Han, W.M. (2014), 'A Decision Support Approach to the Selection of Functional Size Measurement Methods', *Journal of Information Management*, Vol. 21, No. 2, pp. 185-206

# A Decision Support Approach to the Selection of Functional Size Measurement Methods

Wen-Ming Han*
Department of Management Information System, Takming University of Science and Technology

## Abstract

Practitioners require realistic size measurement techniques to track scope creep and to estimate effort. However, using an unsuitable functional size measurement method (FSMM) to measure software size is ineffective. This study proposes a decision support approach to assist managers who lack adequate software engineering knowledge and expertise to perform functional size measurements when selecting a suitable FSMM among candidate FSMMs. A chi-squared automatic interaction detection (CHAID) model was developed based on 14 information system characteristics obtained from 537 International Software Benchmarking Standards Group (ISBSG) projects. A gains chart and 10-fold cross-validation showed the performance of the proposed approach. Four IS characteristics were retained in the final approach, including distributed data processing, facilitate change, complex processing, and multiple sites, thereby enabling meaningful decisions to be made on whether function point analysis can be used to quantify the functional requirements of unprecedented software. This study proposes a structured and traceable method rather than a rule-of-thumb method to support decision-making for FSMM selection problems. In addition, our findings revealed crucial features for management information system (MIS) projects that should be carefully controlled to ensure success.

**Keywords:** Functional size measurement, FSM, Function point analysis, Software metrics, ISO/IEC 14143.

* Corresponding author. Email: wmhan@takming.edu.tw
2013/1/17 received; 2013/10/19 revised; 2014/1/9 accepted

韓文銘（2014），『支援軟體功能性規模度量方法論選擇之決策模式』，
*資訊管理學報*，第二十一卷，第二期，頁 185-206。

# 支援軟體功能性規模度量方法論選擇之決策模式

韓文銘*
德明財經科技大學資訊管理系

## 摘要

實務者需要可靠的規模度量技術以便追蹤範疇遞增與工作量預估，然而，使用不適當的功能性規模度量方法論來度量軟體規模是沒有意義的。本研究提出一個決策模式來協助經驗不足的管理者選擇一個相對適當的軟體功能性規模度量方法。這個以 CHAID 決策樹為基礎的模式是經由使用包含 14 個系統特徵值的 537 筆 ISBSG 專案資料所建立，接著再使用累積增益圖與 10 等份交叉驗證法來檢視與呈現模式決策績效。共有 4 個系統特徵值（分散式資料處理、容易修改、複雜邏輯與多個站點）被保留於最後的決策模式中以便支援一個無前例可循系統是否適合用功能點分析來度量。因此這個結構化且可追蹤的決策模式可以用來支援軟體功能性規模方法的選擇問題，此外，本研究亦發現四個可確保管理資訊系統專案成功的關鍵因素。

**關鍵詞**：功能性規模度量、功能性規模度量方法論、功能點分析、軟體度量、ISO/IEC 14143

* 本文通訊作者。電子郵件信箱：wmhan@takming.edu.tw
2013/1/17 投稿；2013/10/19 修訂；2014/1/9 接受

# 1. Introduction

Functional size measurement method (FSMM), when correctly employed, provides useful quantitative size information for cost estimation and supporting requirement changes. FSMMs were conceptually derived from function point analysis (FPA) (Albrecht 1979), which was designed to provide a technology-independent measure for estimating development and maintenance efforts, analyzing productivity, and comparing the functionality of alternative systems. This method describes functionality using five user-identifiable, logical "function" types (two data function types and three transactional function types). For a given software application, each of these elements is quantified and weighted and their characteristic elements are counted, such as logical fields or file references. Presently, FPA is regulated by the International Function Point Users Group (IFPUG).

The inadequacy of FPA in the non-MIS domain (e.g., real-time software) has been identified as a significant drawback (Reifer 1990; Abran et al. 1997; Jones 2008). Thus, in case of non-MIS systems, functional size measurements using FPA are neither complete nor adequate, which may adversely affect cost estimation and resource allocation. Several approaches have been adopted to develop alternative sizing methods that would overcome the known limitations of FPA (e.g., Boeing 3D function points (Whitmire 1995) and COSMIC (Abran et al. 1999)). Presently, five methods for functional size measurement are fully compliant with the ISO/IEC 14143-1 standard (2007) and have been approved as ISO standards (ISO/IEC 20968 2002; ISO/IEC 24570 2005; ISO/IEC 20926 2009; ISO/IEC 29881 2010; ISO/IEC 19761 2011).

Unless the premise of employing the most appropriate FSMM is fulfilled for measuring a given project, any measurement effect will prove insignificant. For example, according to existing literature, the underlying structure of FPA is more appropriate for measuring MIS software (Albrecht & Gaffney 1983; Kitchenham 1997; Dumke & Albrecht 2011). Thus, although International Function Point Users Group (IFPUG) specifies extra white papers (IFPUG 2007; IFPUG 2008; IFPUG 2009a) and case studies (IFPUG 2005; IFPUG 2009b) for measuring the functional sizes for various non-MIS systems, the measurements made may not fully reflect the "pure" functional sizes of such systems, resulting in further inaccuracy in cost estimation and schedule planning.

With rapid developments in information technology, the application type of the

software can vary to suit specific business requirements (e.g., knowledge management system, global logistics management system, etc.). Unfortunately, it is impossible to have a readily available practical FSMM selection guideline for such unprecedented software. Therefore, the selection of an FSMM is a problematic task during the quantification of functional user requirements by engineers. One possible way to minimize this problem is to develop an empirical decision approach to selecting a relatively suitable FSMM from among candidate FSMMs.

ISO/IEC 14143-6 (2006) presents a FSMM selection process to assist users in selecting an effective FSMM. More specifically, this FSMM selection process begins with categorizing a software system to be developed into functional domains as application types (e.g., MIS and real time) to ensure the appropriate corresponding measured functional size. Undoubtedly, the type of functional domain is limited compared with the information system (IS) categorization, which is continuously varied. Therefore, the question arises of how to analyze the characteristics of unprecedented software to judge the similarity with existing functional domains. If the difference between unprecedented software and functional domains can effectively enhance quantitative understanding using similarity analysis, then these reflective indicators can provide early signs to assist engineers in selecting a relatively appropriate FSMM among candidate FSMMs.

In this study, we emphasize the importance of selecting an appropriate FSMM, and we propose a useful decision support approach in order to assist project managers. The remainder of this paper is organized as follows. In Section 2, we review studies related to FSMMs and discuss the nature of information systems. In Section 3, we introduce the International Software Benchmarking Standards Group (ISBSG) dataset and briefly explain the basic concepts of a decision tree. In Section 4, we describe the steps in the development of the proposed empirical approach. In Section 5, the experimental results are presented and compared with the results obtained without the prediction model. Finally, the conclusions and scope for future research are discussed in Section 6.

## 2. Functional Size Measurement Method

FPA inspired the concept that the size of an information system can be quantified in terms of the functions delivered to the user in order to overcome the limitations in LOC-based measurement and analysis. Since the public release of FPA in 1979, several sizing methods based on FPA have been developed for non-MIS environments

(Whitmire 1995; Abran et al. 1999). However, the inconsistency in the terminology definitions, concepts scope, and counting steps of these methods increases the users' learning costs. Therefore, International Organization for Standardization (ISO) has published ISO/IEC 14143-series standards to eliminate such inconsistencies and to define explicit principles for the development of new FSMMs. Presently, five FSMMs have been approved by ISO. Figure 1 show the evolution of these five FSMMs.

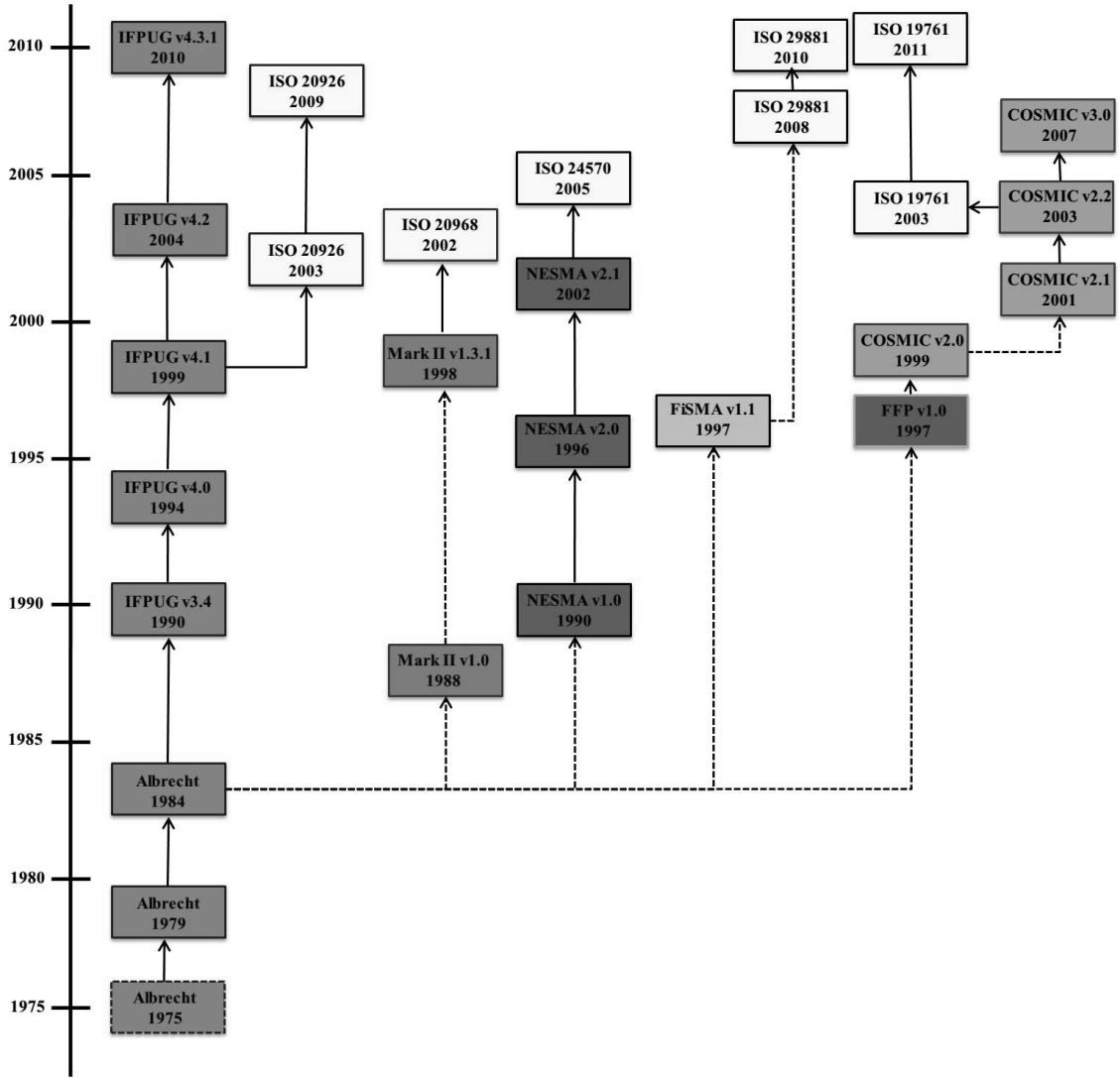


Figure 1: Evolution of FSMMs

As is shown in Figure 1, ISO/IEC 20926 is a transposition of the IFPUG 4.1 Unadjusted FSMM. In addition, each of the four retained additional FSMMs applies its

own approach to measure the functional size of a piece of software. ISO/IEC 19761 specifies the set of definitions, conventions, and activities of the COSMIC FSMM. ISO/IEC 20968 includes the counting of entities and relationships in a data model and computes function points by multiplying two factors: information processing size and technical complexity adjustment. ISO/IEC 24570 calculates Enhanced Function Points by weighting the function points with an impact factor. The impact factor ranges from 0.25 to 1.5 depending on the extent to which the functionality of the application is impacted by the enhancement. ISO/IEC 29881 is service oriented, and identifies seven distinct base functional component classes provided by the piece of software.

ISO/IEC 14143-5 (2004) states that the measuring capabilities of FSMMs can be varied in different functional domains such as software types (e.g., MIS and real time) so that users may identify the functional domain to which the measured software belongs before they carry out functional size measurement. Unfortunately, with the rapid diversification of software types, it is difficult to categorize new software types, i.e., unprecedented software, into existing functional domains. Presently, "How to select the most appropriate FSMM among candidate FSMMs?" is the most frequently asked question on the IFPUG Bulletin Board.

Past studies on FSMMs have largely focused on size conversion among various FSMMs (Cuadrado-Gallego et al. 2010), size counting from entity relationship - data flow diagram specifications or business process models (Živkovič et al. 2005), and correlation analysis of base functional types (Buglione & Gencel 2008). Work indicating the affecting factors of FSMM selection, including supporting software tools, familiarity of existing FSMMs, and training/consulting resources, is scant (Ebert et al. 1998; ISO/IEC 14143-6 2006; Total Metrics Inc. 2007). However, the practical value of these studies is poor because of the lack of a decision support model, which has higher usability than the subject with bias such as cognitive differences and untraceable causality. Therefore, this study develops a useful decision support approach to select an appropriate FSMM among candidate FSMMs to facilitate adequate size measurement. This is an area that has not been well covered.

# 3. Material and Methods

## 3.1 ISBSG Dataset

The dataset used in this study was acquired from the International Software Benchmarking Standards Group. The ISBSG data repository release 10 comprises 4106

completed software projects. Each project in this dataset is enrolled by over 50 meaningful fields (e.g., productivity, effort, and size attributes). Therefore, several researchers (Huang and Han 2006; Sentas et al. 2008) have used this dataset in software engineering studies. Table 1 lists the 15 fields used in this study; they are classified into two groups: project classification variables and general system characteristics.

Table 1: Fields used in this study

| Variable | Code | Scale | Definition |
|---|---|---|---|
| | | | Project classification variables |
| Application type | AT | Nominal | The nature of software (e.g., MIS, real time) |
| | | | Definition |
| | | | General system characteristics |
| Data Communication | G01 | Ordinal | The degree to which the application communicates directly with the processor. |
| Distributed Data Processing | G02 | Ordinal | The degree to which the application transfers data among physical components of the application |
| Performance | G03 | Ordinal | The degree to which response time and throughput performance considerations influenced the application development |
| Heavily Used Configuration | G04 | Ordinal | The degree to which computer resource restrictions influenced the development of the application |
| Transaction rate | G05 | Ordinal | The degree to which the rate of business transactions influenced the development of the application |
| Online Data Entry | G06 | Ordinal | The degree to which data is entered or retrieved through interactive transactions |
| End User Efficiency | G07 | Ordinal | The degree of consideration for human factors and ease of use for the user of the application measured |
| Online Update | G08 | Ordinal | The degree to which internal logical files are updated on-line |
| Complex Processing | G09 | Ordinal | The degree to which processing logic influenced the development of the application |

| | | | |
|---|---|---|---|
| Reusability | G10 | Ordinal | The degree to which the application and the code in the application have been specifically designed, developed, and supported to be usable in other applications |
| Installation Easy | G11 | Ordinal | The degree to which conversion from previous environments influenced the development of the application |
| Operational Easy | G12 | Ordinal | The degree to which the application attends to operational aspects, such as start-up, back-up, and recovery processes |
| Multiple Sites | G13 | Ordinal | The degree to which the application has been developed for different hardware and software environments |
| Facilitate Charge | G14 | Ordinal | The degree to which the application has been developed for easy modification of processing logic or data structure.. |

We regard the 14 general system characteristics (GSCs) of FPA as meaningful variables that reflect the different types of software applications on the basis of the following factors. First, the fourteen GSCs represent the inherent features of information systems. Hence, they can be individually used to classify software applications (Lokan 2000; Garmus and Herron 2001; Huang and Han 2006). Second, each characteristic has explicit definitions and assessment guidelines that guarantee the reliability and consistency of quantifiable evaluations (IFPUG 2011). Third, ISBSG Release 10 (R10) Data Repository includes considerable historical data on GSCs, which provides an effective foundation for model construction. Table 2 and 3 presented a scoring example and hits of distributed data processing.

Table 2: Scoring example of distributed data processing

| Score | Descriptions |
|---|---|
| 0 | Data is not transferred or processed on another component of the system |
| 1 | Data is prepared for transfer, then is transferred and processed on another component of the system, for user processing |
| 2 | Data is prepared for transfer, then is transferred and processed on another component of the system ,not for user processing |

| Score | Descriptions |
|---|---|
| 3 | Distributed processing and data transfer are on-line and in one direction only. |
| 4 | Distributed processing and data transfer are on-line and in both directions |
| 5 | Distributed processing and data transfer are on-line and are Dynamically performed on the most appropriate component of the system |

Table 3: Hits for the scoring of distributed data processing

| Score | Hints |
|---|---|
| 0 | Presentation, processing, and I/O components are all in the same place. |
| 1 | 1. Application downloads data to a user's client machine, so the user can use Excel or other reporting tools to prepare graphs and perform other analysis.<br>2. Process that transfers data from mainframe to an external component for user processing. This transfer is performed using a simple protocol such as FTP.<br>3. Transferred to a user for processing. |
| 2 | 1. Process that transfers data from mainframe to mid-tier. For example, processing with SAS-PC.<br>2. Application sends data to client or server. This data is then processed or used to produce reports, etc. No data or confirmation is sent back to the client or server.<br>3. Transferred to a component for processing. |
| 3 | 1. Data is sent between client and server in one direction only.<br>2. This data is then processed or used to produce reports, etc. by the receiving application. This data typically includes transactions that update an ILF on the client or server.<br>3. For example – client-server or web-enabled applications. |
| 4 | 1. Data is sent between client and server in either direction. This data is then processed or used to produce reports, etc. by the receiving application. This data typically includes transactions that update an ILF on the client or server.<br>2. For example - client-server or web-enabled applications.<br>3. The application runs under an operating system that automatically handles the allocation between components, however, the use of the operating system did not influence the design and implementation of the application |
| 5 | 1. The developer must consider special application software that looks at multiple processors and runs the application on a specific type of processor. This is invisible to the user. |

| Score | Hints |
|---|---|
| | 2. The application runs under an operating system that automatically handles the dynamic allocation between components, and the use of the operating system specifically influenced the design and implementation of the application. |

### 3.2 Decision Tree

A decision tree (DT) is a popular data-mining approach to solving classification and prediction problems. Unlike other data mining techniques that cannot be easily understood, interpreted, or applied to research results (e.g., neural networks and k-means), a DT can provide visual flowchart-like diagrams and if-then rules to aid managers in decision making. Therefore, DTs have been applied to a wide variety of business problems, especially credit scoring and disease detection (Lee et al. 2006; Yeh et al. 2011).

A DT is a supervised learning approach that learns the implicit patterns between known inputs and known outputs to expose hidden interactions and create a predictive model. Presently, many algorithms can be used to generate DTs, including Iterative Dichotomiser 3 (Quinlan 1979), Chi-squared Automatic Interaction Detection (CHAID) (Kass 1980), and Classification and Regression Trees (Breiman et al. 1984). In this study, we used the CHAID algorithm because it can provide more than two branches of obtained results.

CHAID can explore interactions between nominal, ordinal, and continuous variables, and it depicts results in a hierarchical tree structure, which provides results are that are easy to read and interpret. A CHAID tree is constructed by repeatedly splitting subsets of a space into two or more child nodes, beginning with an entire data set (Michael & Gordon 1997). To determine the best split at any node, any appropriate pair of categories of predictor variables is merged until there is no statistically significant difference within the pair with respect to a target variable. CHAID handles all missing values by treating them as a single valid category and does not perform pruning.

## 4. Approach Outline

As shown in Figure 2, the proposed methodology involves three distinct steps.

In Step 1, all 4106 software projects in the ISBSG R10 were preliminarily analyzed

in order to obtain a suitable subset of projects that would facilitate model building in the next step. Table 4 lists the four filtration tasks in the data preparation step.

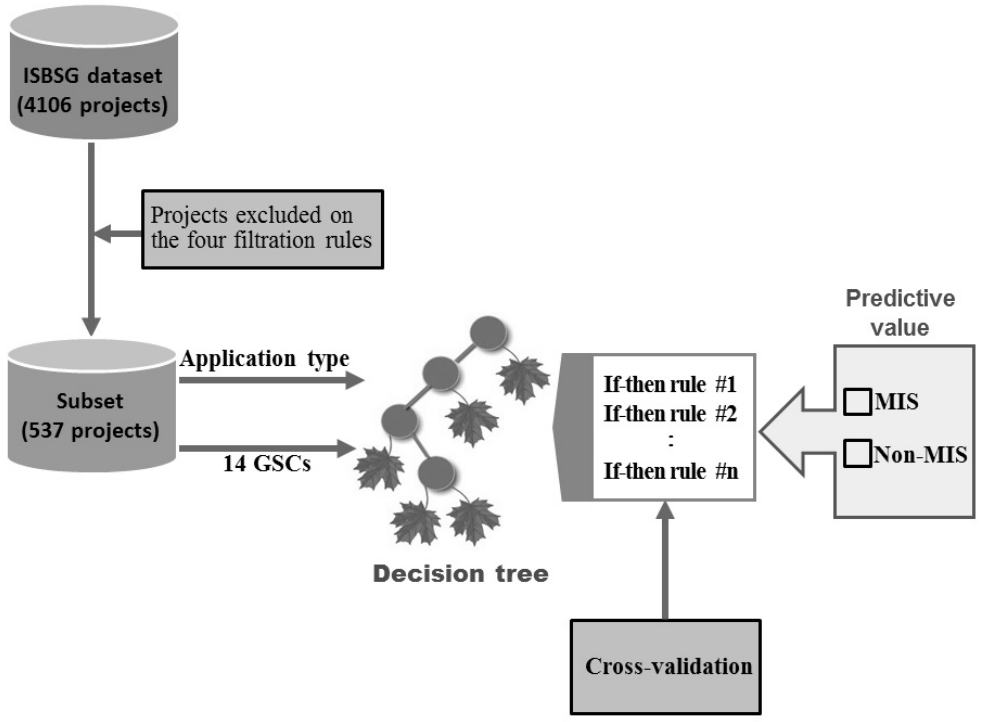


Figure 2: Steps in the Proposed Method

Table 4: Filtration of ISBSG R10

| Task | Field | Filter | Projects excluded | Projects remaining |
|---|---|---|---|---|
| 1 | Data Quality Rating | ≠{ Null-value \| C \| D } | 295 | 3811 |
| 2 | Application Type | ≠ { Null-value \| Not specified } | 1121 | 2690 |
| 3 | Any sets of the 14 GSCs | ≠{ Null-value \| 0 \| 3 \| 2&3} | 2133 | 557 |
| 4 | Count Approach | ≠{ Null-value \| COSMIC \| FiSMA \| NESMA} | 20 | 537 |

First, the projects with A and B ratings were obtained from the field "Data Quality Rating." This principle is based on an ISBSG recommendation, and it has been widely adopted in previous studies based on ISBSG R10 to minimize the influence of defective data (Buglione and Gencel 2008; Cuadrado-Gallego et al. 2008). Second, since the

application type and individual GSC fields are necessary for this study, projects with a single missing value among the 15 fields are excluded. Third, the ISBSG repository manager indicated that sets of the 14 GSCs with default values (all GSCs = 0, all GSCs = 3, or all GSCs = 2 & 3) can be ignored. Finally, we selected those projects whose "Count Approach" was acquired from IFPUG. Thus, a sample of 537 projects remained.

Unfortunately, 37 of the remaining 39 application types contain less than 20 projects. In order to ensure the reliability of the proposed approach, the 537 projects are thus classified on the basis of them being MIS and non-MIS software. Table 5 lists the characteristics of the 537 projects selected from the ISBSG dataset.

Table 5: Characteristics of 537 projects used in the study

| Project characteristics | Total (n=537) | MIS (n=193) | Non-MIS (n=344) |
|---|---|---|---|
| Project Elapsed Time | | | |
| Minimum | 1 | 1 | 1 |
| Mean | 8.70 | 9.57 | 8.23 |
| Maximum | 84 | 84 | 78 |
| Summary work effort | | | |
| Minimum | 26 | 90 | 26 |
| Mean | 5255.96 | 5289.80 | 5238.10 |
| Maximum | 150040 | 58700 | 150040 |
| Max team size | | | |
| Minimum | 1 | 1 | 1 |
| Mean | 7.89 | 5.69 | 8.71 |
| Maximum | 53 | 25 | 53 |
| Project delivery rate | | | |
| Minimum | 0.30 | 0.70 | 0.3 |
| Mean | 15.81 | 16.63 | 15.36 |
| Maximum | 108.30 | 98.40 | 108.30 |

In Step 2, the 537 projects were used to build a CHAID tree. This tree architecture has 15 attributes—14 GSCs as the input variables and the application type as the output variable. The chi-square statistic to determine node merging and splitting was calculated using the Pearson ratio. The significance level for merging and splitting nodes was set to 0.05, and the maximum depth of the CHAID tree was three. SPSS 18.0 was used to construct the CHAID trees.

As the final step, we evaluated the predictive accuracy of the proposed model on

the basis of the overall misclassification rate and cross validation. In addition, we used a cumulative gains chart to visualize the actual performance with and without the predictive model.

# 5. Results

## 5.1 Classification Tree and Rules

Classification trees are charts that illustrate decision rules (Table 6). Such decision rules provide specific information about risk factors based on rule induction. As is shown in Figure 3, the resulting classification tree includes nine leaf nodes, six of which are terminal nodes. Each node of the decision tree can be expressed in terms of an "if-then" rule. For example, if an observation whose Distributed Data Processing value is between 0 and 1 and Facilitate Change is between 2 and 5 it falls under terminal node 5, whose similarity to MIS is 56.2%. Similarity to MIS can be regarded as the risk involved in the use of the FSMM. In the example described above, we believe that IFPUG FPA is the most appropriate alternative for measuring the functional size of this software.

Of the 14 predictor variables that were considered for CHAID analysis, only four were retained in the final model—Distributed Data Processing (G02), Facilitate Change (G14), Complex Processing (G09), and Multiple Sites (G13). The first split was observed to occur on the basis of Distributed Data Processing; thus, Distributed Data Processing is the most important predictor, and it is associated with three branches (*p-value = 0.000*).

The following observations were made in the three branches of the parent node. (1) At the first intercourse, Facilitate Change and Complex Processing are the best predicting variables for projects with low (<=1) and high (>2) Distributed Data Processing values, respectively. (2) For projects with a medium Distributed Data Processing (1-2) value, Distributed Data Processing is the only significant predictor for discriminating between MIS and non-MIS projects. In this case, 86.5% of the projects fall into the non-MIS category. (3) At the first intercourse, for projects with high Distributed Data Processing (>2) and low Complex Processing (<=2) values, the next significant variable is Multiple Sites; projects with a high Multiple Sites (<=1) value have a higher probability of falling into the non-MIS category. Table 6 summarizes the six extracted rules in terms of their degrees of confidence in assisting users with determining the similarity to MIS projects.

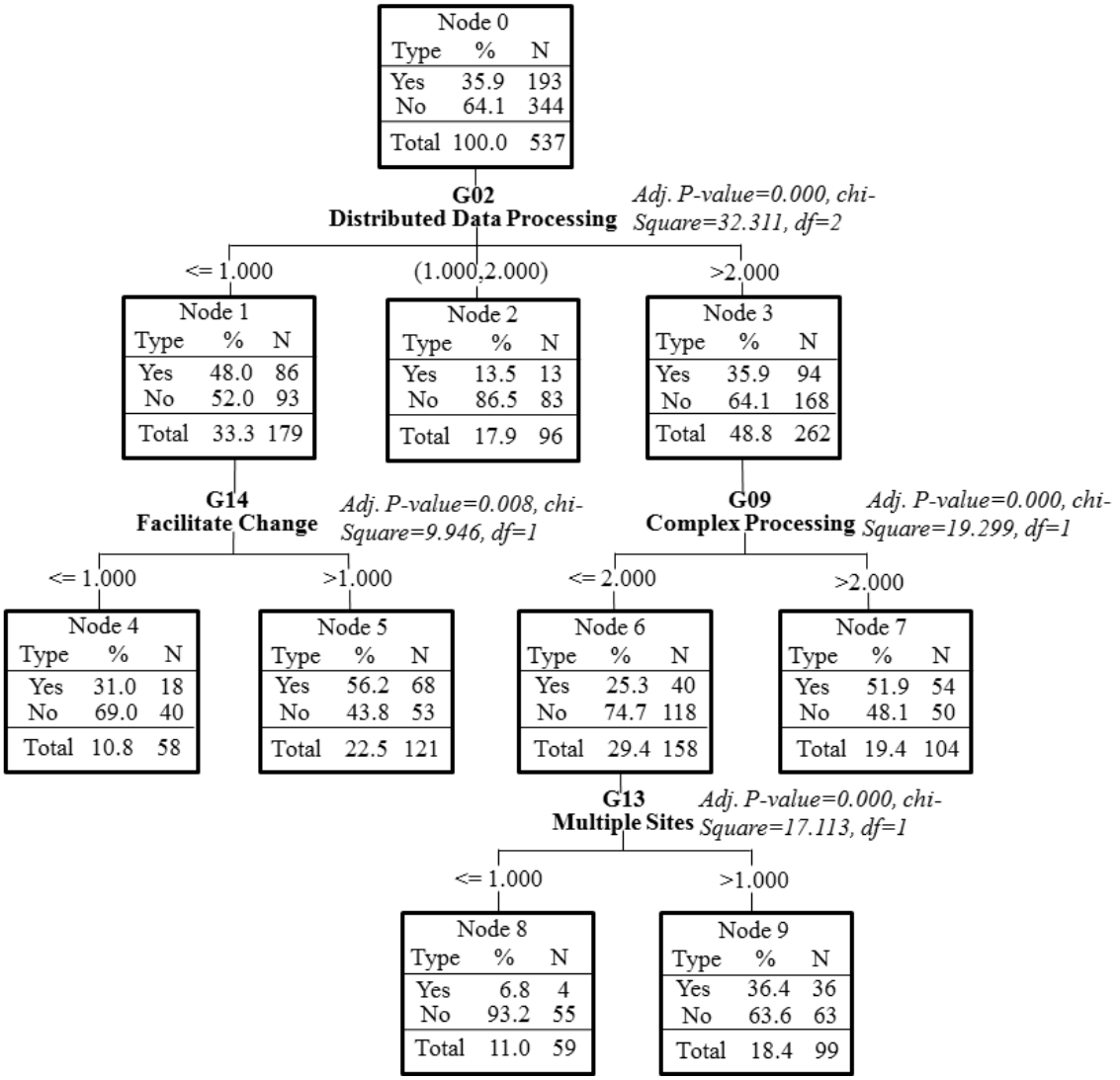


Figure 3: Output of CHAID Tree

Table 6: Decision rules for prediction of MIS similarity

| Rule # | Node | If | Then | Correct Rates |
|---|---|---|---|---|
| Rule 1 | 8 | G02 > 2 and G09 < = 2 and G13 < = 1 | Non-MIS | 93.22% |
| Rule 2 | 2 | G02 > 1 and G02 < = 2 | Non-MIS | 86.46% |
| Rule 3 | 4 | G02 < = 1 and G14 < = 1 | Non-MIS | 68.97% |
| Rule 4 | 9 | G02 > 2 and G09 < = 2 and G13 > 1 | Non-MIS | 63.64% |
| Rule 5 | 5 | G02 < = 1 and G14 > 1 | MIS | 56.20% |
| Rule 6 | 7 | G02 > 2 and G09 > 2 | MIS | 51.92% |

As shown in Table 7, the overall misclassification rate for the proposed model is 32.4%, indicating that 67.6% of the projects will be classified correctly using the current CHAID decision tree. In addition, the 10-fold cross-validated results indicate that the average risk for all the trees is 0.341.

Table 7: Prediction results of CHAID model

| Actual Type | Classified type | | Correct |
|---|---|---|---|
| | MIS | Non-MIS | |
| MIS | 122 | 71 | 63.2% |
| Non-MIS | 103 | 241 | 70.1% |
| Total Percent | 41.9% | 58.1% | 67.6% |

## 5.2 Model Performance

The gains chart provides a useful summary that reflects the performance of the proposed model. In the gains chart, the nodes are sorted by the number of cases in the target category (MIS projects) for each node. For a categorical target variable, the gain score equals the percentage of cases with the target category in this case. The cumulative statistics provide a better understanding of the effectiveness of the predictive model than that in the case without the predictive model (baseline), which involves guesswork. The node-by-node statistics and cumulative statistics are summarized in Table 7.

Table 8: Gains chart obtained using CHAID algorithm

| Node | Node-by-Node | | | | | | Cumulative | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Node | | Response | | Gain (%) | Index (%) | Node | | Response | | Gain (%) | Index (%) |
| | N | % | N | % | | | N | % | N | % | | |
| 5 | 121 | 22.5 | 68 | 35.2 | 56.2 | 156.4 | 121 | 22.5 | 68 | 35.2 | 56.2 | 156.4 |
| 7 | 104 | 19.4 | 54 | 28.0 | 51.9 | 144.5 | 225 | 41.9 | 122 | 63.2 | 54.2 | 150.9 |
| 9 | 99 | 18.4 | 36 | 18.7 | 36.4 | 101.2 | 324 | 60.3 | 158 | 81.9 | 48.8 | 135.7 |
| 4 | 58 | 10.8 | 18 | 9.3 | 31.0 | 86.3 | 382 | 71.1 | 176 | 91.2 | 46.1 | 128.2 |
| 2 | 96 | 17.9 | 13 | 3.7 | 13.5 | 37.7 | 478 | 89.0 | 189 | 97.9 | 39.5 | 110.0 |
| 8 | 59 | 11.0 | 4 | 2.1 | 6.8 | 18.9 | 537 | 100.0 | 193 | 100.0 | 35.9 | 100.0 |

As shown in Table 8, for node 5, 68 of 121 projects are MIS projects (i.e., the MIS-project rate is 56.2%). The index score denotes the proportion of MIS projects for this particular node to the overall proportion of MIS projects. For node 5, the index score is around 156.4%, indicating that the proportion of respondents for this node is around 1.5 times the MIS-project rate for the overall sample. In other words, a random guess about the similarity of unprecedented software to an MIS application without the use of a model has a 50% chance of being correct. In order to obtain a minimum hit rate of at least 50% in the CHAID mode, nodes 5 and 7 are considered. This information provides additional objective evidence that the predictive accuracy of the CHAID model is superior to that of the baseline case (random guesswork).

# 6. Conclusion

## 6.1 Summary of Results

With the increasing diversification and costs of software projects, managers require realistic size measurement techniques to effectively track scope creep, estimate effort, and determine productivity. Unfortunately, although many FSMM solutions are available for sizing software, it is difficult to select the most suitable FSMM for a given application. In order to overcome this challenge, we propose an empirical approach that enables managers lacking adequate software engineering knowledge and expertise in functional size measurement to select a relatively suitable FSMM among candidate FSMMs. By our approach, the results are presented in a comprehensible manner using CHAID decision trees, which can be easily understood, interpreted, and applied by users.

The proposed approach provides a prediction accuracy rate of 67.6%. Therefore, when a new project with unprecedented software is correctly predicted as a MIS or non-MIS project, managers have sufficient information to make a meaningful decision as to whether FPA can be suitably employed to quantify the functional requirements of the project. The results also indicate that distributed data processing, facilitate change, complex processing, and multiple sites can be employed as reliable predictor variables to determine the similarity to MIS software. Not only do our findings facilitate a better understanding of MIS applications, but also provide possible breakthroughs in management thinking to ensure success of MIS projects.

## 6.2 Limitations and Future Research

Although this approach inspires novel methods for improving the accuracy of functional size measurement, this work has some limitations. This is the first study to apply a structured and traceable method rather than a rule of thumb method to enable project managers to select an appropriate FSMM. Because the data size is only robust enough to discriminate two application types and based on the claims of previous significant works that IFPUG and FPA are applicable to measure MIS type applications, our approach suggests (i.e., to use FPA) that readers use caution when applying this approach to real-world cases.

Lokan (2005) indicated that the GSC profile also reflects application types, although the 14 GSC are initially defined in IFPUG to reflect the relative effort required for various projects. The 14 GSC has recently been regarded as options to new IFPUG Counting Practices Manual and excluded from FSMM standards, but continuously collected by ISBSG. Therefore, we used existing dataset from ISBSG to construct the decision support model. The use of 14 GSC to discriminate the difference in IS projects may be somewhat insufficient. Future research should seek to include and collect additional inherent characteristics to enhance the proposed model performance.

This study relied on cross-validation as the internal reliability to explain the performance of the proposed approach; therefore, external reliability research can be conducted to provide more beneficial evidence. For example, we intend to apply the proposed approach to a real-world unprecedented software system (e.g., cloud software) by implementing more than one FSMM to measure the functional size of the same application, comparing the correlation among effort, LOC, and size measures, and validating the correctness of the original suggestions (i.e., whether FPA is relatively suitable for size measurement).

Future study can also expand the model by considering other selection perspectives such as personnel experience and existing tool potency. Because this study used a decision tree to construct a predictive model, we can devote future efforts to using neural networks and k-means clustering, either in combination or in alternation, to improve prediction accuracy.

# Acknowledgements

The authors would like to thank two anonymous reviewers and the editor of Journal of Information Management for their valuable feedback and suggestions. This

work was supported by the National Science Council of Taiwan, under operating grants of NSC 101-3114-C-147-001-ES.

# References

Abran, M.M., St-Pierre, D. and Desharnais, J. M.(1997), 'Adapting function points to real time software', *American Programmer*, Vol. 10, No. 11, pp. 32-43.

Abran, D.J.M., Oligny S., St-Pierre, D. and Symons, C. (1999), 'COSMIC-FFP Measurement Manual version 2.0', *Software Engineering Management Research Laboratory*, November.

Albrecht, A.J. (1979), 'Measuring application development productivity', *Proceedings of the Joint SHARE, GUIDE, and IBM Application Development Symposium*, Monterey, California, October 14-17, pp. 83-92.

Albrecht, A.J. and Gaffney, J.E. (1983), 'Software functions, source lines of code, and development effort prediction: a software science validation', *IEEE Transactions on Software Engineering*, Vol. 9, No. 6, 639-648.

Breiman, L., Driedman, J.H., Olshen, R.A. and Stone, C.J. (1984), *Classification and Regression Trees*, Belmont, California: Wadsworth, US.

Buglione, L. and Gencel, C. (2008), 'Impact of base functional component types on software functional size based effort estimation', *Proceedings of the 9th International Conference on Product-Focused Software Process Improvement*, Rome, Italy, June 23-25, pp. 75-89.

Cuadrado-Gallego, J.J., Garre, M., Rejas, R.J. and Sicilia, M.A. (2008), *Analysis of Software Functional Size Databases*, *Software Process and Product Measurement*, Springer-Verlag, Berlin, Germany.

Cuadrado-Gallego, J.J., Buglione L., Domínguez-Alda, M.J., Sevilla, M.F.de., Antonio Gutierrez de Mesa, J. and Demirörs, O. (2010), 'An experimental study on the conversion between IFPUG and COSMIC functional size measurement units, *Information and Software Technology*,Vol. 52, No. 3, 347-357.

Dumke, R. and Albrecht, A.J. (2011), *COSMIC Function Points: Theory and Advanced Practices*, Auerbach Publications, New York, US.

Ebert, D.R., Bundschuh, M. and Schmietendorf, A. (2004), *Best Practices in Software Measurement*, Springer Publication, New York, US.

Garmus, D. and Herron, D. (2001), *Function Point Analysis—Measurement Practices for Successful Software Projects*, Addison-Wesley, Boston, US.

Gencel, C. and Demirors, O. (2008), 'Functional size measurement revisited', *ACM Transactions on Software Engineering and Methodology*, Vol. 17, No. 3.

Huang, S.J. and Han, W.M. (2006), 'Selection priority of process areas based on CMMI continuous representation', *Information and Management*, Vol. 43, No. 3, pp. 297-307.

ISO/IEC, ISO/IEC 14143-1, (2007), *Information Technology – Software Measurement – Functional Size Measurement – Part 1: Definition of Concepts*, International Organization for Standardization.

ISO/IEC, ISO/IEC TR 14143-5, (2004), *Information Technology – Software Measurement – Functional Size Measurement – Part 5: Determination of Functional Domains for Use with Functional Size Measurement*, International Organization for Standardization.

ISO/IEC, ISO/IEC 14143-6, (2006), *Information Technology – Software Measurement – Functional Size Measurement – Part 6: Guide for Use of ISO/IEC 14143 Series and Related International Standards*, International Organization for Standardization.

ISO/IEC, 19761, (2011), *Software Engineering — COSMIC-FFP: A Functional Size Measurement Method*, International Organization for Standardization.

ISO/IEC, ISO/IEC 24570, (2005), *Software Engineering – NESMA Functional Size Measurement Method Version 2.1 – Definitions and Counting Guidelines for the Application of Function Point Analysis*, International Organization for Standardization.

ISO/IEC, ISO/IEC 20926, (2009), *Software and Systems Engineering – Software Measurement – IFPUG Functional Size Measurement Method*, International Organization for Standardization.

ISO/IEC, ISO/IEC 20968, (2002), *Software Engineering – Mk II Function Point Analysis – Counting Practices Manual*, International Organization for Standardization.

ISO/IEC, ISO/IEC 29881, (2010), *Information Technology – Software and Systems Engineering – FiSMA 1.1 Functional Size Measurement Method*, International Organization for Standardization.

IFPUG, (2005), *IFPUG Case Study 4, Release 2.0: Counts Function Points for A Traffic Control System with Real Time Components*, International Function Point Users Group, Westerville, OH, US.

IFPUG, (2007), *Hints to Counting Enterprise Data Warehouses. New Environments Committee White Paper*, International Function Point Users Group, Westerville, OH,

US.

IFPUG, (2008), *Hints to GUI. IFPUG New Environments Committee White Paper*, International Function Point Users Group, Westerville, OH, US.

IFPUG, (2009a), *Function Points & Counting Middleware Software Applications. IFPUG New Environments Committee White Paper*, International Function Point Users Group, Westerville, OH, US.

IFPUG, (2009b), *IFPUG Case Study 3.0: Counts Function Points during Analysis and Construction for A Human Resources Application*, International Function Point Users Group, Westerville, OH, US.

IFPUG, (2011), *Function Point Counting Practices Manual Release 4.3.1, International Function Point Users Group*, International Function Point Users Group, Westerville, OH, US.

Jones, C. (2008), *Applied Software Measurement: Global Analysis of Productivity and Quality*, McGraw-Hill, US.

Kass, G. V. (1980), 'An exploratory technique for investigating large quantities of categorical data', *Applied Statistics*, Vol. 29, No. 2, pp. 119-127.

Kitchenham, B. (1997), 'Counterpoint: The Problem with Function Points', *IEEE Software*, Vol. 14, No. 2, pp. 29-31.

Lee, T.S., Chiu, C.C., Chou, Y.C. and Lu, C.J. (2006), 'Mining the customer credit using classification and regression tree and multivariate adaptive regression splines', *Computational Statistics and Data Analysis*, Vol. 50, No. 4 ,pp. 1113-1130.

Lokan, C.J. (2000), 'An empirical analysis of function point adjustment factors', *Information and Software Technology*, Vol. 42, No. 9, pp. 649-660.

Lokan, C.J. (2005), 'Function points', *Advances in Computers*, Vol. 65, No. 7, pp. 297-347.

Michael, J.A. and Gordon, S.L. (1997), *Data Mining Technique for Marketing, Sales and Customer Support*, Wiley, New York, US.

Quinlan J.R. (1979), *Discovering Rules from Large Collections of Examples: A Case Study*, Edinburgh University Press, Edinburgh, UK.

Reifer J. (1990), 'Asset-R: a function point sizing tool for scientific and real-time systems', *Journal of Systems and Software*, Vol. 11, No. 3, pp. 159-171.

Sentas, P., Angelis, L. and Stamelos, I. (2008), 'A statistical framework for analyzing the duration of software projects', *Empirical Software Engineering*, Vol. 13, No. 2, pp. 147-184.

Total Metrics Inc. (2007), 'How to decide which method to use metrics', available at

http://www.totalmetrics.com/function-point-resources/downloads/R185_Why-use-Function-Points.pdf (accessed 10 February 2014).

Whitmire, S.A. (1995), 'An introduction to 3D function points', *Software Development*, Vol. 3, No. 4, pp. 43-53.

Yeh, Y., Cheng, C.H. and Chen, Y.W. (2011), 'A predictive model for cerebrovascular disease using data mining', *Expert Systems with Applications*, Vol. 38, No. 7, pp. 8975-8982.

Živkovič, A., Rozman, I. and Heričko, M. (2005), 'Automated software size estimation based on function points using UML models', *Information and Software Technology*, Vol. 47, No. 13, pp. 881-890.