binary classification, but now it is effectively extended to multi-class problem. In the last several years, SVM has received much attention in research area and succeeded in solving real-life problems. It has been applied to a number of practical problems, such as optical character recognition, face identification, analyzing DNA data, signal processing, network intrusion detection, and so on (Cristianini and Shawe-Taylor, 2000; Scholkopf, et al., 1999; Osuna, et al., 1997; Muller, et al., 2001; Zhang, 1999; Kuh, 2001; Chen, et al., 2001).

The original idea of SVM is to use a linear separating hyperplane to maximize the margin between two classes (shown in Figure 5-1). In this figure, Figure 5(b) is a better separation.

Size Estimation for IS from their CDM *Estimating KLOC Using SVM*

Figure 5-1 (a) A shyperplane separation with small margin (b) A hyperplane separation with larger margin

Given the training data $(x_1, y_1), \dots, (x_l, y_l)$, where $x_i \in R^n$, and $y_i \in \{-1, +1\}$, $i = 1, \dots, l$, let us consider the Δ -margin separating hyperplane.

That is, we call a hyperplane

$$w \bullet x + b = 0, \quad \|w\| = 1. \quad (5.1)$$

where w is the weight vector, x is the input vector, b is the bias that determine the position of the separating hyperplane. A separating hyperplane is called a Δ -margin separating hyperplane if it satisfies:

$$y = \begin{cases} 1, & \text{if } (w \bullet x) + b \geq \Delta, \\ -1, & \text{if } (w \bullet x) + b \leq -\Delta. \end{cases} \quad (5.2)$$

where Δ is the margin. Classifications of vectors x that fall into the margin $(-\Delta, \Delta)$ are undefined.

Size Estimation for IS from their CDM Estimating KLOC Using SVM

It can be said that the set of vectors is separated by the optimal hyperplane if it is separated without error and the distance between the closest vector and the hyperplane is maximal. Let us describe the separating hyperplane by using the following equation:

$$y_i(\mathbf{w} \bullet \mathbf{x}_i + b) \geq 1 \quad (5.3)$$

where $\mathbf{w}$ is the weight vector, b is the bias, $\mathbf{x}_i$ is the i th training example, and y_i is the correct output of the SVM for the i th training example. The value y_i is +1 for the positive example in a class and -1 for the negative example. It is easy to check that the optimal hyperplane is the one that satisfies the conditions (5.3) and minimizes the magnitude of the weights. The problem is then transformed into a quadratic programming problem (Scholkopf, et al., 1999):

$$\min \frac{1}{2} \|\mathbf{w}\|^2 \quad (5.4)$$

The quadratic programming problem can be solved by considering the dual problem (Vapnik, 1995):

$$W(\alpha) = \sum_{i=1}^l \alpha_i - \frac{1}{2} \sum_{i=1}^l \sum_{j=1}^l \alpha_i \alpha_j y_i y_j (\mathbf{x}_i \bullet \mathbf{x}_j) \quad (5.5)$$

subject to $\alpha_i \geq 0$, $i = 1, \dots, l$ and $\sum_{i=1}^l \alpha_i y_i = 0$. α is the lagrange multiplier vector. Those inputs $\mathbf{x}_i$ s with $\alpha_i > 0$ are called support vectors. These support vectors are the critical vectors in determining the optimal margin classifier and usually a small subset of the sample set which can be denoted by X_{SVM} . For an unknown vector $\mathbf{x}_i$ classification, the decision function can be written as:

$$f(\mathbf{x}) = \text{sign} \left(\sum_{\mathbf{x}_i \in X_{SVM}} \alpha_i y_i (\mathbf{x} \bullet \mathbf{x}_i) + b \right) \quad (5.6)$$

Size Estimation for IS from their CDM *Estimating KLOC Using SVM*

where

$$w = \sum_{x_i \in X_{SVM}} \alpha_i y_i x_i$$

and the sum is over those nonzero support vectors with $\alpha_i > 0$.

SVM is made more powerful as it can be extended in the following two methods:

- To construct nonlinear decision functions (Boser, et al., 1992) by introducing kernel functions that satisfy Mercers Theorem (Scholkopf, et al., 1999). Function $\phi(x)$ is used to transform the original inputs space into a higher dimensional feature space. The inner products are replaced with kernel functions $K(x_i, x_j) = \phi(x_i) \bullet \phi(x_j)$.
- To handle the data that is linearly nonseparable (Cortes and Vapnik, 1995) by introducing slack variables to the quadratic programming problem. Data points are penalized if they are misclassified.

SVM with these problems can be described as:

$$\min \frac{1}{2} \|w\|^2 + P \sum_{i=1}^l \xi_i \quad (5.7)$$

subject to $y_i(w \bullet \phi(x_i) + b) \geq 1 - \xi_i$, ξ_i is the slack variable introduced to the i th inequality condition, $\xi_i \geq 0$, $i = 1, \dots, l$, $P > 0$.

A penalty term $P \sum_{i=1}^l \xi_i$ in the objective function and training errors are allowed. If the penalty parameter P is large enough and the data is linear separable, the problem goes back to (5.3) as all ξ_i will be zero. Then the constraint is now $0 \leq \alpha_i \leq P$ and the decision function can be:

Size Estimation for IS from their CDM Estimating KLOC Using SVM

$$W(\alpha) = \sum_{i=1}^l \alpha_i - \frac{1}{2} \sum_{i=1}^l \sum_{j=1}^l \alpha_i \alpha_j y_i y_j (\phi(x_i) \bullet \phi(x_j)) \quad (5.8)$$

subject to $0 \leq \alpha_i \leq P$, $i = 1, \dots, l$ and $\sum_{i=1}^l \alpha_i y_i = 0$.

5.1.2 Support Vector Regression

Originally, Support Vector Machine is designed for binary classification. Today, it has been effectively extended to multi-class problem. However, the predicted value of SVM is just from negative one to positive one. Support Vector Machine Regression (SVR), also can be called Support Vector Regression, is an approach extended from SVM to predicting real-valued outputs. It has been successfully applied in many applications, including financial forecasting (Yang, et al., 2002), bioinformatics (Zheng and Bailey, 2004), image recognition (Li, et al., 2000) and so on.

SVR uses the ε -insensitive loss function which is called ε -SV regression (Vapnik, 1995). The goal of ε -SV regression is to find a function that has at most ε deviation from the actually obtained targets for all the training data, and is as flat as possible at the same time.

Giving a set of training data $(x_1, y_1), \dots, (x_l, y_l)$, where $x_i \in R^n$, $y_i \in R^l$, and $i = 1, \dots, l$, suppose a linear function $f(x)$ is of the following form:

$$f(x) = w \bullet x + b \quad (5.9)$$

Where w is the weight vector, b is the bias. Slack variables ξ_i and ξ_i^* are introduced to account for the errors. Here, ξ_i computes the error for

Size Estimation for IS from their CDM *Estimating KLOC Using SVM*

underestimating the function while ξ_i^* computes the error for overestimating the function. The ε -insensitive loss function $|\xi|_\varepsilon$ is expressed as

$$|\xi|_\varepsilon = \begin{cases} 0, & \text{if } |\xi| \leq \varepsilon, \\ |\xi| - \varepsilon, & \text{otherwise.} \end{cases} \quad (5.10)$$

Then SVR solve the following problem:

$$\min \frac{1}{2} \|w\|^2 + P \sum_{i=1}^l (\xi_i + \xi_i^*) \quad (5.11)$$

subject to $-\varepsilon - \xi_i^* \leq w \bullet \phi(x_i) + b - y_i \leq \varepsilon + \xi_i$, $\xi_i, \xi_i^* \geq 0$ and $P > 0$, $i = 1, \dots, l$. Here, parameter P determines the trade-off between the flatness of $f(x)$ and the amount up to which deviation larger than ε is tolerated. x_i is the i th training example, and y_i is the output for the i th training example. $\phi(x)$ is a nonlinear function that maps x to a higher dimensional feature space.

The above problem can be transformed to the dual problem. Its solution is given by

$$f(x) = \sum_{i=1}^l (\alpha_i - \alpha_i^*) K(x_i \bullet x_j) + b \quad (5.12)$$

where the dual variables are constrained to $[0, P]$ and the kernel function (Smola and Scholkopf, 2004) is used to map nonlinear data to a higher dimensional space, it is $K(x_i, x_j) = \phi(x_i) \bullet \phi(x_j)$. The variable b can be computed as:

$$b = \begin{cases} y_i - w_i \bullet x_i - \varepsilon, & \text{for } \alpha_i \in (0, P), \\ y_i - w_i \bullet x_i + \varepsilon, & \text{for } \alpha_i^* \in (0, P). \end{cases}$$

Size Estimation for IS from their CDM *Estimating KLOC Using SVM*

SVR is also based on kernel functions similar to SVM. Although many new kernels has been proposed by researchers, the four basic kernels of SVR are still:

- Linear: $K(x_i, x_j) = x_i \bullet x_j$.
- Polynomial: $K(x_i, x_j) = (\gamma x_i \bullet x_j + r)^d$, $\gamma > 0$. Here d is the degree of the polynomial, larger d results in more complex polynomial kernel.
- Radial basis function (RBF): $K(x_i, x_j) = \exp(-\gamma \|x_i - x_j\|^2)$, $\gamma > 0$.
- Sigmoid: $K(x_i, x_j) = \tanh(\gamma x_i \bullet x_j + r)$.

Here, γ , r , and d are all kernel parameters. These parameters can be viewed as scale parameters that control the resolution of a kernel.

5.2 Model Building and Evaluation

5.2.1 Model Building

As described in the previous section, there are four basic kernels in SVR. Kernel should be selected in doing SVR experiments. The RBF kernel nonlinearly maps samples into a higher dimensional space. Unlike the linear kernel, it can handle the case when the relation between class labels and attributes is nonlinear. Furthermore, the linear kernel is a special case of RBF (Keerthi and Lin, 2003). Therefore, the linear kernel with a penalty parameter $\tilde{P}$ has the same performance as the RBF kernel with some parameters (P , γ). In addition, the sigmoid kernel behaves like RBF for certain parameters (Lin and Lin, 2003).

Size Estimation for IS from their CDM *Estimating KLOC Using SVM*

The other factor is the number of hyperparameters which influences the complexity of model selection. The polynomial kernel has more hyperparameters than the RBF kernel.

Finally, the RBF kernel has less numerical difficulties. One key point is $0 < K_{ij} \leq 1$ in contrast to polynomial kernels of which kernel values may go to infinity ($(\gamma x_i \bullet x_j + r > 1)$) or zero ($(\gamma x_i \bullet x_j + r < 1)$) while the degree is large. Moreover, we must note that the sigmoid kernel is not always valid (i.e. not the inner product of two vectors). Under some parameters, it may not be valid (Vapnik, 1995). RBF kernel are also applied in different research areas (Wang and Zhong, 2003; Xu and Zhang, 2004).

According to the rationales stated in the above discussion, in this project, Radial basis function (RBF) kernel is selected. In this research project, we use three vectors, vector-C, vector-R, vector- $\bar{A}$, that contain the total numbers of classes, the total numbers of relationships and the average attributes per class respectively for all the systems in a sample. That is, $X = (\text{vector-C}, \text{vector-R}, \text{vector-}\bar{A})$. In the model building, we select the value of P and γ in the range of 2^{-15} to 2^{15} . The goal is to identify good pair (P, γ) so that the testing data can be accurately predicted.

LIBSVM package (Chang and Lin, 2001) is one of the popular SVR tools and is selected to perform support vector regression in this project.

5.2.2 Evaluation

We evaluate the model built by collecting an independent dataset to validate it. We use mean magnitude of relative error and prediction level as the criteria

Size Estimation for IS from their CDM *Estimating KLOC Using SVM*

for validating the model built against the data collected in the evaluating dataset.

Mean magnitude of relative error (MMRE) and Prediction at level l (Pred(l)) (Conte, et al.,1986) are the most commonly used assessments for validation. MMRE is defined as follows:

$$MMRE = \frac{1}{n} \sum_{i=1}^n \left| \frac{y_i - \hat{y}_i}{y_i} \right| \quad (5.13)$$

where y_i and $\hat{y}_i$ are the i th real and the estimate values respectively of a project and n is the number of data points.

Pred(l) is defined as the ratio of number of cases in which the estimates are within the l absolute limit of the actual values divided by the total number of cases. A common accepted prediction level is 0.25. In our research project, Pred(0.25) is adopted. It is the ratio of number of cases in which the estimates are within the 25% of the actual values divided by the total number of cases.

The main criteria for validation lies in checking the values of MMRE and Pred(0.25) obtained from the evaluation dataset when the model built is used. A usual criterion for acceptable value of MMRE is less than 0.25. Lowest MMRE is preferred. And a standard criteria for acceptable value of Pred(0.25) is not less than 75%. Highest Pred(0.25) is preferred.

5.3 Experimental Results

The same pairs of datasets used in the multiple linear regression method are used for building and validating SVR models. The details of experiment design are shown in Appendix E. Five pairs of datasets are used in the experiments. Each pair has one dataset for model building and another dataset

Size Estimation for IS from their CDM *Estimating KLOC Using SVM*

for model validation. These pairs of datasets are: Industry VB-based Systems, Open-Source PHP-based Systems, Industry Java-based Systems, Open-Source Java-based Systems and the Combined Java-based Systems. The following sections will discuss the result of model building and evaluation for each pair of datasets.

5.3.1 Industry VB-based Systems

The dataset for training is shown in Table E-1. After training by selecting P and γ in the range from 2^{-15} to 2^{15} , good testing result can be obtained with the value of pair(P, γ) at (32, 0.000977).

The validation of the testing dataset is shown in Table 5-1. MMRE and Pred(0.25) computed are 0.16 and 0.88 respectively. These values fall within the acceptable level. Therefore, the evaluation result supports the validity of the LOC model built from the training dataset.

5.3.2 Open-Source PHP-based Systems

The dataset for training is shown in Table E-3. After training by selecting P and γ in the range from 2^{-15} to 2^{15} , good testing result can be obtained with the value of pair(P, γ) at (1024, 0.000086).

The validation of the testing dataset is shown in Table 5-2. MMRE and Pred(0.25) computed are 0.23 and 0.77 respectively. These values fall within the acceptable level. Therefore, the evaluation result supports the validity of the LOC model built from the training dataset.

Size Estimation for IS from their CDM *Estimating KLOC Using SVM*

Table 5-1 The SVR validation for VB-based LOC estimation model

System No	C	R	$\bar{A}$	Actual Size (KLOC)	Estimated Size (KLOC)	MRE
1	16	8	6.188	27.217	25.452	0.06
2	7	6	16.286	14.53	16.268	0.12
3	28	24	13.179	65.872	54.416	0.17
4	24	21	14.417	41.435	44.190	0.07
5	36	37	15.000	72	80.504	0.12
6	16	16	27.563	29.52	26.997	0.09
7	31	24	10.645	52.76	58.426	0.11
8	29	24	6.931	46.92	55.372	0.18
9	42	44	33.524	97.88	74.972	0.23
10	18	12	11.000	38.764	27.504	0.29
11	12	14	8.000	31.665	23.306	0.26
12	30	32	7.533	77.52	67.748	0.13
13	10	7	9.500	16.81	19.613	0.17
14	32	26	14.781	59.332	62.630	0.06
15	41	45	16.561	107.8	85.839	0.20
16	24	18	9.125	53.66	40.860	0.24
MMRE = 0.16, Pred(0.25) = 0.88						

5.3.3 **Java-based Systems**

There are three pairs of datasets collected for Java-Based systems: Industry Java-based systems, Open-Source Java-based systems and Combined Industry and Open-Source Java-based systems.

5.3.3.1 **Industry Java-based systems**

The dataset for training is shown in Table E-5. After training by selecting P and γ in the range from 2^{-15} to 2^{15} , good testing result can be obtained with the value of pair(P, γ) at (64, 0.000977).

Size Estimation for IS from their CDM *Estimating KLOC Using SVM*

Table 5-2 The SVR validation for Open-Source PHP-based LOC estimation model

System	C	R	$\bar{A}$	Actual Size (KLOC)	Estimated Size (KLOC)	MRE
ackerTodo	8	5	5.000	9.47	8.909	0.06
Bookmark4u	10	10	5.800	19.381	17.823	0.08
ByteMonsoon	6	2	8.000	4.219	4.432	0.05
Core	8	5	8.125	17.065	9.768	0.43
e107	34	12	7.912	65.649	54.469	0.17
eFiction	9	6	6.222	12.443	11.447	0.08
fdcl	5	2	11.600	3.856	4.684	0.21
galant	6	4	11.333	10.225	7.742	0.24
Infocentral	19	13	8.053	47.734	34.318	0.28
jamdb	4	5	8.500	4.554	6.384	0.40
openrealty	9	8	6.333	16.846	14.139	0.16
phpESP	16	15	5.438	33.645	32.707	0.03
Phpnews	6	5	6.167	6.532	7.428	0.14
phpollster	4	3	7.000	1.818	3.708	1.04
PhpScheduleIt	6	9	8.667	16.006	13.245	0.17
Phpsera	6	6	7.000	15.846	8.874	0.44
Phpwims	6	4	9.333	10.507	7.116	0.32
Phpwscookbook	5	5	10.200	7.607	7.724	0.02
plume	11	7	6.455	15.875	14.870	0.06
Quantum_Star_SE	30	23	9.533	57.643	70.464	0.22
Rasmp	10	4	6.600	9.452	9.961	0.05
Rimps	10	6	6.500	10.81	12.512	0.16
so-net	20	14	10.100	38.781	38.236	0.01
Supersurf	4	3	7.000	1.489	3.708	1.49
Usebb	6	3	13.333	6.118	7.229	0.18
videodb	14	10	4.643	24.227	21.976	0.09
Webaddressbook	3	2	20.000	8.363	6.241	0.25
wikiwig	15	9	5.800	23.457	22.016	0.06
yabbse	11	10	12.818	22.475	21.383	0.05
zebraz	6	5	7.833	8.59	7.881	0.08
ztml	10	3	9.600	11.724	9.600	0.18
MMRE = 0.23, Pred(0.25) = 0.77						

Size Estimation for IS from their CDM *Estimating KLOC Using SVM*

The validation of the testing dataset is shown in Table 5-3. MMRE and Pred(0.25) computed are 0.13 and 0.94 respectively. These values fall within the acceptable level. Therefore, the evaluation result supports the validity of the LOC model built from the training dataset.

Table 5-3 The SVR validation for Industry Java-based LOC estimation model

System No	C	R	$\bar{A}$	Actual Size (KLOC)	Estimated Size (KLOC)	MRE
1	49	22	5.898	84.89	91.579	0.08
2	12	7	4.250	20.446	17.965	0.12
3	8	6	7.750	17.29	15.772	0.09
4	16	12	5.125	34.075	27.358	0.20
5	18	14	6.389	40.113	33.022	0.18
6	17	14	7.412	28.722	32.434	0.13
7	31	27	6.871	71.37	70.014	0.02
8	30	29	10.300	60.34	73.320	0.22
9	11	11	5.091	20.45	21.658	0.06
10	35	36	8.486	93.27	82.722	0.11
11	19	20	4.789	31.69	42.978	0.36
12	30	33	6.267	77.52	73.867	0.05
13	21	24	7.714	45.384	53.703	0.18
14	24	28	6.000	52.1	61.682	0.18
15	7	9	4.857	19.36	17.248	0.11
16	18	26	5.944	56.744	52.616	0.07
MMRE = 0.13, Pred(0.25) = 0.94						

Size Estimation for IS from their CDM *Estimating KLOC Using SVM*

5.3.3.2 Open-Source Java-based systems

The dataset for training is shown in Table E-7. After training by selecting P and γ in the range from 2^{-15} to 2^{15} , good testing result can be obtained with the value of pair(P, γ) at (1024, 0.00061).

The validation of the testing dataset is shown in Table 5-4. MMRE and Pred(0.25) computed are 0.22 and 0.75 respectively. These values fall within the acceptable level. Therefore, the evaluation result supports the validity of the LOC model built from the training dataset.

5.3.3.3 Combined Industry and Open-Source Java-based systems

The dataset for training is shown in Table E-9. After training by selecting P and γ in the range from 2^{-15} to 2^{15} , good testing result can be obtained with the value of pair(P, γ) at (1024, 0.000073).

The validation of the testing dataset is shown in Table 5-5. MMRE and Pred(0.25) computed are 0.19 and 0.80 respectively. These values fall within the acceptable level. Therefore, the evaluation result supports the validity of the LOC model built from the training dataset.

5.4 Threats to Validity

Many organizations in the industry did not construct conceptual data models for information systems developed at the early stage of software development. Furthermore, to ensure that the data collected from a system is correct, we have to verify the data with the system documentations. Due to the view of confidentiality, many industry organizations did not allow us to access their system documentations to verify the accuracy of the data supplied by them.

Size Estimation for IS from their CDM *Estimating KLOC Using SVM*

Table 5-4 The SVR validation for Open-Source Java-based LOC estimation model

System	C	R	$\bar{A}$	Actual Size (KLOC)	Estimated Size (KLOC)	MRE
abaguibuilder	29	23	7.345	83.176	63.937	0.23
Art	14	7	6.929	17.67	19.706	0.12
Bofhms	12	4	4.333	10.507	12.856	0.22
Contineo	11	11	7.364	21.067	22.396	0.06
Dspace	32	20	4.250	47.57	63.109	0.33
Ejen	5	3	4.600	4.437	5.835	0.31
Imcms	59	30	4.712	91.424	113.044	0.24
Itracker	19	14	6.579	38.721	35.733	0.08
jdbforms	19	15	6.789	50.72	39.286	0.23
jmbase	8	8	5.250	19.723	14.718	0.25
jwma	22	19	8.045	63.228	48.087	0.24
Jbooks	11	7	3.727	12.817	15.543	0.21
Jfolder	10	8	12.800	29.288	19.809	0.32
Jgossip	14	9	5.714	28.047	21.968	0.22
Jpo	5	5	5.800	8.54	8.652	0.01
Jwnl	7	5	3.571	6.913	9.576	0.39
mp3cattle	9	10	3.444	14.598	17.930	0.23
openjms	17	16	6.824	59.72	36.198	0.39
Openhre	9	4	3.444	9.932	9.842	0.01
Quartz	11	11	4.364	22.954	21.294	0.07
sqlunit	26	20	7.808	74.089	55.443	0.25
tau_lastest	14	16	8.000	48.039	33.431	0.30
testsuite	15	9	5.067	30.076	22.910	0.24
Tesuji	2	1	16.000	8.01	6.032	0.25
MMRE = 0.22, Pred(0.25) = 0.75						

Size Estimation for IS from their CDM *Estimating KLOC Using SVM*

Table 5-5 The SVR validation for Combined Java-based LOC estimation model

System	C	R	$\bar{A}$	Actual Size (KLOC)	Estimated Size (KLOC)	MRE
abaguibuilder	29	23	7.345	83.176	62.202	0.25
Art	14	7	6.929	17.67	20.895	0.18
Bofhms	12	4	4.333	10.507	13.714	0.31
Contineo	11	11	7.364	21.067	22.787	0.08
Dspace	32	20	4.250	47.57	60.447	0.27
Ejen	5	3	4.600	4.437	5.539	0.25
Imcms	59	30	4.712	91.424	111.223	0.22
Itracker	19	14	6.579	38.721	35.931	0.07
jdbforms	19	15	6.789	50.72	37.396	0.26
jmbase	8	8	5.250	19.723	14.709	0.25
jwma	22	19	8.045	63.228	47.499	0.25
Jbooks	11	7	3.727	12.817	15.962	0.25
Jfolder	10	8	12.800	29.288	20.805	0.29
Jgossip	14	9	5.714	28.047	22.809	0.19
Jpo	5	5	5.800	8.54	8.380	0.02
Jwnl	7	5	3.571	6.913	9.323	0.35
mp3cattle	9	10	3.444	14.598	17.391	0.19
openjms	17	16	6.824	59.72	36.251	0.39
Openhre	9	4	3.444	9.932	10.121	0.02
Quartz	11	11	4.364	22.954	21.273	0.07
sqlunit	26	20	7.808	74.089	54.143	0.27
tau_lastest	14	16	8.000	48.039	33.230	0.31
testsuite	15	9	5.067	30.076	23.661	0.21
Tesuji	2	1	16.000	8.01	6.315	0.21
1	49	22	5.898	84.89	88.351	0.04
2	12	7	4.250	20.446	17.309	0.15
3	8	6	7.750	17.29	13.488	0.22
4	16	12	5.125	34.075	28.773	0.16
5	18	14	6.389	40.113	34.565	0.14
6	17	14	7.412	28.722	33.871	0.18
7	31	27	6.871	71.37	70.282	0.02
8	30	29	10.300	60.34	73.764	0.22
9	11	11	5.091	20.45	21.630	0.06
10	35	36	8.486	93.27	89.097	0.04
11	19	20	4.789	31.69	43.142	0.36
12	30	33	6.267	77.52	76.809	0.01
13	21	24	7.714	45.384	52.971	0.17

Size Estimation for IS from their CDM *Estimating KLOC Using SVM*

14	24	28	6.000	52.1	61.558	0.18
15	7	9	4.857	19.36	14.725	0.24
16	18	26	5.944	56.744	50.860	0.10
MMRE = 0.19, Pred(0.25) = 0.80						

This makes the data collection from industry extremely difficult. As a result, our industry datasets only have 16 data points. This may affect the accuracy of the SVR training and testing. To address this issue, in addition to the industry datasets, we have also collected some datasets from Open-Source systems to train and test the SVR models constructed in this project.

It is not possible for our samples to cover the full range of possible LOC values well. Thus, the models that we have built are not appropriate for estimating systems with LOC values that are significantly larger or smaller than the LOC values in the respective samples. The current evaluation criteria of using MMRE and PRED(0.25) for building models is also not perfect (Stensrud, et al., 2002).

SVM also has some limitations. One limitation lies in choice of the kernel and the selection of the kernel function parameters. Another limitation of SVM is the speed. Training and testing data all require much time.

In summary, we do not claim that we have built models that are readily for use. However, in the worst case, at least, our work has shown that it is useful and promising to experiment the proposed method on much larger scale to address the critical software sizing problem in the industry. Clearly, such experiment requires organized industry and research community cooperation on a long term basis.

5.5 Summary

SVM has been a commonly used in classification and regression problems. SVM has been successfully applied in many areas from data mining to bioinformatics. This chapter uses SVM to build software size estimation model based on the size model proposed in Chapter 3. The results of the experiments are encouraging and give strong support to the software size estimation model proposed.

The success of SVM method depends on the tuning of several parameters which affect the generalization error. Till now, there is no any tool that can give an effective way to tune the parameters of support vector regression function. It is a problem that further research can be explored.

Threats to validity were also stated in this chapter. The sizes of the industry datasets are not very sufficient because it is quite difficult to collect data from the industry. Thus, it may affect the accuracy of the validation. Furthermore, the datasets collected from Open-Source systems contain the data from reengineering the systems source codes and database schemas. Thus, the data may be more accurate than the data collected at the early stage of software development. Therefore, model built from Open-Source systems also face some threat to validity. The models we built currently may not be readily for use. However, at least, our work has shown that the proposed software sizing model is useful. It is worthwhile for the industry to carry out much larger scale experiments to build software size estimation model based on the proposed method. The result of this may address the critical software estimating problems in the industry.

Chapter 6 Estimating KLOC Using Artificial Neural Network

6.1 Artificial Neural Network

6.1.1 Overview

The branch of artificial intelligence called neural networks dates back to the 1940s, when McCulloch and Pitts developed the first neural model in 1943. Artificial Neural Network (ANN) has developed rapidly since the 1980s. From then on, ANN is a durative issue in research and is applied in many areas such as prediction of time series (Hoptroff, 1993; Tang and Fishwick, 1993), financial phenomena (Dutta and Shekhar, 1988; Refenes, 1993) and software cost estimation (Huang, 2003).

The ANN is an information processing system simulating the structure and functions of the human brain. It consists of numerous simple processing elements connected together according to certain rules and is able to respond dynamically to an outside stimulus and process information. Like the human brain, the structure and processing sequence of ANN are parallel. The ANN has a very strong learning ability and can adapt itself to the outside

Size Estimation for IS from their CDM *Estimating KLOC Using ANN*

environment by learning. In an ANN, knowledge is not stored in specific memories, but distributed in the whole system. The ANN can give an approximately optimal solution from limited and distorted information if it is properly trained.

Artificial neural networks are purely data driven models which through training iteratively transition from a random state to a final model. They do not depend on assumptions about functional form, probability distribution or smoothness, and have been proven to be universal approximators (Funahashi, 1989).

However, there is no universally accepted definition of an ANN. But most people may agree that an ANN is a network of many simple processors, units, each possibly having a small amount of local memory. The units are connected by communication channels (connections) which usually carry numeric (as opposed to symbolic) data, encoded by any of various means. The units operate only on their local data and on the inputs they receive via the connections. The restriction to local operations is often relaxed during training.

6.1.2 Neural Model and Network Architecture

An artificial neural network consists of a set of simple processing units which communicate by sending signals to each other over a large number of weighted connections. These processing units are called neurons. A neuron receives a number of inputs which are from neighbors or external sources and uses these to compute an output signal which is propagated to other neurons. Each input has an associated weight which can be modified. The activation signal is passed through an activation function which is also called a transfer

Size Estimation for IS from their CDM *Estimating KLOC Using ANN*

function to produce the output of the neuron. A neuron with a single input is shown in Figure 6-1.

Figure 6-1 A single-input Neuron

In Figure 6-1, the scalar input p is transmitted through a connection that multiplies its strength by the scalar weight w to form the scalar wp . Here the weighted input wp is the only argument of the transfer function f , which produces the scalar output a . The transfer function net input n is the sum of the weighted input wp and the bias b . This sum is the argument of the transfer function f . Here f is a transfer function, typically a step function or a sigmoid function, which takes the argument n and produces the output a . Note that w and b are both adjustable scalar parameters of the neuron. The central idea of neural networks is that such parameters can be adjusted so that the network exhibits some desired or interesting behavior.

A neuron can have a single R -element input vector $p_1, p_2, \dots, p_R$ that each individual element inputs are multiplied by weights $w_{1,1}, w_{1,2}, \dots, w_{1,R}$ and the weighted values are fed to the summing junction. A multiple-input neuron is shown in Figure 6-2.

Size Estimation for IS from their CDM *Estimating KLOC Using ANN*

Figure 6-2 A multiple-input Neuron

Two or more neurons can be combined in a layer. An ANN can have several layers. Usually a neural network has three layers: input layer, hidden layer and output layer. It can have more than one hidden layers based on the experiment requirements. Within each layers, neurons usually have the same type of activation functions and the same pattern of connections to other neural layers. Figure 6-3 gives the example of a multiple-layer neural network which has one input layer, one output layer and one hidden layer. Each layer has a weight matrix W , a bias vector b and an output vector a . Between the layers, transfer functions f are set.

A neural network is characterized by its architecture (connection pattern between the neurons), its method to determine the connection weights (training algorithm), and its activation functions or transfer functions. Normally, neurons within a layer are not interconnected at all, and neurons between two contiguous layers are fully interconnected. If there is no other connection, we call this kind of neural networks are feed forward neural

Size Estimation for IS from their CDM *Estimating KLOC Using ANN*

networks. If part of the input features depend on the result of the output layer, or there are feedbacks from the upper neural layer to the lower neural layer, we call this kind of neural networks are recurrent neural networks.

In general, there are several neural network architectures: Perceptron neural networks, Winner-Take-All neural networks, Dynamic feedback neural networks, Self-Organization Mapping neural networks, Adaptive Resonance Theory (ART) neural networks and Bayesian neural networks.

Figure 6-3 A multiple-layer Neural Network

The development of a neural model is begun by first developing an appropriate layout of neurons, or connections between network nodes. This includes defining the number of layers of neurons, the number of neurons within each layer, and the manner in which they are all linked. The weighted estimating functions between the nodes and the specific training algorithm to be used must also be determined.

Size Estimation for IS from their CDM *Estimating KLOC Using ANN*

Learning is an important task of neural networks. It refers to the rules, which govern the updating of parameters within neurons and between neurons. Supervised learning and unsupervised learning are two major learning methods that can be used for adaptive ANN:

- *Supervised learning*: The network is trained by providing it with input and matching output patterns. These input-output pairs can be provided by an external teacher or by the system which contains the network (self-supervised). During the learning process global information may be required. Paradigms of supervised learning include error-correction learning, reinforcement learning and stochastic learning.
- *Unsupervised learning*: An output is trained to respond to clusters of pattern within the input. In this paradigm the system is supposed to discover statistically salient features of the input populations. Paradigms of unsupervised learning are Hebbian learning and competitive learning. Unlike the supervised learning paradigm, there is no a priori set of categories into which the patterns are to be classified. Usually, supervised learning is performed off-line, whereas unsupervised learning is performed on-line.

6.2 Model Building and Evaluation

Comparing to multiple linear regression method, ANN does not care the data distribution and the linear relationship between independent variables and dependent variables. It is applicable to multivariate non-linear problems. ANN also does not care about the transformations of the variables since these transformations can be automated done in the computational process. Thus

Size Estimation for IS from their CDM *Estimating KLOC Using ANN*

ANN is adopted as one of the method to build the software size estimation models in this research.

6.2.1 Rationale for Selecting the Neural Network Architecture

As described in section 6.1.2, there are various artificial neural network architectures. Among those architectures, the feed-forward multilayer Perceptron with the back-propagation training algorithm is the most common type of neural network architecture. This type of artificial neural network architecture has been widely used in many areas, such as software reliability assessment (Su and et al., 2005), software quality prediction (Wang and et al., 2005), biomedicine (Catley and et al., 2006; Ranganathan and Gunasekaran, 2006), measurement (El-Sheimy and et al., 2006) and data preparation (Yu and et al., 2006). This type of neural network architecture is also commonly adopted in software estimation area. Many researchers have applied this neural network approach to estimate software effort and cost (Jorgensen, 1995; Witting and Finnie, 1997; Idri and et al., 2002; Tadayon, 2005). As they pointed out, for software estimation, the feed-forward multilayer Perceptron is the most commonly adopted artificial neural network architecture, and the back-propagation algorithm is the most commonly used learning algorithm. Thus, in this project, the feed-forward multilayer Perceptron with the back-propagation training algorithm is adopted as this is the primary research to prove whether conceptual data model can be used to estimate software size by using neural network. However, other neural network architectures and other training algorithms may also be suitable in this project. The result may be more accurate. We will consider it as a future work.

Size Estimation for IS from their CDM *Estimating KLOC Using ANN*

6.2.2 Neural Network Construction

The ANN constructed in this project consisted one input layer, one hidden layer and one output layer. There are three neurons in input layer and one neuron in output layer. As we proposed the software sizing model in Chapter 3, three inputs are the total number of classes in the conceptual data model (C), the total number of relationships (R) and the average number of attributes per class ($\bar{A}$) respectively.

The training function used in this experiment is *trainbfg*, BFGS quasi-Newton algorithm. There are many transfer functions in ANN, after studying, transfer function *tan-sigmoid* is used in the hidden layer. The transfer functions *pureline* is carried out in the output layer. For tuning the system, we run the system by tuning the neural network parameters and the number of neurons in hidden layer. Networks were trained to a maximum error of 0.1 for each construction data point, a maximum number of iterations through the construction set (epochs) of 5000.

6.2.3 Validation Criteria

Mean magnitude of relative error (MMRE) and Prediction at level l (Pred(l)) (Conte, et al.,1986) are the most commonly used assessments for validation. MMRE is defined as follows:

$$MMRE = \frac{1}{n} \sum_{i=1}^n \left| \frac{y_i - \hat{y}_i}{y_i} \right| \quad (6.1)$$

where y_i and $\hat{y}_i$ are the i th real and the estimate values respectively of a project. n is the number of data points.

Size Estimation for IS from their CDM *Estimating KLOC Using ANN*

$Pred(l)$ is defined as the ratio of number of cases in which the estimates are within the l absolute limit of the actual values divided by the total number of cases. A common accepted prediction level is 0.25. In this project, $Pred(0.25)$ is adopted. It is the ratio of number of cases in which the estimates are within the 25% of the actual values divided by the total number of cases.

The main criteria for validation lies in obtaining acceptable values of MMRE and $Pred(0.25)$. A usual criterion for acceptable value of MMRE is less than 0.25. Lowest MMRE is preferred. And a standard criteria for acceptable value of $Pred(0.25)$ is not less than 75%. Highest $Pred(0.25)$ is preferred.

6.3 Experimental Results

The same pairs of datasets used in the multiple linear regression method are used for building and validating ANN models. The details of experiment design are shown in Appendix E. The following sections will discuss the result of model building and evaluation for each pair of datasets.

6.3.1 Industry VB-based Systems

The dataset for training is shown in Table E-1. After tuning the neural network parameters, we obtain good testing result. The number of neurons in hidden layer is 25. The value of epochs is 1000.

The validation of the testing dataset is shown in Table 6-1. MMRE and $Pred(0.25)$ computed are 0.14 and 0.94 respectively. These values fall within the acceptable level. Therefore, the evaluation result supports the validity of the LOC model built from the training dataset.

Size Estimation for IS from their CDM *Estimating KLOC Using ANN*

Table 6-1 The ANN validation for VB-based LOC estimation model

System No	C	R	$\bar{A}$	Actual Size (KLOC)	Estimated Size (KLOC)	MRE
1	49	22	5.898	27.217	23.124	0.15
2	12	7	4.250	14.53	15.676	0.08
3	8	6	7.750	65.872	55.159	0.16
4	16	12	5.125	41.435	44.780	0.08
5	18	14	6.389	72	86.473	0.20
6	17	14	7.412	29.52	28.698	0.03
7	31	27	6.871	52.76	60.451	0.15
8	30	29	10.300	46.92	57.135	0.22
9	11	11	5.091	97.88	102.049	0.04
10	35	36	8.486	38.764	27.920	0.28
11	19	20	4.789	31.665	23.702	0.25
12	30	33	6.267	77.52	69.893	0.10
13	21	24	7.714	16.81	17.986	0.07
14	24	28	6.000	59.332	64.792	0.09
15	7	9	4.857	107.8	102.289	0.05
16	18	26	5.944	53.66	41.449	0.23
MMRE = 0.14, Pred(0.25) = 0.94						

6.3.2 **Open-Source PHP-based Systems**

The dataset for training is shown in Table E-3. After tuning the neural network parameters, we obtain good testing result. The number of neurons in hidden layer is 25. The value of epochs is 1000.

The validation of the testing dataset is shown in Table 6-2. MMRE and Pred(0.25) computed are 0.18 and 0.81 respectively. These values fall within the acceptable level. Therefore, the evaluation result supports the validity of the LOC model built from the training dataset.

Size Estimation for IS from their CDM *Estimating KLOC Using ANN*

Table 6-2 The ANN validation for Open-Source PHP-based LOC estimation model

System	C	R	$\bar{A}$	Actual Size (KLOC)	Estimated Size (KLOC)	MRE
ackerTodo	8	5	5.000	9.47	7.765	0.18
Bookmark4u	10	10	5.800	19.381	17.104	0.12
ByteMonsoon	6	2	8.000	4.219	5.079	0.20
Core	8	5	8.125	17.065	14.761	0.14
e107	34	12	7.912	65.649	61.579	0.06
eFiction	9	6	6.222	12.443	14.434	0.16
Fdcl	5	2	11.600	3.856	4.735	0.23
galant	6	4	11.333	10.225	8.196	0.20
Infocentral	19	13	8.053	47.734	38.149	0.20
jamdb	4	5	8.500	4.554	5.805	0.27
openrealty	9	8	6.333	16.846	13.780	0.18
phpESP	16	15	5.438	33.645	44.041	0.31
Phpnews	6	5	6.167	6.532	5.977	0.09
phpollster	4	3	7.000	1.818	2.290	0.26
PhpScheduleIt	6	9	8.667	16.006	10.349	0.35
Phpsera	6	6	7.000	15.846	18.277	0.15
Phpwims	6	4	9.333	10.507	6.210	0.41
Phpwscookbook	5	5	10.200	7.607	5.100	0.33
plume	11	7	6.455	15.875	12.459	0.22
Quantum_Star_SE	30	23	9.533	57.643	65.108	0.13
Rasmp	10	4	6.600	9.452	7.779	0.18
Rimps	10	6	6.500	10.81	10.183	0.06
so-net	20	14	10.100	38.781	47.456	0.22
Supersurf	4	3	7.000	1.489	1.627	0.09
Usebb	6	3	13.333	6.118	5.099	0.17
videodb	14	10	4.643	24.227	21.441	0.12
Webaddressbook	3	2	20.000	8.363	9.829	0.18
wikiwig	15	9	5.800	23.457	22.054	0.06
yabbse	11	10	12.818	22.475	25.419	0.13
zebraz	6	5	7.833	8.59	7.292	0.15
Ztml	10	3	9.600	11.724	10.909	0.07
MMRE = 0.18, Pred(0.25) = 0.81						

Size Estimation for IS from their CDM *Estimating KLOC Using ANN***6.3.3 Java-based Systems**

There are three pairs of datasets collected for Java-based systems: Industry Java-based systems, Open-Source Java-based systems and Combined Industry and Open-Source Java-based systems.

6.3.3.1 Industry Java-based systems

The dataset for training is shown in Table E-5. After tuning the neural network parameters, we obtain good testing result. The number of neurons in hidden layer is 20. The value of epochs is 1000.

The validation of the testing dataset is shown in Table 6-3. MMRE and Pred(0.25) computed are 0.15 and 0.81 respectively. These values fall within the acceptable level. Therefore, the evaluation result supports the validity of the LOC model built from the training dataset.

6.3.3.2 Open-Source Java-based systems

The dataset for training is shown in Table E-7. After tuning the neural network parameters, we obtain good testing result. The number of neurons in hidden layer is 20. The value of epochs is 1000.

The validation of the testing dataset is shown in Table 6-4. MMRE and Pred(0.25) computed are 0.20 and 0.75 respectively. These values fall within the acceptable level. Therefore, the evaluation result supports the validity of the LOC model built from the training dataset.

Size Estimation for IS from their CDM *Estimating KLOC Using ANN*

Table 6-3 The ANN validation for Industry Java-based LOC estimation model

System No	C	R	$\bar{A}$	Actual Size (KLOC)	Estimated Size (KLOC)	MRE
1	49	22	5.898	84.89	88.752	0.05
2	12	7	4.250	20.446	21.160	0.03
3	8	6	7.750	17.29	19.664	0.14
4	16	12	5.125	34.075	27.202	0.20
5	18	14	6.389	40.113	32.333	0.19
6	17	14	7.412	28.722	30.992	0.08
7	31	27	6.871	71.37	80.292	0.13
8	30	29	10.300	60.34	81.457	0.35
9	11	11	5.091	20.45	22.288	0.09
10	35	36	8.486	93.27	87.940	0.06
11	19	20	4.789	31.69	42.865	0.35
12	30	33	6.267	77.52	83.798	0.08
13	21	24	7.714	45.384	56.346	0.24
14	24	28	6.000	52.1	69.896	0.34
15	7	9	4.857	19.36	19.901	0.03
16	18	26	5.944	56.744	51.921	0.09
MMRE = 0.15, Pred(0.25) = 0.81						

6.3.3.3 **Combined Industry and Open-Source Java-based systems**

The dataset for training is shown in Table E-9. After tuning the neural network parameters, we obtain good testing result. The number of neurons in hidden layer is 30. The value of epochs is 1000.

The validation of the testing dataset is shown in Table 6-5. MMRE and Pred(0.25) computed are 0.19 and 0.75 respectively. These values fall within the acceptable level. Therefore, the evaluation result supports the validity of the LOC model built from the training dataset.

Size Estimation for IS from their CDM *Estimating KLOC Using ANN*

Table 6-4 The ANN validation for Open-Source Java-based LOC estimation model

System	C	R	$\bar{A}$	Actual Size (KLOC)	Estimated Size (KLOC)	MRE
abaguibuilder	29	23	7.345	83.176	64.092	0.23
Art	14	7	6.929	17.67	20.831	0.18
Bofhms	12	4	4.333	10.507	12.373	0.18
Contineo	11	11	7.364	21.067	22.672	0.08
Dspace	32	20	4.250	47.57	57.238	0.20
Ejen	5	3	4.600	4.437	5.464	0.23
Imcms	59	30	4.712	91.424	112.224	0.23
Itracker	19	14	6.579	38.721	35.564	0.08
jdbforms	19	15	6.789	50.72	37.171	0.27
jmbase	8	8	5.250	19.723	14.258	0.28
Jwma	22	19	8.045	63.228	48.297	0.24
Jbooks	11	7	3.727	12.817	14.620	0.14
Jfolder	10	8	12.800	29.288	21.232	0.28
Jgossip	14	9	5.714	28.047	22.581	0.19
Jpo	5	5	5.800	8.54	8.163	0.04
Jwnl	7	5	3.571	6.913	8.912	0.29
mp3cattle	9	10	3.444	14.598	15.811	0.08
openjms	17	16	6.824	59.72	35.959	0.40
Openhre	9	4	3.444	9.932	9.761	0.02
Quartz	11	11	4.364	22.954	19.901	0.13
sqlunit	26	20	7.808	74.089	55.396	0.25
tau_lastest	14	16	8.000	48.039	33.649	0.30
testsuite	15	9	5.067	30.076	22.544	0.25
Tesuji	2	1	16.000	8.01	6.573	0.18
MMRE = 0.20, Pred(0.25) = 0.75						

Size Estimation for IS from their CDM *Estimating KLOC Using ANN*

Table 6-5 The ANN validation for Combined Java-based LOC estimation model

System	C	R	$\bar{A}$	Actual Size (KLOC)	Estimated Size (KLOC)	MRE
abaguibuilder	29	23	7.345	83.176	63.214	0.24
Art	14	7	6.929	17.67	14.136	0.20
Bofhms	12	4	4.333	10.507	10.297	0.02
Contineo	11	11	7.364	21.067	18.960	0.10
Dspace	32	20	4.250	47.57	35.202	0.26
Ejen	5	3	4.600	4.437	4.215	0.05
Imcms	59	30	4.712	91.424	75.882	0.17
Itracker	19	14	6.579	38.721	35.236	0.09
jdbforms	19	15	6.789	50.72	37.026	0.27
jmbase	8	8	5.250	19.723	15.581	0.21
jwma	22	19	8.045	63.228	48.053	0.24
Jbooks	11	7	3.727	12.817	9.356	0.27
Jfolder	10	8	12.800	29.288	22.552	0.23
Jgossip	14	9	5.714	28.047	22.718	0.19
Jpo	5	5	5.800	8.54	6.320	0.26
Jwnl	7	5	3.571	6.913	2.765	0.60
mp3cattle	9	10	3.444	14.598	11.824	0.19
openjms	17	16	6.824	59.72	35.832	0.40
Openhre	9	4	3.444	9.932	7.449	0.25
Quartz	11	11	4.364	22.954	20.888	0.09
sqlunit	26	20	7.808	74.089	54.826	0.26
tau_lastest	14	16	8.000	48.039	33.627	0.30
testsuite	15	9	5.067	30.076	23.159	0.23
Tesuji	2	1	16.000	8.01	6.328	0.21
1	49	22	5.898	84.89	82.343	0.03
2	12	7	4.250	20.446	17.584	0.14
3	8	6	7.750	17.29	15.215	0.12
4	16	12	5.125	34.075	27.942	0.18
5	18	14	6.389	40.113	34.096	0.15
6	17	14	7.412	28.722	23.839	0.17
7	31	27	6.871	71.37	70.656	0.01
8	30	29	10.300	60.34	43.445	0.28
9	11	11	5.091	20.45	19.428	0.05
10	35	36	8.486	93.27	92.337	0.01
11	19	20	4.789	31.69	20.915	0.34
12	30	33	6.267	77.52	75.194	0.03
13	21	24	7.714	45.384	36.761	0.19

Size Estimation for IS from their CDM *Estimating KLOC Using ANN*

14	24	28	6.000	52.1	41.159	0.21
15	7	9	4.857	19.36	15.488	0.20
16	18	26	5.944	56.744	51.070	0.10
MMRE = 0.19, Pred(0.25) = 0.75						

6.4 Threats to Validity

Many organizations in the industry did not construct conceptual data models for information systems developed in the early stage of software development. Furthermore, to ensure that the data collected from a system is correct, we have to verify the data with the system documentations. Furthermore, due to the view of confidentiality, many industry organizations did not allow us to access their system documentations to verify the accuracy of the data supplied by them. This makes the data collection from industry extremely difficult. As a result, our industry datasets only have 16 data points. This may affect the accuracy of the neural network training and testing. To address this issue, in addition to the industry datasets, we have also collected some datasets from Open-Source systems to train and test the neural network constructed in this project.

Although steps have been taken to ensure that only conceptual entities but not implementation data are included in the models, the conceptual data models collected from Open-Source are more accurate than the one collected from the industry. Therefore, models built from Open-Source samples also face some threat to validity.

However, ANN method itself still has some disadvantages. First of all, ANN cannot provide a clear equation to characterize the relationships among the variables. Secondly, it requires a great deal of computational effort compared

Size Estimation for IS from their CDM *Estimating KLOC Using ANN*

to multiple linear regression method. And it also needs a large size sample data to training the network.

Furthermore, surely, it is not possible for our samples to cover the full range of possible LOC values well. So, the models that we have built are not appropriate for estimating systems with LOC values that are significantly larger or smaller than the LOC values in the respective samples. The current evaluation criteria of using MMRE and PRED(0.25) for building models is also not perfect (Stensrud, et al., 2002).

In summary, we do not claim that we have built models that are readily for use. However, in the worst case, at least, our work has shown that it is useful and promising to experiment the proposed method on much larger scale to address the critical software sizing problem in the industry. Clearly, such experiment requires organized industry and research community cooperation on a long term basis.

6.5 Summary

Artificial Neural Network has been applied in many areas, including software cost estimation (Wittig and Finnie, 1994). In this chapter, we apply the Artificial Neural Network to build software size estimation model based on the proposed method. We successfully build the proposed software sizing models through the neural network's parameters tuning. All the validation based on the early dataset support the models built.

Threats to validity were also discussed in this chapter. The data collection from the industry is a quite difficult work. Thus, insufficient dataset may affect the accuracy of neural network training and testing. Furthermore, because reengineering is done during the data collection from the

Size Estimation for IS from their CDM *Estimating KLOC Using ANN*

Open-Source systems, these data may be more accurate than the data collected at the early stage of software development. Therefore, model built from Open-Source systems also face some threat to validity. The models we built currently may not be readily for use. However, at least, our work has shown that it is useful and promising to experiment the proposed software estimation method on much larger scale to address the critical software sizing problem in the industry.

Size Estimation for IS from their CDM Comparison with the FP Method

Chapter 7 Comparison of the Proposed Method with the FP Method

7.1 Function Point Analysis

Function Point Analysis (FPA) is established by Allan Albrecht of IBM and was first published in 1979 (Albrecht, 1979). In 1984, the International Function Point Users Group (IFPUG) was set up to establish the rules, set standards, and educate their use and manage the evolution (Abran and Robillard, 1996). FPA provides a standardized methodology for measuring the size of software through the various functions of a software application.

The core of FPA is the metric Function Points (FP) for measuring software size. The function points (FPs) are obtained by analysing the software application from the following two different perspectives:

- The functional size (UFP): it is calculated by assigning weights to each individual type of functions. Functional size is measured by the measure, unadjusted function points (UFP).

Size Estimation for IS from their CDM Comparison with the FP Method

- The valued adjustment factor (VAF): It is used to measure the environment and processing complexity of the software application. VAF is calculated by using general systems characteristics (GSC).

Thus Adjusted Function Points (FP) can be computed by using the value of VAF to adjust the functional size UFP. The computation will be discussed towards the end of this section.

The counting of the functional size, UFP, is based on the following five elements (Garmus and Herron, 2000):

- External Input (EI): An external input is an elementary process of the software application that processes data or control information that receives from the external environment of the application;
- External Output (EO): An external output is an elementary process of the software application that generates data or control information that exits from the software application to its external environment;
- External Query (EQ): An external inquiry is an elementary process of the software application that results in retrieval of data or control information. The information is delivered to the external environment of the software application;
- Internal Logical File (ILF): An internal logical file is a user-identifiable group of logically related data or control information that resides entirely within the application boundary and is maintained through external inputs;
- External Interface File (EIF): An external interface file is a user-identifiable group of logically related data or control information that is used for reference purpose only. The data resides entirely

Size Estimation for IS from their CDM Comparison with the FP Method

outside the application and is maintained by another application. The external interface file is an internal logical file for another application.

The VAF is based on 14 general system characteristics (GSC) that measure the general functionality of the application being counted. Each characteristic has associated descriptions that help determine the degrees of influence of the characteristics. The degrees of influence range on a scale of zero to five, from no influence to strong influence. These GSC in FPA are: data communications, distributed data processing, performance, heavily used configuration, transaction rate, on-line data entry, end-user efficiency, on-line update, complex processing, reusability, installation ease, operational ease, multiple ease and facilitate change.

The Function Point Analysis (FPA) is a well-known method for software sizing including the estimation of LOC. Indeed, since the proposed methods in Chapter 4, 5 and 6 are based on the conceptual data model, it has a clear advantage over the FP method on having the required information more readily available at the early stage of software development. However, for additional information, we compared the LOC estimation from the proposed methods with the FP method based on Open-Source datasets. Such comparison is not possible for industry datasets due to the unavailability of their FPs.

In the comparison, for the FP method, we used adjusted Function Point (FP):

$$FP = VAF \times UFP \quad (7.1)$$

A value adjustment factor (VAF) is calculated as follows:

$$VAF = (TDI \times 0.01) + 0.65 \quad (7.2)$$

Size Estimation for IS from their CDM Comparison with the FP Method

The TDI is calculated as the total of all the degrees of influence (each of them is ranked from 0 to 5) by the 14 general system characteristics (Garmus and Herron, 2000).

7.2 Comparison of the Proposed Method with the FP

Method

Function Points of the systems collected in this research project are counted based on UFPs and VAFs through reading the system documents and conducting testing. We followed the counting rules proposed by (Garmus and Herron, 2000; British Standards, 2003).

7.2.1 Comparison Based on PHP Open-Source Systems

For PHP based systems, since there is no existing formula available for estimating LOC directly from Function Point, we built models respectively in the same way as we did for the proposed methods by using three techniques, Multiple Linear Regression (MLR), Support Vector Machine (SVM) and Artificial Neural Network (ANN). Therefore, UFPs and VAFs were counted for both datasets in the pair of datasets for PHP based Open-Source system. The FPs and related data for the two datasets are shown in Table 7-1 and Table 7-2.

Size Estimation for IS from their CDM Comparison with the FP Method

Table 7-1 The FP data for the model building dataset for Open-Source

PHP-based systems

System Name	UFP	VAF	FP
Bannerex3a	126	1.01	127
Castor	476	1.01	481
Cmsmadesimple	634	1.02	647
Comendar	337	1.02	344
Commerce	1517	1.03	1563
Coppermine	447	1.02	456
EclipseBB	1190	1.02	1214
Jdcms	99	1.01	100
Linkbase	172	1.03	177
Linha	1003	1.02	1023
Lotgd	1045	1.01	1055
Mailwatch	1183	1.01	1195
Mantis	1240	1.03	1277
Mundimail	90	1.01	91
opendocman	246	1.01	248
Openrating	265	1.02	270
php4Flicks	119	1.01	120
Phpalumni	201	1.02	205
Phpcollegeex	183	1.01	185
Phpman	249	1.03	256
phpmylibrary	549	1.03	565
phpstudentcenter	244	1.03	251
phptimesheet	140	1.02	143
Poppawid	304	1.02	310
Refbase	283	1.01	286
Replex	103	1.01	104
Timeclock	81	1.01	82
Uccass	782	1.03	805
Uma	120	1.01	121
Vallheru	1065	1.01	1076
WebFileSystem	210	1.01	212
Winventory	212	1.01	214

Size Estimation for IS from their CDM Comparison with the FP Method

Table 7-2 The FP data for the validation dataset for Open-Source PHP-based systems

System Name	UFP	VAF	FP
ackerTodo	205	1.02	209
Bookmark4u	421	1.04	438
ByteMonsoon	129	1.01	130
Core	256	1.02	261
E107	1023	1.01	1033
eFiction	245	1.02	250
fdcl	121	1.01	122
galant	182	1.01	184
Infocentral	684	1.01	691
jamdb	132	1.03	136
openrealty	312	1.03	321
phpESP	659	1.03	679
Phpnews	193	1.03	199
phpollster	87	1.02	89
PhpScheduleIt	274	1.04	285
Phpsera	257	1.01	260
Phpwims	218	1.02	222
Phpwscookbook	169	1.02	172
plume	337	1.02	344
Quantum_Star_SE	898	1.02	916
Rasmp	224	1.01	226
Rimps	262	1.01	265
so-net	747	1.02	762
Supersurf	75	1.03	77
usebb	163	1.01	165
videodb	446	1.02	455
Webaddressbook	177	1.02	181
wikiwig	438	1.02	447
yabbse	405	1.03	417
zebraz	156	1.02	159
ztml	211	1.01	213

Size Estimation for IS from their CDM Comparison with the FP Method**7.2.1.1 Comparison of proposed MLR method with the FP Method**

In the same manner, we built a linear regression model from the dataset that built the proposed model in Chapter 4 successfully for estimating LOC from FP. Since there is only one independent variable, FP, simple linear regression model is constructed. We used the resulting model to estimate LOCs for the systems in the validation dataset. The FP model built is as follows:

$$Size = -2.485 + 0.055FP \quad (8.3)$$

With the use of the FP model built to estimate LOC of systems in the validation dataset (shown in Table 7-3), MMRE and Pred(0.25) computed are 0.14 and 0.77 respectively. Note that for use of the LOC estimation model built from the proposed method using the linear regression, MMRE and Pred(0.25) computed for the same dataset are 0.15 and 0.81 respectively. There is no clear difference between the results from the two methods.

7.2.1.2 Comparison of proposed SVR method with the FP Method

We trained the dataset used in model building in the same way as described in Chapter 5 and get the testing result for estimating LOC from FP. After training by selecting P and γ in the range from 2^{-15} to 2^{15} , good testing result can be obtained with the value of pair(P, γ) at (1024, 0.000086). The validation result for the testing dataset from FP is shown in Table 8-4. MMRE and Pred(0.25) computed are 0.15 and 0.84 respectively. As stated in Chapter 5, MMRE and Pred(0.25) computed for the same dataset are 0.23 and 0.77 respectively by using of the LOC estimation model built from the proposed method using the SVR. There is no clear difference between the results from the two methods.

Size Estimation for IS from their CDM Comparison with the FP Method

Table 7-3 The FP validation dataset result of using linear regression for
Open-Source PHP-based systems

System	Actual Size (KLOC)	Estimated Size (KLOC)	MRE
ackerTodo	9.47	9.016	0.05
Bookmark4u	19.381	21.596	0.11
ByteMonsoon	4.219	4.681	0.11
Core	17.065	11.877	0.26
E107	65.649	54.343	0.17
eFiction	12.443	11.260	0.10
fdcl	3.856	4.237	0.10
galant	10.225	7.625	0.25
Infocentral	47.734	35.511	0.26
jamdb	4.554	4.993	0.10
openrealty	16.846	15.190	0.10
phpESP	33.645	34.847	0.04
Phpnews	6.532	8.448	0.29
phpollster	1.818	2.396	0.32
PhpScheduleIt	16.006	13.188	0.18
Phpsera	15.846	11.791	0.26
Phpwims	10.507	9.745	0.07
Phpwscookbook	7.607	6.996	0.08
plume	15.875	16.421	0.03
Quantum_Star_SE	57.643	47.893	0.17
Rasmp	9.452	9.958	0.05
Rimps	10.81	12.069	0.12
so-net	38.781	39.422	0.02
Supersurf	1.489	1.764	0.18
usebb	6.118	6.570	0.07
videodb	24.227	22.536	0.07
Webaddressbook	8.363	7.445	0.11
wikiwig	23.457	22.087	0.06
yabbse	22.475	20.458	0.09
zebraz	8.59	6.267	0.27
ztml	11.724	9.236	0.21
MMRE = 0.14, Pred(0.25) = 0.77			

Size Estimation for IS from their CDM Comparison with the FP Method

Table 7-4 The FP validation dataset result of using SVR for Open-Source
PHP-based systems

System	Actual Size (KLOC)	Estimated Size (KLOC)	MRE
ackerTodo	9.47	9.016	0.05
Bookmark4u	19.381	21.596	0.11
ByteMonsoon	4.219	4.681	0.11
Core	17.065	11.877	0.26
E107	65.649	54.343	0.17
eFiction	12.443	11.260	0.10
fdcl	3.856	4.237	0.10
galant	10.225	7.625	0.25
Infocentral	47.734	35.511	0.26
jamdb	4.554	4.993	0.10
openrealty	16.846	15.190	0.10
phpESP	33.645	34.847	0.04
Phpnews	6.532	8.448	0.29
phpollster	1.818	2.396	0.32
PhpScheduleIt	16.006	13.188	0.18
Phpsera	15.846	11.791	0.26
Phpwims	10.507	9.745	0.07
Phpwscookbook	7.607	6.996	0.08
plume	15.875	16.421	0.03
Quantum_Star_SE	57.643	47.893	0.17
Rasmp	9.452	9.958	0.05
Rimps	10.81	12.069	0.12
so-net	38.781	39.422	0.02
Supersurf	1.489	1.764	0.18
usebb	6.118	6.570	0.07
videodb	24.227	22.536	0.07
Webaddressbook	8.363	7.445	0.11
wikiwig	23.457	22.087	0.06
yabbse	22.475	20.458	0.09
zebraz	8.59	6.267	0.27
ztml	11.724	9.236	0.21
MMRE = 0.15, Pred(0.25) = 0.84			

Size Estimation for IS from their CDM Comparison with the FP Method**7.2.1.3 Comparison of proposed ANN method with the FP Method**

We trained the dataset used in model building in the same way as described in Chapter 6. Neural network parameters were tuned in training process. The value of epochs is 1000 and 35 neurons are used in hidden layer. The validation result for the testing dataset from FP is shown in Table 7-5. MMRE and Pred(0.25) computed are 0.17 and 0.84 respectively. As stated in Chapter 6, MMRE and Pred(0.25) computed for the same dataset are 0.18 and 0.81 respectively by using of the LOC estimation model built from the proposed method using the ANN. There is no clear difference between the results from the two methods.

7.2.2 Comparison Based on Java Open-Source Systems

For Java based language, the formula for converting FP to LOC estimation is available (Boehm, et al., 2000). The estimated LOC is Function Point (FP) multiplied by 53. Therefore, UFPs and VAFs were counted for only the validation dataset and are shown in Table 7-6.

With the use of the above formula to estimate LOC from FP, MMRE and Pred(0.25) computed are 0.17 and 0.75 respectively. Note that as stated in Chapter 4 for use of the LOC estimation model built from the proposed method using the multiple linear regression, MMRE and Pred(0.25) computed for the same dataset are 0.20 and 0.79 respectively. There is also no clear difference between the results from the two methods.

As stated in Chapter 5, MMRE and Pred(0.25) computed for the same dataset are 0.22 and 0.75 respectively by using of the LOC estimation model built from the proposed method using the Support Vector Machine. There is no clear difference between the results from the two methods.

Size Estimation for IS from their CDM Comparison with the FP Method

Table 7-5 The FP validation dataset result of using NN for Open-Source
PHP-based systems

System	Actual Size (KLOC)	Estimated Size (KLOC)	MRE
ackerTodo	9.47	7.387	0.22
Bookmark4u	19.381	17.765	0.08
ByteMonsoon	4.219	3.776	0.11
Core	17.065	16.126	0.06
E107	65.649	77.131	0.17
eFiction	12.443	7.997	0.36
fdcl	3.856	4.612	0.20
galant	10.225	8.124	0.21
Infocentral	47.734	61.481	0.29
jamdb	4.554	3.625	0.20
openrealty	16.846	19.794	0.17
phpESP	33.645	26.381	0.22
Phpnews	6.532	7.087	0.08
phpollster	1.818	1.138	0.37
PhpScheduleIt	16.006	14.897	0.07
Phpsera	15.846	18.762	0.18
Phpwims	10.507	12.283	0.17
Phpwscookbook	7.607	6.85	0.10
plume	15.875	18.367	0.16
Quantum_Star_SE	57.643	44.645	0.23
Rasmp	9.452	12.259	0.30
Rimps	10.81	10.086	0.07
so-net	38.781	31.859	0.18
Supersurf	1.489	1.946	0.31
usebb	6.118	5.533	0.10
videodb	24.227	26.698	0.10
Webaddressbook	8.363	6.696	0.20
wikiwig	23.457	21.426	0.09
yabbse	22.475	24.612	0.10
zebraz	8.59	9.879	0.15
ztml	11.724	13.549	0.16
MMRE = 0.17, Pred(0.25) = 0.84			

Size Estimation for IS from their CDM Comparison with the FP Method

Table 7-6 The FP data for the validation dataset for Open-Source Java-based systems

System	UFP	VAF	FP	Actual Size (KLOC)	Estimated Size (KLOC)	MRE
abaguibuilder	1602	1.02	1634	83.176	23.030	0.30
Art	426	1.02	435	17.67	12.055	0.15
Bofhms	223	1.02	227	10.507	18.754	0.11
Contineo	337	1.05	354	21.067	21.387	0.27
Dspace	923	1.03	951	47.57	50.387	0.06
Ejen	57	1.06	60	4.437	3.202	0.28
Imcms	1767	1.04	1838	91.424	97.397	0.07
Itracker	554	1.06	587	38.721	31.124	0.20
jdbforms	794	1.02	810	50.72	9.406	0.27
jmbase	298	1.04	310	19.723	67.521	0.07
jwma	1249	1.02	1274	63.228	86.416	0.17
Jbooks	174	1.02	177	12.817	20.035	0.29
Jfolder	388	1.04	404	29.288	86.604	0.04
Jgossip	367	1.03	378	28.047	16.426	0.17
Jpo	131	1.06	139	8.54	7.360	0.14
Jwnl	142	1.03	146	6.913	7.752	0.12
mp3cattle	277	1.05	291	14.598	42.924	0.15
openjms	854	1.05	897	59.72	15.415	0.06
Openhre	201	1.04	209	9.932	11.079	0.12
Quartz	291	1.04	303	22.954	47.525	0.20
sqlunit	1583	1.03	1630	74.089	16.040	0.30
tau_lastest	682	1.05	716	48.039	37.953	0.21
testsuite	482	1.03	496	30.076	26.312	0.13
Tesuji	122	1.02	124	8.01	6.595	0.18
MMRE = 0.17, Pred(0.25) = 0.75						

As stated in Chapter 6, MMRE and Pred(0.25) computed for the same dataset are 0.20 and 0.75 respectively by using of the LOC estimation model built from the proposed method using the Artificial Neural Network. There is no clear difference between the results from the two methods.

Size Estimation for IS from their CDM Comparison with the FP Method

7.3 Comparative Discussion

We have proposed and validated the use of proposed LOC estimation method using three standard estimating methods, Multiple Linear Regression (MLR), Support Vector Machine (SVM) and Artificial Neural Network (ANN), for estimating LOCs for information systems from their conceptual data models. As the proposed method is based on conceptual data model, clearly, at the early stage of software development cycle, the information required by the proposed methods are more readily available than the information required by the current well-known FP method. Moreover, our comparison on the accuracy also shows that the proposed methods are comparable with the FP method.

Since the proposed methods estimates LOC from conceptual data model, all the data required for the method is fully available at the end of requirements analysis. Furthermore, the derivation of the required data from conceptual model for the proposed methods is simple and does not subject to any further judgment, decision and interpretation.

Estimation of software size is a crucial activity in software management. The most commonly used size measures are Line of Code (LOC) and Function Point (FP). Due to the lack of information required by existing sizing methods, many practitioners and project managers continue to produce estimates based on ad-hoc or so called “expert” approaches (Armour, 2002; Miranda, 2000). Despite of its problems on the difficulty to derive the required information at the early stage of software development, Function Point has achieved a worldwide acceptance to estimate LOC of business systems and to indirectly predict the effort and cost of software project (Costagliola, et al., 2005). Some variants to the FP method have also been proposed. In addition to the

Size Estimation for IS from their CDM Comparison with the FP Method

above-mentioned major problem, although originally conceived to be independent of methodology used to develop the system under measurement, the application of the FP method turns out to be rather unnatural when applied to object-oriented (OO) systems (Costagliola, et al., 2005). As a consequence of using use-cases to model software requirements in OO approach, Use Case Point has been proposed as a method to estimated software effort (Smith, 1999). However, the information such as complexities of use-cases and details of scenarios required by this method are difficult to predict at the early stage of software development. Recently, the Class Point (CP) method that generalizes the FP method for OO systems has been proposed (Costagliola, et al., 2005). The CP method is based on information from design documentation. Clearly, much of the information required by the CP method is also available only when the design of the system completes.

Chapter 8 Application of the Proposed Estimation Models

8.1 Software Cost Estimation

From late of 1940s, software development and maintenance have become increasingly more complex and expensive. The rapid growth has also resulted to numerous software projects running into problems. Software that does not meet objectives and requirements is common. Project delay and cost overrun are very common in the industry. One of the main reasons of this is the inability to estimate software size and cost more accurately. Therefore, it is an important issue for project managers and software engineers to be able to predict the size of software and the cost of software development, so that the projects can meet user requirements, schedules, and budget.

Size Estimation for IS from their CDM Application of Proposed Models

8.2 Major Software Cost Estimation Models

Software cost estimation has been in research for more than 40 years. A large number of software cost estimation models have been proposed. In this section, we will give a summary of the software cost estimation models.

8.2.1 Models proposed before 1970s

As Mohanty pointed out in (Mohanty, 1981), the earliest efforts in estimating software cost arose from the standard industrial practice of measuring productivity for industrial workers. With the growth of the software development from 1950s, software cost estimation has started to be an important research topic. Next, we shall give a brief discussion on the major methods proposed in this period.

Farr and Zagorski Model (Farr and Zagorski, 1965) was proposed in 1965 at an ICC Symposium on Economics of Automatic Data Processing. This is the earliest software cost estimation model. This model includes thirteen cost drivers that include delivered statements, document types, size of database, and etc., which were derived by regression analysis of cost data to provide three formulas for estimating cost measured by man-month effort. The estimation is carried out based on a given complete operational specification.

Systems Development Corporation (SDC) Model (Nelson, 1966) uses a static, multivariable, linear model involving eleven predictors. The size is not explicitly used but the model estimates cost based on per thousands of lines of source code (KLOC).

Though the software cost estimation models proposed before 1970s were quite sophisticated; they were not accurate and were usually developed from a limited database, both in number of programs and variety of applications. Furthermore, according to renowned software economics researcher Dr. Barry

Size Estimation for IS from their CDM *Application of Proposed Models*

Boehm, these early models differed widely in their underlying assumptions and definitions of size. Hence, their cost equations varied substantially (Boehm, B. W., 1981).

8.2.2 Models Proposed in 1970s

The development of software cost estimation models was a very hot research topic in this decade. There are a large number of new cost estimation models were presented, such as *Wolverton Model* (Wolverton, 1974), *Doty model* (Herd, 1977), *Daly model* (Daly, 1977), *ESD model* (Mohanty, 1981), GRC model (Mohanty, 1981), and etc. Among these models, there are four important models that have been extended and refined in the later and still used at present, *Putnam Model* (Putnam, 1978), *RCA model* (Freiman and Park, 1979), Function Point Analysis (Albrecht, 1979) and *COCOMO* (Boehm, B. W., 1981). Next, we shall give a brief discussion on the major methods proposed in this period.

Putnam established and developed a complete analysis of the data collected from the software projects of the US Army Computer Systems Command, which resulted in a software estimation model, known as the Putnam Model (Putnam, 1978). This is a dynamic, multivariable model based on a Norden/Rayleigh function. There is a non-linear trade-off between the cost, the delivery time and the size. Putnam offers a complete methodology for which the size estimate is one of the starting points for the whole estimation process.

RCA model (Price-S) was the first software cost estimation models supported by the commercial tools (Freiman and Park, 1979). It is a proprietary estimating model developed and maintained by PRICE Systems Division of

Size Estimation for IS from their CDM *Application of Proposed Models*

RCA, New Jersey. The estimation is based on the historical cost data to predict and estimated cost.

FPA is firstly introduced by Albrecht in 1979, which can estimate both software size and software cost (Albrecht, 1979). In this method, a new metric Function Points (FP) is proposed to measure software size. This marked the beginning of a completely new way of measuring software size. Until today, it is still the most popular method that most of the software estimation methods are based on.

Constructive Cost Model (COCOMO) widely uses regression technique. This model is refined and calibrated in the next two decades and to be the most well known model in software cost estimation.

In this decade, identifying the major cost drivers attracts more interest of the researchers. In most of the models established at that time, statistical estimation and prediction techniques were used in analysis of project data.

8.2.3 Models Proposed in 1980s

In this decade, the research in software cost estimation field continued. SOFTCOST (Tausworthe, 1981), Estimacs (Rubin, 1983), COPMO (Conte, et al., 1986), SPQR/20 (Jones, 1986) were proposed. Furthermore, improving and extending the existing models become a active part of research. Next, we shall give a brief discussion on the major methods proposed in this period.

COCOMO was extended in 1981 with more sophisticated equations and were published in (Boehm, 1981). Jensen (1980) extended Putnam model by eliminating some of the undesirable behavior of SLIM

In this period, the extension of *FPA* research obtains a significant progress. Many FP variations appeared. Due to the rapid growth of Function Points, a

Size Estimation for IS from their CDM *Application of Proposed Models*

non-profit organization, International Function Point Users Group (IFPUG), was founded in order to develop standards for function point counting. Among the FP variations, Mark II Function Points (Symons, 1988) is the most important version. DeMarco function points was proposed in (DeMaro, 1982) which is called Bang Points. DeMarco function points measure the amount of functionality by counting certain components of a structured analysis specification model. Another FP variation is SPR Function Points (Jones, 1986). It is a superset of the IFPUG Function Point metric and introduces a new element in addition to the five standard Function Point parameters.

The research on software cost estimation was still active and most of the major models were consolidated in this decade. Main software cost estimation tools were developed and marketed for commercial use in this period.

8.2.4 Models Proposed in 1990s and Later

In last two decades, the main focus of software cost estimation research is on estimation techniques. Different techniques are adopted in software cost estimation. In addition, function points are continuing to grow in prominence. The IFPUG meets twice a year to discuss issues related to Function Points and to publish and update standards. Next, we shall give a brief discussion on the major methods proposed in this period.

Many new estimation techniques are proposed for software cost estimation, such as Case-based reasoning (Mukhopadhyay, et al. 1992), analogy (Shepperd, et al., 1996), Neural Network (Lee, et al., 1998), categorical regression (Angelis, et al., 2001) and regression tree (Briand and Wust, 2001). Furthermore, a number of promising tools such as ESTOR (Mukhopadhyay, et al. 1992) and ANGEL (Shepperd, et al., 1997) were developed.

Size Estimation for IS from their CDM *Application of Proposed Models*

Several FP variants were presented in last two decades. 3D Function Points (Desharnais and Abran, 1995) was introduced by Boeing Company. It was designed to address two classic problems associated with the Albrecht FPA based on analyzing data, function and control. Full Function Points (FFP) and COSMIC-FFP (COSMIC, 1999; Abran et al., 2003) extended the application of IFPUG FP method to real-time systems, embedded systems and complex mathematics systems.

Another notable event during this period is the appearance of COCOMO II. Barry Boehm made major enhancements to COCOMO and released the revision as COCOMO II (Boehm, et al., 2000). COCOMO II is a set of three models that are useful in different phases of the software development cycle.

Strike (Strike, et al., 2001) constructed a software cost estimation model with incomplete data. The estimation is based on least squares regression. Three techniques, listwise deletion, mean imputation and eight different types of hotdeck imputation, are used to do the simulation study when building the software cost estimation models.

Till now, the main focus in software estimation is moved to the estimation and prediction techniques. Many techniques are applied in software estimation model. The assessment of these techniques becomes a new research topic, and how to compare these techniques is also an interesting issue in software cost estimation field.

8.3 Application of the Proposed Estimation Models

The objective of the proposed model in this thesis is to estimate software size, which is a key factor in software cost estimation. Difficulty in estimating

Size Estimation for IS from their CDM *Application of Proposed Models*

software size is the main problem faced by all existing software sizing methods.

In this section, we will discuss the use of the proposed model in the most popular existing software cost estimation techniques. Clearly, the proposed method can be used in the commercial tools for providing more practical software size estimation.

8.3.1 Application in COCOMO II Model

COCOMO II is the most popular software cost estimation model. It has a set of three models: Applications Composition Model, Early Design Model and Post-Architecture Model.

In COCOMO II, software effort is measured by the commonly used term called person-months. A person-month is defined as the amount of time one person spends working on the software development project for one month. In COCOMO II, the number of person-hours per person-month is an adjustable factor with a nominal value of 152 hours. Software Size is a main factor in those three sub-models and can be measured in terms of KLOC, FP or Object Points.

Effort multiplier and scale factor are used in effort estimation. Cost drivers that have a multiplicative effect on predicting effort are called effort multipliers. Each effort multiplier has a rating level that expresses the impact of the multiplier on development effort. These ratings can range from extra low to extra high. Cost drivers are used to capture characteristics of the software development that affect the effort to complete the project. The specific value of the weight of the scale driver's rating level is called scale factor. The selection of scale drivers is based on the rationale that they are a significant source of exponential variation on a project's effort or

Size Estimation for IS from their CDM *Application of Proposed Models*

productivity variation. Each scale driver has a range of rating levels, from very low to extra high. A project's scale factors are summed across all of the factors.

In all the equations presented in this section, E is the software development effort which measured in man-months, S is the software size which measured in thousand lines of source code (KLOC), Function Point or Object Points. The values of the coefficients a and b depend on the COCOMO modeling system being used (application composition model, early design model, post-architecture model), and on the mode of the project modes are described (Organic, Semi-detached, Embedded).

The application composition model supports the estimation of the effort for the earliest phases or spiral cycles that generally involve prototyping and any other prototyping activities occurring later in the life cycle. The application composition model is as following:

$$E = a \times S^b \quad (8.1)$$

The early design model is used in the early stages of a software project when very little may be known about the size of the product to be developed, the nature of the target platform, the nature of the personnel to be involved in the project, or the detailed specifics of the process to be used. This model could be employed in either Application Generator, System Integration, or Infrastructure development sectors. The estimation of software effort based on the early design model is described as the following:

$$E = a \times S^b \times \prod_{i=1}^{15} EffortMultiplier_i \quad (8.2)$$

The Post-Architecture model is the most detailed estimation model and it is intended to be used when the software lifecycle architecture has been

Size Estimation for IS from their CDM *Application of Proposed Models*

developed. This model is used in the development and maintenance of software products in the Application Generators, System Integration, or Infrastructure sectors. The post-architecture model has the following form:

$$E = a \times S^b \times \prod_{i=1}^{17} EffortMultiplier_i \quad (8.3)$$

where $b = 1.01 + 0.01 \times \sum_{j=1}^5 ScaleFactor_j$.

Clearly, the proposed software size estimation model can be used to estimated the size in all the above software effort estimation models.

8.3.2 Application in SLIM Model

Software Life Cycle Model (SLIM) is a proprietary software cost-estimating tool offered by Quantitative Software Management, Incorporated, Virginia (Putnam, 1982; Putnam and Fitzsimmons, 1979). SLIM is usually revised annually as a result of continuous data-collection activities (Jones, 1998). This model is based on a Rayleigh distribution curve (shown in Figure 9-1) that is described as:

$$\frac{dy}{dt} = 2Kate^{-at^2} \quad (8.4)$$

Where dy/dt is the manpower in man-years per year, t is the elapsed time from the start of design to product replacement, K is the total software effort (equivalent to the area under the curve), and a is a constant that determines the shape of the curve.

Size Estimation for IS from their CDM *Application of Proposed Models*

Figure 8-1 Rayleigh distribution curve

According to Putnam, the Rayleigh distribution curve depicts the profile of a software development project, with time on the horizontal axis and manpower on the vertical axis.

Productivity is used to link the Rayleigh distribution curve to the software development characteristics of size and technology factors. That is,

$$P = \frac{S}{E} \quad (8.5)$$

where P is the productivity, S is software size and E is the development effort. Putnam assumes that the peak staffing level in the Rayleigh distribution curve corresponds to development time (t_d). And E was found to be approximately 40% of K . Putnam found that when productivity was high, the initial staff buildup was lower than for projects with lower productivity. Putnam associated initial staff buildup with the difficulty, D , of the project, the slope of the curve at $t=0$ is defined as:

$$D = \frac{K}{t_d^2} \quad (8.6)$$

And the relationship between difficulty D and the productivity P is

$$P = \alpha D^{-\frac{2}{3}} \quad (8.7)$$

Size Estimation for IS from their CDM *Application of Proposed Models*

where α is a proportionality constant that depends on the state of technology existing in the environment where the data was captured. As E was found to be approximately 40% of K , the effort equation can be conducted as:

$$E = 0.4 \times \left(\frac{S}{C} \right)^3 \times \frac{1}{t_d^4} \quad (8.8)$$

where C is a technology constant that accounts for differences among projects. It is equal to 0.4α which is suggested by Putnam. In equation 8.8, software size is a input to estimate software cost, thus, our proposed models can be easily used to do the software cost estimation.

8.3.3 Application in SEER-SEM Model

SEER-SEM is the System Evaluation and Estimation of Resources – Software Estimation Model. It is a commercial tool based on the original Jensen model. It uses either LOC or FP in addition to personal attributes, tools, complexity and constraints as input to estimate cost, schedule, risk and maintenance of a software project. The effort can be calculated as the following equation:

$$E = D \times 0.4 \times \left(\frac{S}{Cte} \right)^{1.2} \quad (8.9)$$

where E is the effort, D is staffing complexity that is a rating of the project's inherent difficulty in terms of the rate at which staff are added to a project, S is the software size, Cte is effective technology, a composite metric that captures factors relating to the efficiency or productivity with which development can be carried out. In SEER-SEM, software size can be measured by LOC, thus our proposed model can be easily used to do the software cost estimation.

SEER-SEM handles a variety of environmental and application configurations, such as client-server, embedded, distributed, and stand-alone

Size Estimation for IS from their CDM *Application of Proposed Models*

systems, varied MIS applications, graphics, signal processing environments and etc. It models the most widely used development methods and languages.

8.3.4 Application in Commercial Software Cost Estimating Tools

Software cost estimating tools increasingly play an important role in the project management area. Most modern software cost estimating tools support both function point and lines of code (LOC) software size measurements. Thus, the proposed cost estimation method can be used to estimate software size in LOC as an input to these tools. In this section, we introduce some main software cost estimating tools.

The PRICE-S model consists of three sub-models that support costs and schedule estimation for the development and support of computer systems. The software size used in PRICE can be SLOC, FP and the new Predictive Object Points. PRICE-S is still a popular estimating tool at present and is marketed by PRICE Systems (<http://www.pricesystem.com>).

SOFTCOST tool is based on the Deep Space Network (DSN) estimation model (Tausworthe, 1981). Tausworthe extended the work of Boehm, Herd, Putnam, Walston-Felix and Wolverton to develop a cost estimation model. This model is similar to Doty and COCOMO in having a nominal scaling equation and a set of multiplicative effort adjustment factors. It uses a linear scaling equation. It also uses the Rayleigh distribution curve to determine the phase distribution of effort. Donald Reifer further extended this model to produce the commercial PC-based SOFTCOST-R model (<http://www.reifer.com>).

ESTIMACS tool is based on the model developed by Dr. Howard Rubin (Rubin, 1983). ESTIMACS is today marketed as CA-Estimacs. It consists of 9 modules. Among these modules, the effort module is the most important and

Size Estimation for IS from their CDM *Application of Proposed Models*

extensive one. In order to produce an estimation, user must answer 25 input questions related to the user organization and to the complexity and size of the software to be developed.

Before You Leap (BYL) is a commercial package based on a link-up between FPA and COCOMO. This tool is developed by Gordon Group (Heemstra, 1992). It estimates the software cost by using LOC as the input to the COCOMO part of BYL. Thus, the amount of FP should be translated into LOC. As our software sizing models predict software size in LOC, it is more convenient for user to use BYL tool to estimate software cost.

SPQR/20, developed by Software Productivity Research Inc., has a simple set of multiple choice questions that address the system characteristics such as project type, project scope, goals, application type, program requirements, user documentation and etc.. It is the first commercial tool explicitly built to support function point metrics (Jones, 1986).

CHECKPOINT is a knowledge-based software project estimating tool from Software Productivity Research (SPR). It is developed from Capers Jones' studies (Jones, 1997). It has a proprietary database of about 8000 software projects and focuses on four areas that need to be managed to improve software quality and productivity. Checkpoint uses function points (or feature points) as its primary input of size. It predicts effort at four levels of granularity: project, phase, activity and task.

8.4 Summary

In this chapter, we give an overview of the software cost estimation models in different decades. We also discussed the use of the proposed software size estimation methods using three most popular software cost estimation models.

Size Estimation for IS from their CDM *Application of Proposed Models*

There is a large number of commercial software cost estimating tools. However, because most of the commercial software cost estimating tool vendors regard their estimating methods and algorithms as trade secrets, there are only several software cost estimation models explain the mechanics and algorithms in details. However, the proposed software size estimation method can also be used to estimate software size as an input to these tools.

Chapter 9 Conclusion

This thesis investigates the issue of estimating software size at the early stage of the software lifecycle. A novel conceptual data model is proposed for software size estimation. Based on the novel conceptual data model, Multiple Linear Regression, Support Vector Machine and Artificial Neural Network approaches are used to build software size estimation models. Five pairs of datasets are collected to build and evaluate the estimation models for lines of source code from the conceptual data model characteristics. The comparison with well-known software estimation method Function Point is also carried out. Both the evaluation and comparison shows that the proposed software size estimation model is good and promising.

9.1 Summary of the proposed model

Software estimation is an important issue in software engineering. Software size estimation is a key in the software estimation. However, existing methods still have many problems due to the need of details that are not available at the early stage of software development.

In this thesis, we proposed a new software size estimation model based on the conceptual data model that is readily available at the early stage of software development. Conceptual data model is usually constructed at the early stage of requirement analysis for information systems. In the proposed size estimation model, we use the total number of classes in the conceptual data model (C), the total number of relationships (R) and the average number of attributes per entity ($\bar{A}$) in the conceptual data model to characterize the size of software.

Multiple linear regression, support vector machine and artificial neural network are three common approaches in model building. These techniques are successfully applied in many areas from financial problems to signal process, software size estimation and cost estimation. In this project, these three methods are applied respectively in building the models we proposed in order to validate the model proposed. The datasets we used in the experiments are collected from both industry and Open-Source systems. Several criteria have been applied in the model building and evaluation. All the evaluations give support to the models built. This shows that the proposed model is promising and encouraging.

Existing software sizing methods including Function Point method have the problems in the need of using design and implementation information that are not available at the early stage of software development. Function Point (FP) method is a well know software sizing method which proposed by Allan Albrecht. Comparison between FP method and the models built by three methods was also carried out. The result shows there is no significant difference between these models and the Function Point method in software size estimation. This gives further support that the proposed methods can be used to address the current problems in software sizing.

9.2 Limitation of Research Work

In this research, three techniques, Multiple Linear Regression, Support Vector Machine and Artificial Neural Network, are applied in model building. However, there are some limitations on using these techniques in doing this research.

Firstly, all these three methods need true historical project data and these data should be in the same scope of the software system. However, it is not possible for our samples to cover the full range of possible LOC values well. Thus, the models that we have built are not appropriate for estimating systems with LOC values that are significantly larger or smaller than the LOC values in the respective samples.

Multiple Linear Regression method can give a regression equation that characterizes the relationships among the variables. However, this method can only analyze the linear relationships among the variables. In the future research, if the variables added do not have linear relationships, this method is not useful.

Support Vector Machine can analyze the variables that have linear or nonlinear relationships. However the training and testing data requires much time. Compared to MLR, it is difficult to do a quick estimation by using SVM.

Artificial Neural Network cannot provide a clear explanation on the conducted model. However this method can do more accurate estimation if the history data are sufficient. And this method is not affected by the relationships between the variables. Using this method need more time to training a network. If the user doesn't care about the time, it is a good choice to use this method.

9.3 Future Work and Recommendations

As presented in chapter 4, collect data is extremely difficult for the security problems and other conditions. However, sufficient dataset is an important issue in experiments. Based on the promising software size model that we proposed, it is the time to carry out much larger scale of experimentation so that larger and more refined datasets can be collected to explore on building software size models for practical use. This would require much larger scale of industry collaboration and support internationally.

Software size is a key factor in most of the software effort and cost estimation. It is highly probable to estimate project effort and cost directly from the three independent variables used in the proposed method in a similar way as the FP method. This is an important further research area. Furthermore, based on the sizing model we proposed in this thesis, more characteristics of the system development and environment may be explored on model construction of software cost estimation in the future.

Currently, it is still a question whether multi-organizational data are more valuable than company-specific data in software estimation. Although many researchers (Wieczorek and Ruhe, 2002; Briand, et al., 1999) have done some research on this issue, it still cannot get conclusive result. The data used in this project are collected from many companies in Singapore as well as the Open-Source systems. Thus, exploring on this comparison between models built on the multi-organizational data and company-specific data is a valuable further research.

Author's Publications

- 1) Tan. H. B. K., and Zhao, Y., "Estimating LOC for information systems from their conceptual data models," Forthcoming in proceeding of 28th Int. Conf. on Software Engineering, 2006, (the top conference on software engineering, 9% acceptance rate).
- 2) Tan. H. B. K., and Zhao, Y., "Estimating Software Size for Data-Intensive Systems," to appear in Trans. IEICE on Information & Systems, Vol. E89-D, No. 4, April, 2006.
- 3) Tan. H. B. K., and Zhao, Y., "Recovery of Data Dependencies," Encyclopedia of Data Warehousing and Mining, 2005, pp. 947-949.
- 4) Tan. H. B. K., and Zhao, Y., "Building empirical-based knowledge for design recovery," Encyclopedia of Data Warehousing and Mining, 2005, pp. 112-117.
- 5) Zhao, Y., and Tan, H. B. K., "Sizing data-intensive systems," in EEE Research, 2005, pp 60-61.

- 6) Tan. H. B. K., and Zhao, Y., "ER-Based software sizing for data-intensive systems," in Proc. 23rd Int. Conf. on Conceptual Modeling, 2004, pp.180-190, (19% acceptance rate).
- 7) Tan. H. B. K., and Zhao, Y., "Automated elicitation of functional dependencies from source codes of database transactions," in Information and Software Technology, Vol. 46, 2004, pp 109-117.
- 8) Zhao, Y., and Tan, H. B. K., "Software cost estimation through conceptual requirement," in Proc. Int. Conf. Quality Software, 2003, pp.141-144.
- 9) Tan. H. B. K., and Zhao, Y., "Automated elicitation of inclusion dependencies from the source code for database transactions," in Journal of Software Maintenance and Evolution: Research and Practice, Vol. 15, 2003, pp 379-392.

Bibliography

- Abran, A., Deshairnes, J.M., Oligny, S., St-Pierre, S., Symons,C.,
Measurement Manual COSMIC Full Function Points 2.2, The COSMIC
Implementation Guide for ISO/IEC 19761, 2003.
- Abran, A., Robillard, P. N., Function Points Analysis: An Empirical Study of
Its Measurement Processes, IEEE Transactions on Software Engineering,
vol.22, no. 12, 1996.
- Albrecht, A. J., and Gaffney, J. E., Software Function, Source Lines of Code,
and Development Effort Prediction: A Software Science Validation, in
IEEE Trans. Software Eng., vol. SE-9, no. 6, pp. 639-648, 1983.
- Albrecht, A, J, Measuring Application Development Productivity, In Proc.
IBM Applications Development Symp., Monterey, Calif., pp.
14-17,1979.
- Angelis, L., Stamelos, I., Morisio, M., Building A Software Cost Estimation
Model Based on Categorical Data, In Proc. of the Seventh Inte. Software
Metrics Symposium (METRICS '01), pp. 1530-1435, 2001.

- Armour, P., Ten Unmyths of Project Estimation: Reconsidering Some Commonly Accepted Project Management Practices, In Comm. ACM 45(11), pp. 15-18, 2002.
- Arnold, M. and Pedross, P., Software Size Measurement and Productivity Rating in a Large-Scale Software Development Department, in Proc. Int. Conference on Software Eng, pp. 490-493, 1998.
- Bell, D., UML Basics Part III: The Class Diagram, The Rational Edge, 2003.
- Belsley, D. A., Kuh, E., and Welsch, R. E., Regression Diagnostics: Identifying Influential Data and Sources of Collinearity, John Wiley, New York, 2004.
- Blaha, M., and Premerlani, W., Object-Oriented Modeling and Design for Database Applications, Prentice Hall, 1998.
- B. W. Boehm, and R. E. Fairley, Software Estimation Perspectives, In IEEE Software, pp. 22-26, 2000,.
- Boehm, B. W., Software Engineering Economics, IEEE Transactions on Software Engineering, vol. 10, no. 1, pp. 4-21, 1984.
- Boehm, B. W., Software Engineering Economics, Englewood Cliffs, NJ, Prentics-Hall, 1981.
- Boehm, B. W., Software Engineering Economics, Tutorial, Software Management: Third Edition, Washington, DC, IEEE Computer Society Press, 1986.
- Boehm, B. W., et al., Software Cost Estimation with COCOMO II, Prentice Hall, 2000.

- Booch, G., Rumbaugh, J., Jacobson, I., The Unified Modeling Language User Guide, second edition, Addison Wesley, 2005.
- Boser, B., Guyon, I., and Vapnik, V., A training Algorithm for Optimal Margin Classifiers, In Proc. of the 5th Annual ACM Workshop on Computational Learning Theory, pp. 144-152, ACM Press, 1992.
- Bozoki, G. J., SSM User's Guide, Redwood City, CA, GJB Associates, 1984.
- Briand, L. C. and Wust, J., Modeling Development Effort in Object-Oriented Systems Using Design Properties, IEEE Trans. On Software Eng., vol. 27, no. 11, 2001.
- Briand, L. C., Emam, K. E. and Maxwell, K. D., An Assessment and Comparison of Common Software Cost Estimation Modeling Techniques, in Proc. of Inter. Conf. of Software Engineering, pp. 313-322, 1999.
- Briand, L. C., Langley, T. and Wiecezorek, I., A Replicated Assessment and Comparison of Common Software Cost Modeling Techniques, in Proc. of Inter. Conf. of Software Engineering, pp. 377-386, 2000.
- British standards: Software engineering - IFPUG 4.1 Unadjusted Functional Size Measurement Method - Counting Practices Manual, British Standards Institution, 2003.
- Burgress, R. S., Structured Program Design Using JSP, ELBS, 1988.
- Canfora, G., Cerulo, L., and Troiano, L., An Experience of Fuzzy Linear Regression Applied to Effort Estimation, in the 16th International Conference on Software Engineering & Knowledge Engineering, pp. 57-61, 2004 .

- Catley, C., Frize, M., Walker, R.C., Petriu, D.C. , Predicting High-Risk Preterm Birth Using Artificial Neural Networks, IEEE Transactions on Information Technology in Biomedicine, vol. 10, Issue 3, pp. 540-549 July 2006.
- Chang, C. C. and Lin, C. J., LIBSVM: a library for support vector machines, 2001. Software available at <http://www.csie.net.edu.tw/~cjlin/libsvm>.
- Chen, S., et al., Support Vector Machine Multiuser Receiver for DS-CDMA Signals in Multipath Channels, IEEE Transactions on Neural Networks, vol. 12, no. 3, pp.604-661, 2001 .
- Chen, P. P., The Entity-relationship Model - towards A Unified View of Data, in ACM Trans. Database Syst. Vol. 1, no. 1, pp. 9-36, 1976.
- Conte, S.D., H.E.Dunsmore, and V.Y.Shen, Software Engineering Metrics and Models, Benjamin/Cummings,1986.
- Cortes, C., and Vapnik, V., Support Vector Networks, Machine Learning, vol. 20, pp. 273-297, 1995.
- COSMIC, Full Functions, Release 2.0, September 1999.
- Costagliola, G., Ferrucci, F., Tortora, G., and Vitiello, G., Class Point: An Approach for the Size Estimation of Object-oriented Systems, in IEEE Trans. Software Eng., vol. 31, no. 1, pp. 52-74, Jan, 2005.
- Cristianini, N., and Taylor, S. J., An Introduction to Support Vector Machines, Cambridge University Press, Cambridge, U.K., 2000
- Daly, E. B. , Management of Software Development, IEEE Transactions on Software Engineering, pp. 229-242, 1977.
- Data modeling: <http://www.utexas.edu/its/windows/database/datamodeling/>

- DeMarco, T., Controlling Software Projects: Management, Measurement & Estimation, foreword by Barry W. Boehm, Yourdon Press., New York, 1982.
- Desharnais, J. M., Abran, A., 3D Function Point Rules - Project Overview and Issues , in IFPUG Fall Conference , Toronto , International Function Point Users Group, 1995.
- Dolado, J. J., A study of the Relationships among Albrecht and Mark II Function Points, Lines of Code, 4GL and Effort, J. Systems and Software, vol. 37, no. 2, pp. 161-173, 1997.
- Dolado, J. J., Limits to the Methods in Software Cost Estimation, Limerick University Press, pp 63-68, 1999.
- Dutta, S., and Shekhar, S., Bond rating: A Non-conservative Application, In Proc. of the Inter. Joint Conf. on Neural Networks, pp. 443-450, 1988.
- El-Sheimy, N., Chiang, K.-W., and Nouredin, A., The Utilization of Artificial Neural Networks for Multisensor System Integration in Navigation and Positioning Instruments, IEEE Transactions on Instrumentation and Measurement, Vol. 55, Issue 5, pp. 1606-1615, Oct. 2006.
- Fairley, R. E., Recent Advances in Software Estimation Techniques, In proc. of 14th Inter. Conf. on Software Eng., pp. 382-391, 1992.
- Farr, L. and Zagorski, H. J., Quantitative Analysis of Programming Programming Cost Factors: a Progress Report, In Economics of Automatic Data Processing, ICC Symposium Proceedings, Rome, Frielink, A. B. (ed.), North-Holland, 1965.
- Fausett, L., Fundamentals of Neural Networks, New Jersey: Prentice Hall, 1994.

- Fenton, N. E., Shari, L. P., Software Metrics: A Rigorous and Practical Approach, International Thomson Computer Press, London, 1996.
- Ferens, D. V., Software Size Estimation Techniques, In proc. of the IEEE NAECON, pp. 701-705, 1988.
- Foss, T., et al., A Simulation Study of the Model Evaluation Criterion MMRE, IEEE Transactions on Software Engineering, vol. 29, no. 11, pp. 985-995, 2003.
- Freiman, F. R. and Park, R. E., PRICE Software Model - Version 3: An Overview, In proc. of IEEE-PINY, Workshop on Quantitative Software Models, IEEE Catalog No. TH0067-9, pp. 32-41, 1979.
- Freshmeat, <http://freshmeat.net>.
- Funahashi, K., On the Approximate Realization of Continuous Mappings by Neural Networks, Neural Networks, pp. 183-192, 1989.
- Garmus, D., and Herron, D., Function Point Analysis: Measurement Practices for Successful Software Projects, Addison Wesley, 2000.
- Ghezzi, C., Jazayeri, M. and Mandrioli, D. Fundamentals of Software Engineering. 2nd Edition, Prentice Hall, 2003.
- Glantz, S. A., and Slinker, B. K. Primer of Applied Regression & Analysis of Variance. 2nd Edition, McGraw-Hill, 2001.
- Gray, A., and MacDonell, S., A Comparison of Techniques for Developing Predictive Models of Software Metrics, Information and Software Technology, 1997.
- Heemstra, F.J., Software Cost Estimation, Information and Software Technology, Vol. 34, No. 10, pp 627-639, 1992.

- Herd, J. R., et al., Software Cost Estimation Study - Study Results, Final Technical Report, RADC-TR-77-220, vol. 1 of 2, Doty Associates Inc., Rockville, MD, 1977.
- Hoptroff, A. R., The Principles and Practice of Time Series Forecasting and Business Modelling Using Neural Nets, Neural Computing and Applications, vol. 1, no. 1, pp. 59-66, 1993.
- Huang, X. S., Capretz, L.F., Jing, R., Ho, D., A Neuro-Fuzzy Model for Software Cost Estimation, In Proc. of Int. Conference on Quality Software, pp. 126 – 133, 2003.
- Hughes, R.T., Expert Judgement as an Estimating Method, Information and Software Technology, Vol. 38, No. 2, pp. 67-75, 1996.
- Idri, A., Khoshgoftaar, T. M. and Abran, A., Can Neural Networks Be Easily Interpreted in Software Cost Estimation, FUZZ-IEEE, Hawaii, pp. 1162-1166, 2002.
- Ikatura, M., and Takayanagi, A., A Model for Estimating Program Size and Its Evaluation, In Proc of sixth Inter. Conf. on Software Engineering, pp. 104-109, 1982.
- Jeffery, D. R., and Walkerden, F., An Empirical Study of Analogy-based Software Effort Estimation, in Empirical Software Engineering, Kluwer Academic Publishers, vol. 4, no 2, pp 135-158, 1999.
- Jensen, R.W. A Macro-level Software Development Cost Estimation Methodology, Conf. Record of the Fourteenth Asilomar Conf. on Circuits Systems & Computers, 1980.
- Jones, C., Programming Productivity, McGraw-Hill, 1986.
- Jones, C., Applied Software Measurement, McGraw Hill, 1997.

Jones, T. C, Estimating Software Costs, McGraw-Hill, 1998.

Jones, C., Software Assessments, Benchmarks, and Best Practices, Addison-Wesley, 2000.

Jorgensen, M., Experience with Accuracy of Software Maintenance Task Effort Prediction Models, IEEE Transaction on Software Engineering, Vol. 21(8), pp. 674-681, 1995.

Keerthi, S. S. and Lin, C. J., Asymptotic Behaviors of Support Vector Machines with Gaussian Kernel, Neural Computation, vol. 15, no. 7, pp. 1667-1689, 2003.

Kennedy, P., A Guide to Econometrics, Blackwell Publishing, 5th Edition, 2003.

KOLODNER, J.L. (1993) Case-Based Reasoning. Morgan Kaufmann.

Kuh, A., Adaptive Kernel Methods for CDMA Systems, IJCNN'01, Washington D. C., pp 1404-1409, 2001.

Lai, R., and Huang, S. J., A Model for Estimating the Size of a Formal Communication Protocol Application and Its Implementation, in IEEE Trans. Software Eng., vol. 29, no. 1, pp. 46-62, Jan, 2003.

Lambert, J. M., A Software Sizing Model, ISPA Journal of Parametrics, vol. 5, no. 4, pp. 75-87, 1986.

Lee, A. Cheng C. H., Balakrishnan, J., Software Development Cost Estimation: Integrating Neural Network with Cluster Analysis, Information & Management, vol. 34, no. 1, pp. 1-9, 1998.

- Li, Y., Gong, S., and Liddell, H, Support Vector Regression and Classification Based Multiview Face Detection and Recognition, in Proc. of the IEEE Int. Conf. on Automatic Face and Gesture Recognition, 2000.
- Lin, H. T. and Lin, C. J., A Study on Sigmoid Kernels for SVM and the Training of Non-PSD Kernels by SMO-type Methods , technical report, Department of Computer Science and Information Engineering, National Taiwan University.
- Lo, B. W. N. and Gao, X., Assessing Software Cost Estimation Models: Criteria for Accuracy, Consistency and Regression, Australian Journal of Information Systems, vol. 5, no. 1, pp. 30-44, 1997.
- Loney, K., Oracle DBA Handbook, 1994.
- Mair, C., Kadoda, G., Lefley, M., Phalp, K., Schofield, C., Shepperd, M., and Webster, S., An Investigaion of Machine Learning Based Prediction Systems, Bournemouth University, 1999, <http://dec.bournemouth.ac.uk/ESERG/ANGEL/Mair99.pdf>.
- McClave, J. T., and Sincich, T., Statistics, 9th Ed, Prentice Hall, 2003.
- McCulloch, W., and W. Pitts, A Logical Calculus of the Ideas Imminent in Nervous Activity, Bulletin of Mathematical Biophysics, pp. 115-33, 1943.
- Miranda, E., An evaluation of the paired comparisons method for software sizing, in Proc. Int. Conf. on Software Eng., 2000, pp. 597-604.
- Mohagheghi, P., Anda, B. and Conradi, R., Effort Estimation of Use Cases for Incremental Large-Scale Software Development, In Proc. of Int. Conference on Software Eng, pp. 303-311, 2005.

- Mohanty, S.N. Software cost estimation: present and future, *Software Practice and Experience*, vol. 11, no. 2, pp 103-121, 1981.
- Molokken, K., and Jorgensen, M. A review of surveys on software effort estimation. In *Proc. Int. Symposium on Empirical Software Eng.*, 2003, 223-230.
- Mukhopadhyay, T., Vicinanza, S.S., and Prietula, M.J. Examining the feasibility of a case-based reasoning model for software effort estimation, *Management Information Systems Quarterly*, vol. 16, no. 2, pp. 155-171, 1992.
- Muller, K., Mika, S., G. Ratsch, K. Tsuda, and B. Scholkopf, An Introduction to Kernel-Based Learning Algorithms, *IEEE Trans. on Neural Networks*, Vol. 12, #2, 181-202, 2001.
- Najberg, Andrew C., *Software Data Base Development, Volume I*, Reading, MA, the Analytic Sciences Corporation (Technical Report 4612-S-1): 1984, pp. 2-4.
- Nelson, E. A., *Management handbook for the estimation of computer programming costs*, AD-648-750, Systems Development Corporations, 1966.
- Neter, J., Kutner, M. H., Nachtsheim, C. J., and Wasserman, W., *Applied Linear Regression Models*, IRWIN, 1996.
- Nigrin, A., *Neural Networks for Pattern Recognition*, Cambridge, MA: The MIT Press, 1993.
- Osuna, E., Freund, R., and Girosi, F., Training Support Vector Machines: An application to Face Detection: 3, in *Proc. Of CVPR'97*, Puerto Rico, pp. 130-136, 1997.

- Putnam, L. H., Fitzsimmons, A., Estimating Software Costs, Datamation, pp. 189-198, 1979.
- Putnam, L. H., Progress in Modelling the Software Life Cycle in a Phenomenological Way to Obtain Engineering Quality Estimates and Dynamic Control of the Process, IEEE computer society publication, NO. 78CH1390-4C, 1978.
- Putnam, L. H., The Real Economics of Software Development, in The Economics of Information Processing, Vol. 2, Operations, Programming and Software Models, John Wiley & Sons, pp.167-176, 1982.
- Ranganathan, H., Gunasekaran, N., Simple Method for Estimation of Hemoglobin in Human Blood Using Color Analysis, IEEE Transactions on Information Technology in Biomedicine, Vol. 10, Issue 4, pp. 657-662, Oct. 2006.
- Reese, R. M., and Tamulevicz, J., A Survey of Software Sizing Methodologies and Tools, ISPA Journal of Parametrics, vol. 7, no. 2, pp. 36-55, 1987.
- Refenes, A. N., Currency Exchange Rate Prediction and Neural Network Design Strategies, Neural Computing and Applications, vol. 1, no. 1, 1993.
- Rubin, H.A., Macro-estimation of Software Eevelopment Parameters: the ESTIMACS System, SOFTFAIR. In Proc. Conf on Software Development Tools Techniques and Alternatives, IEEE Comput. Soc. Press, Silver Spring MD USA, 1983.
- Ruthe, M., Jeffery, R. and Wieczorek, I., Cost Estimation for Web Applications, in Proc. Int. Conf. On Software Eng., pp. 285-294, 2003.

- SAS/STAT User's Guide,
<http://www.id.unizh.ch/software/unix/statmath/sas/sasdoc/stat/>.
- Scholkopf, B., Burges, C., and Smola, A., *Advances in Kernel Methods Support Vector Learning*, MIT Press, Cambridge, MA, 1999.
- Shepperd, M., Schofield, C., Kitchenham, B. A., *Effort Estimation Using Analogy*, In Proc. of the 18th Inter. Conf. on Software Eng., pp. 170-175, 1996.
- Shepperd, M., Cartwright, M., and Kadoda G., *On Building Prediction Systems for Software Engineers*, *Empirical Software Engineering*, 5(3), pp. 175-182, 2000.
- Singhal, M., *Software Size Estimator*, *ISPA Journal of Parametrics*, vol. 6, no. 4, pp. 38-50, 1986.
- Smith, J., *The Estimation of Effort Based on Use Cases*, Rational Software White Paper, 1999.
- Smola, A. J. and B. Scholkopf, *A Tutorial on Support Vector Regression*, *Statistics and Computing*, 14(3), pp. 199-222.
- SourceForge.net, <http://sourceforge.net/>.
- Stensrud, E., Foss, T., Kitchenham, B., Myrtveit, I., *An Empirical Validation of the Relationship Between the Magnitude of Relative Error and Project Size*, in Proc. IEEE Symp. Software Metrics, pp. 3-12, 2002.
- Strike, K., Emam, K. E., and Madhavji, N., *Software Cost Estimation with Incomplete Data*, *IEEE Transactions on Software Engineering*, Vol. 27, No. 10, pp. 890-908, 2001.

- Su, Y. S.; Huang, C. Y., Chen, Y. S., Chen, J. X., An Artificial Neural-Network-Based Approach to Software Reliability Assessment, TENCON 2005, 2005 IEEE Region 10, pp.1-6, Nov. 2005.
- Symons, C.R. Function Point Analysis: Difficulties and Improvements, IEEE Trans. on Software Eng., vol. 14, no. 1, pp. 2-11, 1988.
- Tadayon, N., Neural Network Approach for Software Cost Estimation, in Proc. of the International Conference on Information Technology: Coding and Computing (ITCC'05), Vol. 2, pp. 815-818 April 2005.
- Tan, H. B. K., and Zhao, Y., ER-based Software Sizing for Data-intensive Systems, in Proc. Int. Conf. on Conceptual Modeling, pp. 180-190, 2004.
- Tan, H. B. K., and Zhao, Y., Estimating LOC for Information Systems from their Conceptual Data Models, Forthcoming in proceeding of 28th Int. Conf. on Software Engineering, 2006.
- Tang, Z., and Fishwick, P. A., Feedforward Neural Nets as Models for Time Series Forecasting, ORSA Journal on Computing, vol. 5, no. 4, pp. 374-385, 1993.
- Tausworthe, R.C., Deep Space Network Estimation Model, Jet Propulsion Report 817, 1981.
- Teorey, T. J., Yang, D., and Fry, J. P., A Logical Design Methodology for Relational Databases Using the Extended Entity-relationship model, in ACM Computing Surveys, Vol. 18, No. 2, pp. 197-222, 1986.
- Teory, T. J., Database Modeling & Design: The Basic Principles, 2nd ed., Morgan Kaufmann Publishers, Inc, 1994.

- Vapnik, V., Estimation of Dependencies Based on Empirical Data, Springer, New York, 1982.
- Vapnik, V., The Nature of Statistical Learning Theory, Springer, New York 1995.
- Wang, Q., Yu, B., Zhu, J., Extract Rules From Software Quality Prediction Model Based on Neural Network, in Proc. of the 16th IEEE International Conference on Tools with Artificial Intelligence, PP. 191-195, Nov. 2004.
- Wang, X., Zhong, Y., Statistical Learning Theory and State of the Art in SVM, In Proc. of the second IEEE Inter. Conf. on Cognitive Informatics (ICCI'03), pp. 55-59, 2003.
- Wieczorek, I. and Ruhe, M., How Valuable Is Company-specific Data Compared to Multi-company Data for Software Cost Estimation?, In Proc. of the eighth IEEE Symposium on Software Metrics (METRICS'02), pp. 237-246, 2002.
- Wittig G. E. and Finnie G. R., Using Artificial Neural Networks and Function Points to Estimate 4GL Software Development Effort, Australian Journal of Information Systems, 1994.
- Wittig, G. E., Finnie, G. R., Estimating Software Development Effort with Connectionist Models, Information and Software Technology, vol. 39, pp. 469-476, 1997.
- Wolverton, R. W., The Cost of Developing Large-scale Software, IEEE Transactions on Computers, C-23, pp. 615-636, 1974.

commendar is a web application to generate a community calendar web site, which publicizes community events and allows users to keep track of their own, personal appointments. This system provides effective, automated reminders of important events via e-mail.

Commerce : <http://sourceforge.net/projects/commerce>

Commerce project is an open source e-commerce web-based system. It is written in PHP and MySQL.

Coppermine : <http://sourceforge.net/projects/coppermine>

Coppermine is an easily set-up, fast, feature-rich photo gallery script with MySQL database, user management, private galleries, automatic thumbnail creation, ecard feature and a template system for easy customization to match the rest of a site.

EclipseBB : <http://sourceforge.net/projects/eclipsebb>

EclipseBB is a pre-modded phpBB forum. It is usable on any webserver running PHP and MySQL. It is based on the original phpBB forum software offering many modifications to the core of its parent software.

Jdcms : <http://sourceforge.net/projects/jdcms>

Jdcms is a free opensource PHP/MySQL content management system. It is no longer being actively developed.

Linkbase: <http://sourceforge.net/projects/linkbase>

Linkbase is a web application (written in PHP) that helps to manage the Internet bookmarks of an organisation (such as a company, a school...) in a database. There is one link collection for the public and each user can have his private links.

Linpha : <http://sourceforge.net/projects/linpha>

Linpha is a system that includes multilingual, flexible photo/image archive/album/gallery. It is written in PHP. It uses a SQL database to store information about your pictures.

Lotgd : <http://sourceforge.net/projects/lotgd>

Legend of the Green Dragon (Lotgd) is a remake of the old BBS game Legend of the Red Dragon (aka LoRD).

Mailwatch : <http://sourceforge.net/projects/mailwatch>

Mailwatch for MailScanner is a web based front-end to MailScanner written in PHP which uses MySQL to log information about each message processed by MailScanner and JpGraph to produce graphed reports.

Mantis : <http://sourceforge.net/projects/mantisbt>

Mantis is an easily deployable, web based bugtracker to aid product bug tracking. It requires PHP, MySQL and a web server. It is simpler than Bugzilla and easily editable.

Mundimail : <http://sourceforge.net/projects/mundimail>

Mundimail is a web-based newsletter manager utilizing mysql and php. Whereas most mysql/php mailers are designed for 1,000 subscribers, Mundimail was built to handle 100,000 subscribers or more.

opendocman : <http://sourceforge.net/projects/opendocman>

opendocman is a web-based document management system written in PHP designed to comply with ISO 17025 and OIE standard for document

Size Estimation for IS from their CDM

Appendices

management. It features fine grained control of access to files, and automated install and upgrades.

Openrating : <http://sourceforge.net/projects/openrating>

Openrating is a web-based management system written in PHP and MySQL.

php4Flicks : <http://sourceforge.net/projects/php4flicks>

php4Flicks is a powerful PHP/MySQL movie database. The features include storing detailed information about your movies, automatically downloading all the information from IMDb.com which is the world's largest on-line movie database, organing your movie collection in user-defined categories, powerful search and sort function and etc.

Phpalumni : <http://sourceforge.net/projects/phpalumni>

Phpalumni is a web-based alumni management written in PHP/MySQL. It enables former students to stay in touch, post job offerings, search for people, spread news or organize reunions.

Phpcollegeex : <http://sourceforge.net/projects/phpcollegeex>

Phpcollegeex is a college community website with a textbook exchange, and rental finder. Future plans call for professor evaluations, slashdot style news, community calendar, and pizza delivery numbers. Designed to run as a college's unofficial student portal.

Phpman : <http://sourceforge.net/projects/phpman>

Phpman is a PHP/MySQL based system to publish your manuals and documentations. It supports multiple languages, web HTML editor (spaw) and real time PDF generation.

phpmylibrary : <http://sourceforge.net/projects/phpmylibrary>

phpmylibrary is a PHP MySQL Library automation application. The program consist of cataloging, circulation, and the webpac module. The programs also has an import export feature. The program strictly follow the USMARC standard for adding materials.

phpstudentcenter : <http://sourceforge.net/projects/phpstudentcenter>

phpstudentcenter is a powerful student management system written in PHP and MySQL.

phptimesheet : <http://sourceforge.net/projects/phptimesheet>

phptimesheet is a system that includes Timesheet/Timetracking organizer/scheduler using PHP and MySQL ----- manage workers, projects, companies, worktime ----- logging every-day come- and go-times, contracted hours, overtime, workspace and project-hours, attendance-time, and etc.

Poppawid : <http://sourceforge.net/projects/poppawid>

Poppawid is a free, full featured web based email client written in PHP. It runs on any server that can run PHP and mySQL.

Refbase : <http://sourceforge.net/projects/refbase>

Refbase is a web-based, standards-compliant, multi-user interface for managing scientific literature & bibliographic references. Based on PHP and MySQL, it offers a bibliography database with powerful search tools and automatically generated citation lists.

Replex : <http://sourceforge.net/projects/replex>

Size Estimation for IS from their CDM

Appendices

Replex is an administrative tool for managing email on any Unix-like OS. It can handle multiple domains, users, mailing lists, etc. without requiring creation of Unix user accounts for mailboxes or authentication. Based on Postfix, Cyrus, PHP and MySQL.

Timeclock : <http://sourceforge.net/projects/timeclock>

Timeclock is a simple yet effective web-based time clock system. It allows you to track all employee time as well as upcoming vacations and more, it can also replace manual sign-in sheets and such. It is written in php and utilizes a mysql database.

Uccass : <http://sourceforge.net/projects/uccass>

Uccass (Unit Command Climate Assessment and Survey System) is a PHP survey script for creating dynamic surveys. It supports unlimited questions and answers, branching logic, bar graph results with filtering, CSV export, templates, time limits and more.

Uma : <http://sourceforge.net/projects/uma>

Uma is a PHP user management and authentication application. It supports PostgreSQL and MySQL databases.

Vallheru : <http://sourceforge.net/projects/vallheru>

Vallheru is a turn based, web-browser, fantasy MMORPG written in PHP and MySQL (but may work with many other databases) with Smarty Template Engine, ADOdb and PHP Mailer. It is based on Gamers-Fusion ver 2.5 with many modifications in security, speed and features.

WebFileSystem : <http://sourceforge.net/projects/webfilesystem>

WebFileSystem is a web-based, user friendly interface, multi user, File Manager, php and mysql. Upload, download, copy, delete files and folders, share folders with other users. Rules for power users and users, advanced search history tool, manageable mail notifications and more.

Winventory : <http://sourceforge.net/projects/winventory>

Winventory (Windows Inventory) is a Computer Hardware and Software Inventory system. It is based around wmi, vbscript, mysql & php. It can be expandable via modules to include software distribution, remote control, event viewer, disk monitoring, etc.

Appendix B: Description of Open-Source PHP Systems Used for Validation

ackertodo : <http://sourceforge.net/projects/ackertodo>

ackertodo is a web-based todo list manager which supports multiple users, recurring tasks, multiple languages, categories and much more! It's written in PHP /mySQL.

Bookmark4U : <http://sourceforge.net/projects/bookmark4u/>

Bookmark4U is a web-server application, which is based on Apache, PHP and MySQL. Bookmark4U is intended to provide a comfortable bookmarking environment for users.

ByteMonsoon : <http://sourceforge.net/projects/bytemonsoon>

ByteMonsoon is originally based on the bytemonsoon tracker written by dfx. This project has a new goal of becoming a highly optimized PHP based tracker that relies on apache and MySQL allowing it to be run on virtually any webhost.

Core : <http://freshmeat.net/projects/core>

Core is a Polish Content Management System. It is a PHP and MySQL based system.

e107 : <http://sourceforge.net/projects/e107>

e107 is a website system written in PHP and MySQL. It installs a completely dynamic website on your server allowing you complete control of your site from a secure and intuitive, yet powerful and flexible admin area. Coded within the XHTML 1.1 Standard.

eFiction : <http://sourceforge.net/projects/efiction>

eFiction is an easy to use PHP/mysql based story archiving system. It allows for private or public submissions of original stories to a site through web interface. It has a fully skinnable front end and a fully functional back end.

fdcl : <http://sourceforge.net/projects/fdcl>

fdcl is a system that includes full-feature PHP/MySQL photo gallery, Camera Life automatically organizes photos, supports users and privileges, logging changes and rollback, automatic thumbnailing and a simple installer.

gallant : <http://sourceforge.net/projects/gallant>

Size Estimation for IS from their CDM

Appendices

gallant is a Gallery Program in PHP/MySQL that displays photographs, videos, text, sound bytes, and allows for the downloading of files. This system includes unlimited subgalleries, a permission system, comments, and complete customization.

Infocentral : <http://sourceforge.net/projects/infocentral>

Infocentral is a PHP/MySQL membership and management database intended for use by churches and non-profit organizations to track members, families, groups, donations, etc.

jamdb : <http://sourceforge.net/projects/jamdb>

jamdb (Just another music DataBase) is a fast PHP/DB based mp3 collection management software with many interesting features.

openrealty : <http://sourceforge.net/projects/open-realty>

openrealty is a free web based listing management application. It is intended to be both easy to setup and use. Written in PHP, openrealty is designed to be a fast and flexible tool for your website. With a large open-source community working with this application, and solid management from Transparent Technologies open-realty is the top choice for use on your website.

phpESP : <http://sourceforge.net/projects/phpesp>

phpESP is a system that can let non-technical users create surveys, administer surveys, gather results, view statistics. All managed online after database initialization. It is based on MySQL database.

Phpnews : <http://sourceforge.net/projects/newsphp>

Phpnews is very powerful News Publishing software. Features include multiple users (4 different types of access), multi language support, categories, comments, censor list, IP Banning, customisable templates.

phpollster : <http://sourceforge.net/projects/phpollster>

phpollster is a PHP/MySQL application for conducting basic polling on a website, featuring an OO coding style with db layer, preambles, ui wrapping, multi-language support, thumbnails and images, data archiving, and admin backend.

PhpScheduleIt : <http://sourceforge.net/projects/phpscheduleit>

PhpScheduleIt is a web-based resource scheduling system that allows administered management of reservations on any number of resources. Typical applications are conference room or machine reservation management. Written in PHP and tested on MySQL.

Phpsera : <http://sourceforge.net/projects/phpsera>

Phpsera is a PHP/MySQL-based tool for Search Engine Ranking Analysis (SERA). The rankings are based on parsing output of search engines, using simple regular expressions.

Phpwims : <http://sourceforge.net/projects/phpwims>

phpwims is a complete and precise wine inventory management system with an easy-to-use PHP web interface.

Phpwscookbook : <http://sourceforge.net/projects/phpwscookbook>

Phpwscookbook is a native recipe database module for the phpWebSite content management system.

plume : <http://sourceforge.net/projects/plume>

Size Estimation for IS from their CDM

Appendices

plume is a complete set of packages to create and administer a network of diskless computers used as X terminals. Stations root file system is extensively based on Debian with as few new packages as possible.

Quantum_Star_SE : <http://sourceforge.net/projects/quantumstar>

Quantum_Star_SE is a browser based space strategy game played in real time. It is currently undergoing a large scale redevelopment and will see releases in November 2005. Features: MVC design, use of Partholan Game Framework, PHP5 compatible, multiple DB support.

Rasmp : <http://sourceforge.net/projects/rasmp>

Rasmp is a fast, simple and completely flexible Content Management System (CMS, not a Portal) using PHP and supports both PostgreSQL and MySQL Databases. It is easy to integrate with any system and supports extensions for further capabilities.

Rimps : <http://sourceforge.net/projects/rimps>

Rimps is a web-based frontend, written in PHP, to make Apache an MP3 and Ogg server. It's complete with searching, song playing, and playlist management. One can have a personal mp3.com of your own audio files, plus playlists.

so-net : <http://sourceforge.net/projects/sonet>

so-net is a system that streams live multimedia through a LAN. The software uses TV Tuner card to capture live data, and then transports it through the network using Multicast protocols. The clients can then access this stream from any node in the network.

Supersurf : <http://sourceforge.net/projects/supersurf>

Supersurf project is on haitus, waiting to be completely recoded. It will be faster, more secure and hopefully an object oriented environment.

usebb : <http://sourceforge.net/projects/usebb>

usebb is an easy-to-set up and easy-to-use PHP/MySQL bulletin board system with several user and administration possibilities, especially designed for small to medium web sites who do not need the overdose of features found in other forum packages.

videodb : <http://sourceforge.net/projects/videodb>

videodb is a database to manage your personal video collection. It uses PHP and MySQL and features fetching Movie data from the Internet Movie Database (IMDb).

Webaddressbook : <http://sourceforge.net/projects/webaddressbook>

Webaddressbook is an addressbook written in PHP. At the moment it uses MySQL as backend, but ports to other databases wouldn't be difficult. It has many import and export functions like Microsoft WAB, VCF, CSV, XML, Trillian / Gaim Buddielists, whitelist.

wikiwig : <http://sourceforge.net/projects/wikiwig>

wikiwig is a Wysiwyg Wiki in php/mysql. Features are: wysiwyg online editor (htmlarea), user management, lock system to avoid 2 users edition at the same time, file management (attachement), image management (upload and online modifications), versionning and more.

yabbse : <http://sourceforge.net/projects/yabbse>

yabbse is a PHP/MySQL port of the popular forum software YaBB (Yet another Bulletin Board). Incorporating the same intuitive user interface,

and several of the most popular modifications from YaBB's Boardmod program.

zebraz : <http://sourceforge.net/projects/zebraz>

zebraz (pronounced "Zebra Zee") is a membership management system based on PHP and MySQL for clubs or organizations but could also be used for other uses which require tracking of people, their contact information, groups they belong to and payments.

ztml : <http://sourceforge.net/projects/ztml>

ztml retrieves and formats the data stored in MySQL database by using user scripts and together with any HTML or text data, which are also stored in MySQL database, and sends the output to the browser.

Appendix C: Description of Open-Source Java Systems Used for Model Building

Chatterbox : <http://sourceforge.net/projects/chatterbox-php>

Chatterbox is an HTML/JAVA/MySQL based chat application which will have support for avatars, profiles, private rooms, a messaging system and possibly a blog.

churchinfo : <http://sourceforge.net/projects/churchinfo>

churchinfo is a free church database network application written in JAVA/MySQL. This membership and management database is designed

specifically to help automate the operation of a church. It has features to track members, families, groups, donations, etc.

cream : <http://freshmeat.net/projects/creamcrm/>

cream is a customer relationship management (CRM) system that addresses the specific needs of media organizations. Built using Apache Tomcat, Velocity, and Turbine, it uses a PostgreSQL database (but other JDBC-compliant databases should work as well). It has several features specific to media organizations, such as subscription management, support for multiple products, customer communications, easy-to-use reporting and analytical functions, and tracking of special offers such as seasonal discounts and trade shows.

dlog4j : <http://sourceforge.net/projects/dlog4j>

dlog4j provides a full-featured, cross-platform, multi-database, wap-supported blog software, based on JSP/Servlet.

dynasite : <http://sourceforge.net/projects/dynasite>

dynasite is a software development tool to build database, forms, and reporting applications for .NET, ASP or PHP Web Applications. Eliminate hand-coding of your application's infrastructure by quickly generating advanced sophisticated user interfaces with custom page controls, a straightforward class hierarchy that is easily modified and extended, all the SQL statements.

e-library : <http://sourceforge.net/projects/javalibrary>

e-library is a web-based electronic library that can lends books, journals, CDs, videos to borrowers who are registered in the system, as are the

books and magazines. UML is used during OO Analysis/Design. J2EE (JSP, Servlet and JDBC) & MySQL are used for the implementation.

elips : <http://sourceforge.net/projects/opinionplugin>

elips aims to provide respectively the JSP/Java and Groovy/java communities with an open source complete and powerful web-based change management tool (or defect/bug tracker). Both tools are light, reactive and easy to install.

Forumnuke : <http://sourceforge.net/projects/forumnuke>

Forumnuke is a Communication Forum Software written in Java/Jsp intended to build a power bbs system. It's build on many Open-Source Projects such as Struts , Spring , Hibernate , OSCache and so on. High Performance is one of the key features.

GeneaPro : <http://sourceforge.net/projects/geneapro>

GeneaPro is a multi-user, cross-platform Genealogy database program based on the GenTech Genealogical Data Model. It is written in Java using the Eclipse Rich Client Platform (RCP), Hibernate persistence, and SQL/JDBC databases (HSQLDB, MySQL, PostgreSQL).

ibatis : <http://sourceforge.net/projects/ibatisdb>

ibatis will help you design and implement better persistence layers for your Java applications using two main APIs: SQL Maps for reducing JDBC code and Data Access Objects for abstracting the persistence implementation details.

i-tor : <http://sourceforge.net/projects/i-tor>

i-tor is a set of tools and technologies for Open Repositories, based on Linux, Java, MySQL, Mirage and other free components. It harvests OAI and turns databases into Open Archives. It includes similarity, backlinks and related search based on Lucene.

Itracker : <http://sourceforge.net/projects/itracker>

Itracker is a Java/J2EE-based issue / bug tracking system designed to support multiple projects with independent user bases. Features include full i18n support, multiple versions and project components, detailed histories and email notifications.

jcv : <http://sourceforge.net/projects/jcv>

jcv is a powerful multilingual Web application designed for creating, searching and printing resumes. jcv is 100% developed in Java using "best-of-breed" Open Source J2EE frameworks (SOFIA) and reporting tools (JasperReports, iReport).

jwordnet : <http://sourceforge.net/projects/jwn>

jwordnet is a pure Java standalone object-oriented interface to the WordNet database of lexical relationships.

jwp : <http://sourceforge.net/projects/jwp>

JSP WebProject (jwp) is a web-application which was originally developed to support the student research project at the university of cooperative education Stuttgart / AST Horb. It is written in JSP/Java and MySQL.

kaon : <http://sourceforge.net/projects/kaon>

kaon is an ontology management infrastructure targeted for business applications. It includes a comprehensive tool suite allowing easy ontology creation and management. Persistence mechanisms of KAON are based on relational databases.

Kbvt : <http://freshmeat.net/projects/kbvt/>

Kbvt is a tool for small libraries. It consists of a couple of Java programs and uses a MySQL database. It allows you to manage books, users, events, etc. The GUI and documentation are written in German.

malbum : <http://sourceforge.net/projects/malburn-app>

malbum is a web album generator. It allows users to quickly enter captions, generates html using css for formatting and allows you to re-open an album for updating. It requires only Java 1.4.1.

northstarbbs : <http://sourceforge.net/projects/northstarbbs>

northstarbbs (North Star Bulletin Board System) is an intelligent Java based Web Application. It was developed using the MVC architecture and should install on almost any J2EE compliant Web server. Features include flood prevention and PM system.

Personalblog : <http://sourceforge.net/projects/personalblog>

PersonalBlog is a light-weight personal blogging application that is suitable for installing on your own host provider. It's written in JAVA and uses a variety of J2EE technologies, including: ant, servlets, jsp, jdbc, hibernate, struts,tiles and log4j.

planeta : <http://sourceforge.net/projects/planeta>

Size Estimation for IS from their CDM

Appendices

planeta is a Universal Instant Messenger to unify all IM protocols. With a skinning support, Java Support, SMS, ICQ MSN Messenger protocols support, PlanetaMessenger is easy to use and very fast.

racetrack : <http://sourceforge.net/projects/racemanager/>

racetrack is a Cub Scout Pinewood Derby Race Management Application. It supports racer registration, heat scheduling, electronic timer integration, and sound effect playback. It is written in Java and runs on any Java enabled platform.

roller : <http://sourceforge.net/projects/roller>

roller is a Java-based, full-featured, and multi-user weblogging system - a database-driven J2EE web application. Roller is developed and tested on Tomcat and MySQL, but other application servers are possible.

s2j-0.94 : <http://www.bitmechanic.com/projects/s2j/>

sql-0.94 is a free, open source tool that enables Java developers to quickly map a relational database schema to a set of classes. These classes can be supplemented with your own methods and used within any Java application. The generated source code uses standard JDBC methods for persistence, and there is no runtime library required.

Sacash : <http://sourceforge.net/projects/sacash>

Sacash is a Quicken-like financial account manager. It handles multiple accounts, account transfers, and reconciles information. Tested with PostGreSQL, and MySQL.

Sixqos : <http://sourceforge.net/projects/sixqosframework>

Sixqos is a derivative of Struts. Action mappings, form beans are directly persisted to a database, thereby scaling the application over a cluster. It includes security managements and implementation of J2EE patterns, e.g, service locator, DAO, etc.

Storyserver : <http://sourceforge.net/projects/storyserver>

Storyserver is an easy to use and attractive system for tracking stories/use cases during iterative development. It is implemented as a web application and requires postgresql, a servlet engine, and a webserver.

tinapos : <http://freshmeat.net/projects/tinapos/>

tinapos is a point of sales application designed for touch screens. It supports ESC/POS ticket printers, customer displays, and barcode readers. Its multi-user and has a great backoffice with a product entry form, reports, and charts.

Webcockpit : <http://sourceforge.net/projects/webcockpit>

Webcockpit is a Web Application Generator for realtime charting and monitoring solutions. It generates JSP code to create chart images and HTML tables from SQL queries. It supports Drill-Down and pluggable look&feel and uses JFreeChart, Displaytag and Cewolf.

xc-ast : <http://freshmeat.net/projects/xcastqa/>

xc-ast analyzes data about call queues maintained by an Asterisk PBX. After selecting one or more queues, it provides information on agent logon and logoff, taken calls, and lost calls in formatted reports. It is designed to be used for monitoring call centers that require real-time activity reports.

Appendix D: Description of Open Source Java Systems Used for Validation

abaguibuilder : <http://sourceforge.net/projects/abaguibuilder>

abaguibuilder is a visual GUI builder written in Java designed to look like the Delphi/VB designers. The Java GUI Builder was conceived to help the application developer design and create UI Java applications in a rapid manner.

Art : <http://sourceforge.net/projects/art>

Art is a lightweight, multiplatform web based query tool and reporting environment. Scalable and easy to use, SQL query result can be published in a few minutes. Supports tabular, crosstab, charts, scheduling; data exportable to spreadsheet.

Bofhms : <http://sourceforge.net/projects/bofhms>

Bofhms is a toolset for pam based user management. It Includes Java client, database scheme (postgresql), pam and nss modules for postgresql (provided by the sysauth-pgsql project) and interfaces to the system (e.g. rules generation for a traffic shaper).

Contineo : <http://sourceforge.net/projects/contineo>

Contineo is a web-based document management system with user and group management, versioning, email client, and a powerful searchengine. Contineo uses the folder metaphor to structure documents and it is written in Java and Javascript..

Dspace : <http://sourceforge.net/projects/dspace>

Size Estimation for IS from their CDM

Appendices

Dspace is Open Source Digital Asset Management system. It is a groundbreaking digital repository system that captures, stores, indexes, preserves, and redistributes an organization's research data.

Ejen : <http://sourceforge.net/projects/ejen>

Ejen is a general purpose code generation system based on Java/XML/XSLT technologies. It is implemented as an Ant task and features a full EJB (BMP Entity Beans) code generation demonstration for JBoss with HSQLDB, PostgreSQL and MySQL database servers.

Imcms : <http://sourceforge.net/projects/imcms>

Imcms is a Content Management System based on Java Servlets/JSP and Microsoft SQL Server which is very simple to use. It allows you to edit page content in place using a web browser by clicking icons in the page.

Itracker : <http://sourceforge.net/projects/itracker>

Itracker is a Java/J2EE-based issue / bug tracking system designed to support multiple projects with independent user bases. Features include full i18n support, multiple versions and project components, detailed histories and email notifications.

jdbforms : <http://sourceforge.net/projects/jdbforms>

jdbforms is a Java-based (Servlet,JSP/Taglib) Rapid Application Development environment which enables developers to build web based database applications in very short time.

jmbase : <http://sourceforge.net/projects/jmbase>

jmbase is a system with an independent server implementation of parts of the CDDDB protocol. It is written entirely in Java with a MySQL database.

jwma : <http://freshmeat.net/projects/jwma/>

jwma is a web mail implementation based on the JSP Model 2 architecture. It builds on standard Java APIs (Java Mail API, Java Server Pages, and Java Servlet API) and relies on some available open source technologies (Xerces, Castor, Jakarta-ORO). jwma is designed and implemented for use with an IMAP server. It allows complete control over the store (creating, deleting, and moving folders, mailboxes, and messages). It supports MIME messages, supports text/plain message processing (e.g. making plaintext URLs "clickable"), and supports automatic sent & read mail archiving. It features a trash mailbox, includes a small addressbook, allows simple configuration of its features, and can be used in parallel with other mail clients (like Pine or Eudora).

Jbooks : <http://sourceforge.net/projects/jbooks>

Jbooks is a Java-based personal finance application. The main interface is a checkbook-like view, which allows you to enter and see your transactions. It is based around an SQL relational database, using a double-entry system for all transactions.

Jfolder : <http://sourceforge.net/projects/jfolder>

Jfolder is a business application development studio and server. It runs within a J2EE-compliant application. It consists of a workflow engine, workflow studio, and web page studio. Development and administration take place through a web browser.

Jgossip : <http://freshmeat.net/projects/jgossip/>

Jgossip is simple and powerful forum (message board) software implemented on J2EE platform using the Struts framework. It supports all the main features: reply-to-user, page navigation, email notification, and administration of users.

Jpo : <http://sourceforge.net/projects/jpo>

Jpo (Java Persistent Objects) manages Java object persistence, providing a simple framework to store objects in relational databases via JDBC/SQL.

Jwnl : <http://sourceforge.net/projects/jwordnet>

Jwnl is a Java API for accessing the WordNet relational dictionary. WordNet is widely used for developing NLP applications, and a Java API such as JWNL will allow developers to more easily use Java for building NLP applications.

mp3cattle : <http://sourceforge.net/projects/mp3cattle>

mp3cattle is a mp3 manager and ID3v1 Editor, it uses MySQL on the backend, that allows multiple users to input their rankings/genres on mp3s. Playlists can then be generated through queries on a single user or multiple user data.

openjms : <http://sourceforge.net/projects/openjms>

openjms is an Open Source implementation of Java Message Service Specification, which can be run in embedded or standalone mode. It supports the connectors for TCP, SSL, HTTP, HTTPS and RMI and can be used with any JDBC compliant database.

Openhre : <http://sourceforge.net/projects/openhre>

Size Estimation for IS from their CDM

Appendices

Openhre is Open Source Health Records Exchange system. It is dedicated to the proposition that in the National Health Information Infrastructure (NHII) Framework, patient records can be linked and exchanged using Free Open Source Software (FOSS).

Quartz : <http://sourceforge.net/projects/quartz>

Quartz is an enterprise-class job scheduler for integration with stand-alone java applications and full-scale J2EE applications. Advanced features include clustering and participation in container managed transactions.

sqlunit : <http://sourceforge.net/projects/sqlunit>

sqlunit is a unit testing framework for regression testing database stored procedures. It is written in Java/JUnit/XML.

tau_lastest : <http://sourceforge.net/projects/tau>

tau_lastest is an electronic commerce package written in PHP 3.x using PostgreSQL as backend database server. tau_lastest has a web-based interface to store students, staffs, classes, time-tables for classes, attendance of students etc.

testsuite : <http://sourceforge.net/projects/mysqljdbcctest>

testsuite is a suite of utilities to test and benchmark MySQL JDBC drivers. It is placed in the public domain to allow use without fear by proprietary driver developers.

tesuji : <http://sourceforge.net/projects/tesujigolibrary>

tesuji is a library with general purpose modules for a Go playing program by the author of 'Goliath'. It includes a tactical reader, influence module, pattern-matcher, pattern-editor and more.

Appendix E: Experimental Design and Data Collection

E.1 Experimental Design

The experiment for the proposed software size estimation model comprises three stages:

- Data collection: To collect data from both industry and Open-Source systems. The data includes data for model building and evaluation.
- Model building: To use three different software sizing techniques, multiple linear regression, support vector machine and artificial neural network, to build the software size estimation models based on the data collected.
- Model validation: To use statistical criteria to validate the resulting software sizing models from the data collected.

The experimental design is also illustrated in Figure E-1.

Figure E-1 Illustration of experimental design

We aim to collect datasets as large as possible to build the software sizing models and validate them. The datasets used in model building and model validation are separate datasets. As we have three independent variables, according to the rule of thumb, each datasets for model building must have at least 15 systems (5 times of the number of independent variables).

E.2 Data Collection

E.2.1 Preliminary Model

Project data was firstly collected only from the industry in Singapore. A preliminary method was proposed in (Tan and Zhao, 2004). This method is based on ER model using multiple linear regression technique. It uses the total number of entity types (E), total number of relationship types (R) and total number of attributes (A) to characterize an ER model. In this method, the LOC of an information system is estimated from the three variables, E , R and A . We used the method to build and validate LOC estimation models for two programming languages: Visual Basic with SQL; Java with JSP, HTML and SQL. The models were built and validated from limited datasets collected from the industry. The model that we built for Visual Basic based systems is as follows:

$$Size = 6.788 - 0.062E + 2.169R - 0.007A \quad (E.1)$$

The model that we built for Java based systems is as follows:

$$Size = 4.678 + 1.218E + 0.028R + 0.023A \quad (E.2)$$

In both models, size is in thousand lines of code (KLOC).

To continue the work on estimating of LOC for information systems from conceptual data models, further examinations have been carried out to the two models that we built for Visual Basic and Java based systems in (Tan and Zhao, 2004). We observed that the coefficients for E and A in the model for Visual Basic based systems were negative. Furthermore, from the coefficients in the two LOC estimation models, one may expect to see a consistent relative order of influence by E , R and A on LOC. However, there is no such consistent pattern reflected. In the model for Visual Basic based systems, the coefficient for R is the highest, but the coefficients for both A and E are negative. However, in the model for Java based systems, the coefficient for E is the highest, R is the second higher and A is the lowest. To address these problems, we have performed more statistical analysis of the datasets and have also investigated the accuracy of the datasets.

It is well known that collinearity (Belsley, 2004; Kennedy, 2003) can have harmful effects on multiple regression models, both in the interpretation of the results and in how they are obtained. To investigate the above mentioned observations, we run a collinearity diagnostics for the two models. From the result computed by SAS for collinearity diagnostics, we noticed that for both models, some of the condition indexes exceed the threshold value (10). For those variables with condition indexes exceed the threshold value, we further examine its proportion of variation. And, we further found that for both models, there are some variables with proportion of variation exceeding the threshold value (0.90). Thus, from statistical analysis of both models, we

conclude that collinearity (including multicollinearity) may exist among E , R and A in both models.

As most organizations in the industry have the view that their systems are confidential and therefore not accessible to external parties. Most of the data in the datasets presented in (Tan and Zhao, 2004) are supplied by users directly and there is no means for verification. After further investigation, we found some errors in these data.

E.2.2 Source of Data Collection

Based on the finding in section 4.2.1, we have decided to recollect datasets for rebuilding and revalidating models. From the previous experience, for the industry datasets, we decided to collect data only from those systems that we are allowed to verify their documentation and systems. In the datasets collected from the industry, all the conceptual data models used were extracted from requirement specifications (to reflect the early stage of software development). All data supplied by users are verified by us to ensure its correctness. Any questionable data that cannot be further verified are excluded.

Due to this reason, it is extremely difficult to collect larger datasets from the industry. To address this problem, in addition to the industry datasets, we have also collected data from open-source systems through reverse engineering.

Like Internet and personal computer, Open-Source software has become more and more popular these years. In fact, Open-Source software is as old as many other software. In the beginning, there was only free software. In 1960s software came with computers in open source and users could modify and improve it. Later, companies began unbundling software and distributed it only in executable form, so that the users could no longer modify or see it.

Selling software evolved into a separate and a huge industry with proprietary software model dominating the industry. Only recently, industry started considering Open-Source software as an option again.

For this project, most of the systems are obtained from sourceforge website (sourceforge), which is the world largest Open-Source development website, with the largest repository of Open-Source code and applications over the internet. In fact at the time when this thesis is prepared, this website has 108895 registered projects and 1206655 registered users, with major software vendors like IBM driving Open-Source development by contributing several of the open-source projects. We also collected some systems from the website freashmeet (freashmeet) which has the Web's largest index of Unix and cross-platform software, themes and related "eye-candy", and Palm OS software.

There are many code counting tools. In this project, the software used to determine the SLOC is Code Counter Pro. This software is able to count automatically for the SLOC in C/C++, Java, Delphi/Pascal, VB, PHP, APS, even including HTML, INI, TXT and other non-source code text files. Recursive options for file searching and smart comment features are also the part of the package. The result includes total counts for blank lines, comment lines, source lines, and source lines with comments. Results can also be exported to MS Excel, or saved to a CSV file for other processing.

We collected real project industry data from Singapore Computer Systems Pte Ltd, IPACS E-Solution (S) Pte Ltd, NatSteel Ltd, Great Eastern Life Assurance Co. Limited, JTC Corporation, Adroit Pte Ltd, Institute of Technical Education of Singapore, National Institute of Education and National Computer Systems Pte Ltd.

E.2.3 Data Collected for the Research Work

In this project, all the data are collected both from industry and Open-Source systems (sourceforge; freashmeet). In the datasets collected from the industry, all the conceptual data models used were extracted from requirement specifications (to reflect the early stage of software development). All data supplied by users are verified by us to ensure its correctness. Any questionable data that cannot be further verified are excluded.

In the datasets collected from Open-Source systems, all the sample systems were randomly collected from SourceForge and Freshmeat (sourceforge; freashmeet) -- the world's two largest Open-Source software development websites. For each sampled Open-Source system, the following steps were carried:

- Reverse engineer its conceptual data model from the system: A team with 5 students does the reverse engineering. We study the database schema and documentations for the system. We also test the system. From the study and testing, we identify tables in the schema that represent domain entities and concepts. We also identify the access paths between the selected tables. Then, based on the transformation of class diagrams into relational database design discussed in Chapter 13 of (Blaha and Premerlani, 1998), we work backward to identify all the classes and relationships between them.
- Count LOC (line-of-code) of the systems automatically by software tools: Comment lines, blank lines, and text files are excluded in the counting.

For systems that cater for multi-languages, the above tasks are performed only for English language. That is, any source code that is not for English language is excluded.

We have collected the following five pairs of datasets – each pair has one dataset for model building and another dataset for model validation.

- 1) Industry VB-based Systems: These are collected from systems in the industry that were developed using Visual Basic with SQL. Both datasets have 16 systems. The details are showed in Table E-1 and Table E-2.
- 2) Open-Source PHP-based Systems: These are collected from Open-Source systems that were developed using PHP with HTML and SQL. The dataset for model building has 32 systems. The dataset for validation has 31 systems. The details are showed in Table E-3 and Table E-4.
- 3) Java-based Systems: We have the following three pairs of datasets collected from systems that were developed using Java with JSP, HTML and SQL:
 - i) Industry Java-based Systems: These are collected from systems that were developed in the industry. Both datasets have 16 systems. The details are showed in Table E-5 and Table E-6.
 - ii) Open-Source Java-based Systems: These are collected from Open-Source systems. The dataset for model building has 30 systems. The dataset for validation has 24 systems. The details are showed in Table E-7 and Table E-8.
 - iii) The combined Java-based Systems: These are formed by combing the corresponding datasets from the last two pairs of datasets. Therefore, the dataset for model building has 46 systems. The dataset for validation has 40 systems.

The Open-Source systems used to derive the experiment data cover a wide range of application domains including management, administrative, health, educational and financial systems. The following gives a brief description of some of these systems, the details of whole systems are show in Appendices A, B, C and D.

The PHP-based systems include in the following:

- opendocman is a web based document management system written in PHP designed to comply with ISO 17025 and OIE standard for document management. It features fine grained control of access to files, and automated install and upgrades.
- Phpalumni is a web-based alumni management written in PHP/MySQL. It enables former students to stay in touch, post job offerings, search for people, spread news or organize reunions.
- phpmylibrary is a PHP MySQL Library automation application. The program consist of cataloging, circulation, and the webpac module. The programs also has an import export feature. The program strictly follow the USMARC standard for adding materials.
- fdcl is a system that includes full-feature PHP/MYSQL photo gallery, Camera Life automatically organizes photos, supports users and privileges, logging changes and rollback, automatic thumbnailing and a simple installer.
- Infocentral is a PHP/MySQL membership and management database intednded for use by churches and non-profit organizations to track memebbers, families, groups, donations, etc.

The Java-based systems include the following:

- cream is a customer relationship management (CRM) system that addresses the specific needs of media organizations. Built using Apache Tomcat, Velocity, and Turbine, it uses a PostgreSQL database (but other JDBC-compliant databases should work as well). It has several features specific to media organizations, such as subscription management, support for multiple products, customer communications, easy-to-use reporting and analytical functions, and tracking of special offers such as seasonal discounts and trade shows.
- e-library is a web-base electronic library that can lends books, journals, CDs, videos to borrowers who are registered in the system, as are the books and magazines. UML use during OO Analysis/Deisgn. J2EE (JSP, Servlet and JDBC) & MySQL for the implementation.
- jwp is a web-application which was originally developed to support the student research project at the university of cooperative education Stuttgart / AST Horb. It is written in JSP/Java and MySQL.
- Openhre is Open Source Health Records Exchange system. It is dedicated to the proposition that in the National Health Information Infrastructure (NHII) Framework, patient records can be linked and exchanged using Free Open Source Software (FOSS).
- Sacash is a Quicken-like financial account manager. It handles multiple accounts, account transfers, and reconciles information. Tested with PostGreSQL, and MySQL.

Table E-1 The dataset for building VB-based LOC estimation model

System No	Actual Size (KLOC)	C	R	$\bar{A}$
1	37.54	27	8	8.000
2	14.723	8	6	25.375
3	24.667	12	10	16.917
4	42.1	19	25	16.526
5	87.23	35	38	8.343
6	31.445	14	21	6.214
7	67.04	35	27	16.829
8	30.79	17	20	6.176
9	22.402	13	14	5.769
10	69.713	28	31	9.571
11	16.17	6	9	27.333
12	90.854	37	39	21.270
13	64.35	27	33	5.481
14	27.076	13	16	8.000
15	20.933	10	10	6.500
16	40.341	22	20	5.818

Table E-2 The validation dataset for VB-based LOC estimation model

System No	Actual Size (KLOC)	C	R	$\bar{A}$
1	27.217	16	8	6.188
2	14.53	7	6	16.286
3	65.872	28	24	13.179
4	41.435	24	21	14.417
5	72	36	37	15.000
6	29.52	16	16	27.563
7	52.76	31	24	10.645
8	46.92	29	24	6.931
9	97.88	42	44	33.524
10	38.764	18	12	11.000
11	31.665	12	14	8.000
12	77.52	30	32	7.533
13	16.81	10	7	9.500
14	59.332	32	26	14.781
15	107.8	41	45	16.561
16	53.66	24	18	9.125

Table E-3 The dataset for building Open-Source PHP-based LOC estimation model

System	Actual Size (KLOC)	C	R	$\bar{A}$
Bannerex3a	3.038	5	2	10.600
Castor	22.599	17	7	7.000
Cmsmadesimple	32.243	21	13	4.524
Comendar	16.164	13	11	7.077
Commerce	83.862	35	24	6.571
Coppermine	24.22	13	9	8.077
EclipseBB	63.929	35	19	8.029
Jdcms	2.543	5	3	9.400
Linkbase	6.697	5	5	7.000
Linha	55.537	25	14	8.640
Lotgd	55.752	39	10	9.077
Mailwatch	62.602	30	17	7.000
Mantis	67.111	23	22	14.957
Mundimail	2.552	3	1	8.333
opendocman	12.17	10	5	3.700
Openrating	12.757	13	9	5.000
php4Flicks	5.695	7	3	8.429
Phpalumni	7.744	9	6	9.222
Phpcollegeex	7.514	4	1	8.000
Phpman	11.054	9	9	3.667
phpmylibrary	29.77	17	15	3.412
phpstudentcenter	11.653	9	8	8.778
phptimesheet	6.847	5	4	3.600
Poppawid	13.389	7	5	11.714
Refbase	14.45	12	6	16.583
Replex	4.414	6	3	3.667
Timeclock	2.102	3	1	3.333
Uccass	42.819	20	18	3.500
Uma	4.077	4	2	9.000
Vallheru	57.408	33	14	9.242
WebFileSystem	7.428	7	3	7.000
Winventory	8.947	15	5	4.000

Table E-4 The validation dataset for Open-Source PHP LOC estimation model

System	Actual Size (KLOC)	C	R	$\bar{A}$
ackerTodo	9.47	8	5	5.000
Bookmark4u	19.381	10	10	5.800
ByteMonsoon	4.219	6	2	8.000
Core	17.065	8	5	8.125
e107	65.649	34	12	7.912
eFiction	12.443	9	6	6.222
fdcl	3.856	5	2	11.600
galant	10.225	6	4	11.333
Infocentral	47.734	19	13	8.053
jamdb	4.554	4	5	8.500
openrealty	16.846	9	8	6.333
phpESP	33.645	16	15	5.438
Phpnews	6.532	6	5	6.167
phpollster	1.818	4	3	7.000
PhpScheduleIt	16.006	6	9	8.667
Phpsera	15.846	6	6	7.000
Phpwims	10.507	6	4	9.333
Phpwscookbook	7.607	5	5	10.200
plume	15.875	11	7	6.455
Quantum_Star_SE	57.643	30	23	9.533
Rasmp	9.452	10	4	6.600
Rimps	10.81	10	6	6.500
so-net	38.781	20	14	10.100
Supersurf	1.489	4	3	7.000
usebb	6.118	6	3	13.333
videodb	24.227	14	10	4.643
Webaddressbook	8.363	3	2	20.000
wikiwig	23.457	15	9	5.800
yabbse	22.475	11	10	12.818
zebraz	8.59	6	5	7.833
ztml	11.724	10	3	9.600

Table E-5 The dataset for building Industry Java-based LOC estimation model

System No	Actual Size (KLOC)	C	R	$\bar{A}$
1	30.02	22	6	8.182
2	37.288	16	18	6.188
3	60.102	23	26	9.261
4	46.27	24	18	6.583
5	85.009	42	26	5.738
6	12.62	6	5	6.167
7	80.014	40	25	6.675
8	47.658	20	20	5.200
9	20.53	10	11	7.500
10	56.48	24	25	7.792
11	33.602	14	15	7.286
12	63.1	20	26	14.2
13	92.841	45	26	11.622
14	22.08	10	13	6.400
15	14.89	7	6	5.714
16	100.213	52	27	9.481

Table E-6 The validation dataset for Industry Java-based LOC estimation model

System No	Actual Size (KLOC)	C	R	$\bar{A}$
1	84.89	49	22	5.898
2	20.446	12	7	4.250
3	17.29	8	6	7.750
4	34.075	16	12	5.125
5	40.113	18	14	6.389
6	28.722	17	14	7.412
7	71.37	31	27	6.871
8	60.34	30	29	10.300
9	20.45	11	11	5.091
10	93.27	35	36	8.486
11	31.69	19	20	4.789
12	77.52	30	33	6.267
13	45.384	21	24	7.714
14	52.1	24	28	6.000
15	19.36	7	9	4.857
16	56.744	18	26	5.944

Table E-7 The dataset for building Open-Source Java-based LOC estimation model

System	Actual Size (KLOC)	C	R	$\bar{A}$
Chatterbox	11.717	8	6	4.250
churchinfo	47.52	23	19	9.565
cream	84.01	26	40	11.462
dlog4j	26.999	15	14	8.933
dynasite	41.72	20	15	5.900
e-library	13.015	5	6	12.400
elips	30.402	18	7	6.611
Forumnuke	29.159	23	10	6.957
GeneaPro	53.443	28	25	4.179
Ibatis	18.694	13	9	6.615
i-tor	26.384	16	6	5.125
Itracker	38.721	19	16	6.579
jcv	75.643	26	30	6.154
jwordnet	46.72	21	24	6.048
jwp	6.413	7	5	4.143
kaon	79.534	20	37	4.850
Kbvt	36.343	18	17	5.333
malbum	59.684	22	31	6.182
northstarbbs	50.454	15	20	11.600
Personalblog	3.055	4	1	7.000
planeta	63.257	34	17	3.971
racetrack	91.28	35	28	13.571
roller	32.707	11	17	7.545
s2j-0.94	11	5	5	3.600
Sacash	5.543	6	4	3.833
Sixqos	22.686	12	11	6.667
Storyserver	3.911	3	2	6.667
tinapos	20.841	14	7	3.000
Webcockpit	9.269	6	5	3.500
xc-ast	7.732	7	2	11.143

Table E-8 The validation dataset for Open-Source Java-based LOC estimation model

System	Actual Size (KLOC)	C	R	$\bar{A}$
abaguibuilder	83.176	29	23	7.345
Art	17.67	14	7	6.929
Bofhms	10.507	12	4	4.333
Contineo	21.067	11	11	7.364
Dspace	47.57	32	20	4.250
Ejen	4.437	5	3	4.600
Imcms	91.424	59	30	4.712
Itracker	38.721	19	14	6.579
jdbforms	50.72	19	15	6.789
jmbase	19.723	8	8	5.250
jwma	63.228	22	19	8.045
Jbooks	12.817	11	7	3.727
Jfolder	29.288	10	8	12.800
Jgossip	28.047	14	9	5.714
Jpo	8.54	5	5	5.800
Jwnl	6.913	7	5	3.571
mp3cattle	14.598	9	10	3.444
openjms	59.72	17	16	6.824
Openhre	9.932	9	4	3.444
Quartz	22.954	11	11	4.364
sqlunit	74.089	26	20	7.808
tau_lastest	48.039	14	16	8.000
testsuite	30.076	15	9	5.067
Tesuji	8.01	2	1	16.000