Understanding the Influential Factors to Development Effort in Chinese Software Industry

Mei He[1], He Zhang[2], Ye Yang[1], Qing Wang[1]and Mingshu Li[1]

[1] Laboratory for Internet Software Technologies
Institute of Software, Chinese Academy of Sciences
hemei@itechs.iscas.ac.cn
[2] National ICT Australia
University of New South Wales, Australia
he.zhang@nicta.com.au

Abstract. A good understanding of the influential factors to software development effort and further precise effort estimate are undoubtedly crucial to any cost-effective and controllable software development projects. In most effort estimation researches, a large dataset is always a necessary basis of estimation modeling, model calibration and method validation. Among them, different attributes and characteristics of project data will to a large extent affect the applicable scope of particular research result. This research aims to identify the factors that significantly influence development effort, and to investigate how the influence works in Chinese software industry. In this study, six factors and their relationships to development effort are analyzed, prioritized and discussed based upon the dataset recording 999 projects from 140 software organizations in China. In terms of our analysis and findings, some suggestions for effort estimation and control are extracted to assist software practitioners in coping with various types of software projects.

1 Introduction

Software development is considered to be a human-intensive process, and its main cost is largely determined by the effort taken in it. Thereby, a good understanding of the influential factors to software development effort and further precise effort estimate are undoubtedly crucial to any cost-effective and controllable software development projects. Moreover, as software process improvement has been widely accepted and adopted in software industry, the organizations need more reliable and effective methods in predicting and quantitatively controlling project cost in order to improve their process management.

In the research of software development effort estimation, various techniques, like expert judgment, algorithm-based models, analogy and machine learning, have been proposed and applied. However, it is difficult to get consensus on which model or method is better than the others [1], [2]. Furthermore, no matter what technique is used to estimate software development effort, it is always one of the key concerns of software practitioners: "*What factors do influence software development effort and how they influence?*" [1], [3].

M. Ali Babar, M. Vierimaa, and M. Oivo (Eds.): PROFES 2010, LNCS 6156, pp. 306–320, 2010.
© Springer-Verlag Berlin Heidelberg 2010

In most effort estimation researches, a large dataset is always a necessary basis of estimation modeling, model calibration and method validation. Among them, different attributes and characteristics of project data will to a large extent affect the applicable scope of particular research result. For example, the accuracy of some model might be relatively high, but the difficulty in obtaining model inputs would be the holdback to its wide application.

This paper aims to revisit the influence of the typical factors to software development effort, but in the context of Chinese software industry. The large-scale dataset used in this research stores the project data of 999 projects from 140 software organizations throughout China. It can be used to investigate the status quo of software development in China and to explore what factors affect development effort in these projects. Especially in this study, we attempt to identify the factors that significantly influence software development effort, and to investigate how they influence. Some suggestions for effort/cost estimation and control can be extracted to assist software practitioners in coping with different types of software project.

This paper is structured as follows. Section 2 briefly introduces the dataset used in this study, enumerates the typical influential factors to effort and the associated research questions. Next, modeling and analysis procedure and results are described in Section 3. Section 4 discusses the significance of the results for answering the research questions with comments. Our conclusions are drawn in Section 5.

2 Research Questions and Related Work

This section introduces the dataset used in this study, discusses the possible influential factors with the related work, and proposes the corresponding research questions.

2.1 CSBSG Dataset

The dataset used in this paper is from China Software Benchmarking Standard Group (CSBSG). The CSBSG was established in 2006, and it aims to encourage and establish domestic benchmarking standards for system and software process improvement in Chinese software industry. The database was founded and is being maintained by a number of Chinese organizations within China Software Process Improvement Network (CSPIN). The dataset used in our study is the latest version of CSBSG database, recording 999 software project data from 140 organizations located in 15 regions/provinces across China.

Although each project has many metrics recorded, this study only introduces those typical factors that possibly influence development effort and were relatively well recorded in the dataset. In fact, many of those factors have been discussed by other researchers, but there exist significantly different conclusions for each of them. For example, no agreement has been reached yet on whether new development costs more effort than enhancement. In this study, such influences with contradictious discussion are examined based on the analysis of our dataset.

2.2 Project Size

Obviously, Project Size (PS) is an essential parameter for effort estimation that the majority of mainstream and classical effort estimation models all have used it as a key estimator. Particularly during the development of algorithm-based effort estimation models, a number of researchers have chosen the very similar formula in general like Effort=A+B*(Size)^{C}, which explicates the close relationship between *size* and *effort*. For example, COCOMO, a well-known and widely adopted series of models for cost estimation, has continued to use the same form (as shown in Equation 1) for years.

$$PM = A \times \left(\sum Size\right)^{\sum_B} \times \prod (EM) \tag{1}$$

Hereby, in terms of our dataset, the first research question emerges as:

RQ1: **How does project size influence software development effort?**

2.3 Team Size

Team Size (TS), in previous researches, has been identified as a variable influencing software productivity or effort [4], [5], [3], [6], and most of them agreed that increasing team size will reduce productivity or increase effort. In [3], [6], both the *average team size* and *peak team size* had been observed and recorded. In this paper, TS is referred to the maximum number of members involved in the entire project life-cycle, as it is easier to measure than *average team size* over the project.

RQ2: Will a larger team size cause extra expense in effort?

2.4 Duration

Duration (DUR) is measured with calendar days in this study, i.e. the number of days from the project commencement date to the end date (holidays inclusive). Some previous researches discussed the relation between productivity and duration. In [4], the authors found a seeming good regression model while adding duration, lines of code and team size together as independent variables, but they thought that is roughly the definition of lines-of-code (LOC) productivity and thus added nothing to their knowledge. In terms of our project data, the recorded DUR is much longer than the expected schedule by experience; whereas, some projects even spent less than 3 man-hours a day. One possible explanation is that project members took part in multiple projects concurrently, and it could be another case that the schedule pressure was not much. Then another practical question comes out:

RQ3: Will deadline extension cause additional waste of development effort?

2.5 Development Type

Development Type (DT) indicates whether a software project is new development, re-development, or enhancement. Some researchers considered new development costs more effort than enhancement [3], and explained that while new development starts everything from scratch, software enhancement simply adds, changes, or deletes software functionality of legacy systems to adapt to changes in business requirements

[7]. On the other hand, some found no significant difference between them [8]. Some new development projects in ISBSG database also show higher productivity [9]. There is no consensus so far, and here we intend to revisit the influence in our dataset.

RQ4: Does new development really cost more effort than enhancement?

2.6 Business Area

Business Area (BA) denotes the types of business within the organization or industry that the project/product will support. CSBSG dataset covers 13 business areas, i.e. Telecom, Transport, Finance, Retail & Inventory, Media, Energy, Generic, Health Care, Public Administration, Manufacturing, Construction, Education and Society Service. Nevertheless, the last three areas are not included in the later analysis due to the relatively small number of projects recoded in the dataset (less than 10).

BA has been identified as one of the most significant factors influencing productivity for times [4], [10], [8], [9]. However, the most productive area is not consistent among the results by different researchers. For example, banking and assurance, which are classified as "finance" in CSBSG, are the most productive areas reported in [10] but the least in [9]. In practice, many factors, such as personnel application experience, software complexity, requirement volatility etc., would affect the software development for different areas [10], [9]. The state of software development for different business areas in China needs to be further studied.

RQ5: Which business area is relatively more cost-effective?

2.7 Programming Language

The primary Programming Languages (PLs) in software project considered into this research are the ones with more than 10 observations in the dataset: ASP, C, C#, C++, COBOL, Java and VB.

Some previous researches removed the language effect either by merely considering programs written in the same language or by converting all data into one language using conversion factors. Nonetheless, a number of researchers have found that productivity varies with the level of the language [4]. As the language level increases, fewer lines of code are needed to deliver the same functionality. In [3], languages were classified by '*generation'*, and the analysis was seldom on the basis of specific language. In terms of CSBSG dataset, most frequently applied languages are the third generation languages (3GLs), and accordingly the analysis of language influence on productivity is based on the specific languages in this study.

RQ6: Does programming language really matter in predicting effort?

3 Analysis Procedure and Result

3.1 Data Validation and Preliminary Analysis

Project Size is recorded as "Size Total" in CSBSG dataset. For all the 999 projects, 998 ones have their size measured by LOC, and only one exception of Project 867[1] is

[1] Each project was assigned an exclusive ID number from 1 to 999.

recorded in Function Points (FPs). Another 3 projects from the same organization (as Project 867) use the same primary language - Java, and have their sizes recorded in both FP and LOC. All ratios of LOCs per FP are 53, which is consistent with the transformation ratio reported in SPR documentation [11]. In that case, transforming the size of Project 867 into LOC metrics can be reasonable.

The maximum Team Size were not given in these four projects, 3 values are filled up by comparing the phased team size records and selecting the maximum value, while the remainder has no phased team size recorded and is therefore excluded.

In effort modeling, Actual Total Work Effort in man-hours is used as the dependent variable, and the factors are intended to add as independent variables in the model. The modeling procedure and final result may reveal the possible relationships between factors and effort based upon CSBSG dataset.

3.2 Model Development

First, Table 1 lists the modeling variables, scales and descriptions for reference.

Table 1. Summary of the variables considered in the modeling procedure

Variable	Scale	Descriptions
ln_effort	Ratio	Log-transformed Summary Work Effort
ln_size	Ratio	Log-transformed Total lines of code
ln_teamsize	Ratio	Log-transformed Maximum size of the development team
ln_dur	Ratio	Log-transformed Total working days from Start to End Date
DevType	Nominal	Development Type
BusiArea	Nominal	Business area within the organization/industry that the project/application will be supporting
Language	Nominal	Primary programming language

Prior to model development, Effort, Project Size, Team Size and Duration are all taken natural log transformation to redress the skewness for these variables. Fig. 1 is the histograms of log transformed Effort, Project Size, Team Size and Duration, which show normal distribution well.

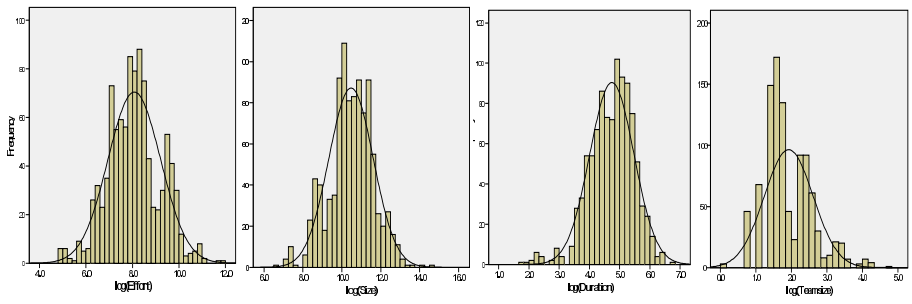
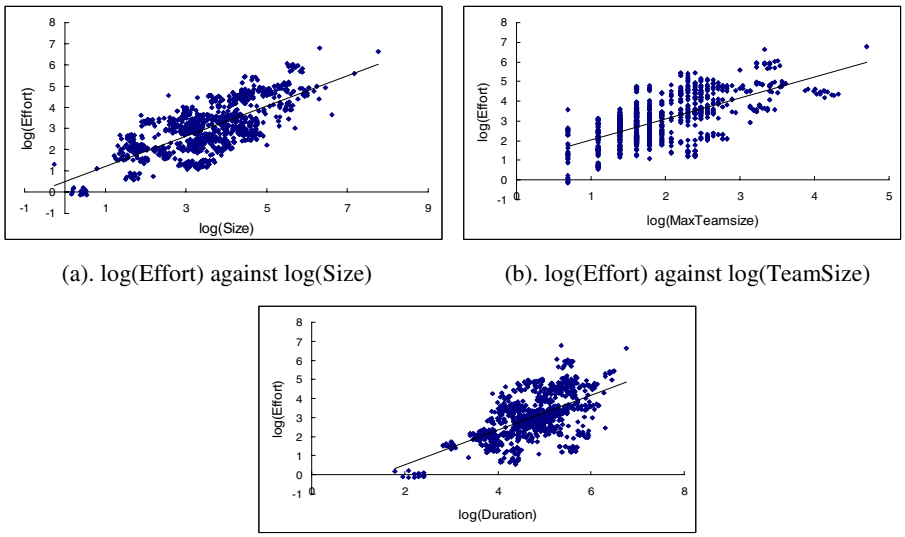


Fig. 1. Distribution of log-transformed numerical variables

After that, the potential relationships between Effort and the factors (Project Size, Team Size and Duration) after log transformation are explored. The three graphs below (see Fig. 2) indicate that linear model can be used to approximate their relationships with effort. A multiple linear regression can be applied to develop our model. The linear model is supposed to be in form of:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_k X_k + \mu \quad (2)$$



(a). log(Effort) against log(Size)

(b). log(Effort) against log(TeamSize)

(c). log(Effort) against log(Duration)

Fig. 2. Scatter plots of effort against factors

Furthermore, the correlation analysis is employed to check whether the problem of multi-colliearity (strong correlations between independent variables) exists in the data. As recommended by Maxwell [12], Spearman's correlation analysis is done to check the numerical variables' independence; ANOVA (analysis of variance) is run to check the independence between the categorical variables and chi-square test for the relationship between the categorical and numerical variables. The result confirms that multi-colliearity within this data is not a problem.

In the modeling procedure, three numerical variables, Project Size, Team Size and Duration, passed the check and can be added into one model; but there exist some correlations between any two of the categorical variables, i.e. DevType, BusiArea and Language.

In addition, to explore the problem of missing values, the metrics with missing data are Duration (22), Development Type (13), and Team Size (1). According to the rule of thumb, a minimum sample size of 50+8k for multiple regression analysis is suggested [13]. The valid sample size here is acceptable.

Once the above issues are solved, the regression model can be developed by following the two steps recommended in [12]. At the same time, we also use the statistical tool (Stata [19]) to assist our analysis.

- **Step 1:** Stepwise regression analysis with numerical variables

Performing stepwise regression procedure helps to determine the relative importance of each numerical independent variable's relationship to the dependent variable. It only takes the variables available for nearly every project into consideration. In our dataset, missing value for the numerical variables, i.e. ln(size) (abbreviated as *lsize*), ln(duration) (as *ldur*), ln(TeamSize) (as *lteam*) in statistical analysis is very little as discussed above, and no problem to apply this procedure.

.sw regress In_effort In_size In_dur In_teamsize, pr(.05)
Begin with full model
P<0.0500 for all trems in model

Srouce	ss	df	MS
Model	900.266105	3	300.088702
Residual	371.92449	988	.376444787
Total	1272.19355	991	1.28374728

Number of obs = 992 F(3, 988) = 797.17 Prob > F = 0.0000
R-squared = 0.7076 Adj R-suqared = 0.7068 Root MSE = .61355

In effort	Coef.	Std. Err.	t	p > | t |	[95% conf.	Interval]
In_size	.2986532	.0238871	12.50	0.000	.2517778	.3455286
In_dur	.535817	.0323085	16.58	0.000	.4724159	.5992181
In_teamsize	.6862529	.0338465	20.28	0.000	.6198336	.7526723
cons	1.08979	.1897075	5.74	0.000	.7175139	1.462066

Fig. 3. Results for forward stepwise regression

The result of running a forward stepwise regression procedure is shown in Fig. 3 (a screen shot from Stata's running result). Given the criteria that if Prob>F is a number less than or equal to 0.05, the model can be accepted. In this case, the value of Prod>F is small enough, which means this model is significant. Thereafter, the result of running a backward stepwise regression procedure is also validated as a significant linear model.

- **Step 2:** Building the multi-variable model with "stepwise ANOVA" [12]

From this step, the best *one-variable* model, best *two-variable* model, best *three-variable* model and so on, are obtained one by one.

At first, to determine which variable (*lsize*, *ldur*, *lteam*, or *devtype*) explains the most variation in *leffort*, regression procedures are run for numerical variables, and ANOVA procedures for the categorical variables. As shown in Table 2, *lsize* explains the most variation in *leffort*. The result confirms the findings from many previous studies which make project size as the most important key variable for cost or effort estimation [14], [2], [1].

Then, *lsize* is added to the model in order to find the best two-variable model. As shown in Table 2, *Devtype* is then added to form the best two-variable model. Such procedure is repeated until there is no possible further improvement in the obtained model. All the outputs are recorded in Table 2.

Table 2. Statistical Output Summary Sheet

Variables	Num Obs	Effect	Adj R^2
1-variable models			
*ln_size	999	+	0.5180
ln_duration	993	+	0.3847
ln_teamsize	998	+	0.4454
DevType			0.0419
Language			0.0867
BusiArea			0.2415
2-variable models with lsize			
ln_duration	993	+	0.5860
ln_teamsize	998	+	0.6256
*DevType			0.6267
Language			0.5779
BusiArea			0.6041
3-variable models with lsize, DevType			
*ln_duration			0.6820
ln_teamsize			0.6772
Language			0.6725
BusiArea			0.6720
4-variable models with lsize, DevType, ldur			
*ln_teamsize			0.7465
Language			0.7330
BusiArea			0.7429
5-variable models with lsize, DevType, ldur, lteam			
Language			0.7778
*BusiArea			0.7854
6-variable models with lsize, DevType, ldur, lteam, BusiArea			
Language			0.8088

Finally, the best model is a six-variable model: *leffort* as a function of all the variables listed in Table 3. To be noticed that the default Development Type is enhancement, default Programming Language is 'Other', and the default development Business Area is manufacturing.

According to coefficients in Table 3, the model equation is extracted as:

$$\begin{aligned} &\ln(effort) = 0.38\times\ln(size) + 0.5\times\ln(teamsize) \\ &+0.55\times\ln(duration) + \alpha_i \times I(DevType_i) \\ &+\beta_j \times I(BusiArea_j) + \chi_k \times I(Language_k) + 0.31 \end{aligned} \tag{3}$$

where the function ***I*** is the indicator function with binary values of 1 or 0 ('1' means the project belongs to such type or uses such language, otherwise '0'); and the coefficients α_i, β_j and χ_k are corresponding to the values in Table 3. The default coefficients for the default types (that is enhancement, 'Other' language, and manufacturing business area) are all zero.

The explanatory power of the fitted model is high at R^2 = 80.9%, which indicates that 80.9% of the variance in the dependent variable can be explained by this model.

As shown in Fig. 4, the predicted values and observed values conform well to each other.

However, we have to emphasize again, the motive of this paper is not to obtain another prediction model, but to revisit and validate the influencing relationship between development effort and these factors in the context of Chinese software industry. In that case, further investigation on the prediction accuracy and comparison with other effort estimation models are not taken into account in this paper.

Table 3. List of fitted coefficient in the final 6-variable model

Regression terms	Coef.	Std. Err.	*p*-value
ln_size	0.38	0.03	0.000
ln_teamsize	0.50	0.04	0.000
ln_dur	0.55	0.03	0.000
Re-Dev	-0.16	0.10	0.092
New Dev	-0.46	0.05	0.000
Telecom	0.32	0.09	0.000
Transport	-0.13	0.16	0.428
Finance	0.48	0.09	0.000
Retail	0.81	0.09	0.000
Media	0.87	0.18	0.000
Energy	0.120	0.09	0.183
Other	0.28	0.10	0.006
Generic	0.17	0.09	0.059
Health care	0.38	0.13	0.004
Public Admin.	0.123	0.08	0.113
Asp	-0.29	0.17	0.086
C	0.23	0.12	0.058
C#	-0.06	0.11	0.554
C++	0.34	0.11	0.002
Cobol	-0.24	0.15	0.115
Java	0.30	0.11	0.004
VB	0.65	0.14	0.000
_cons	0.31	0.26	0.228

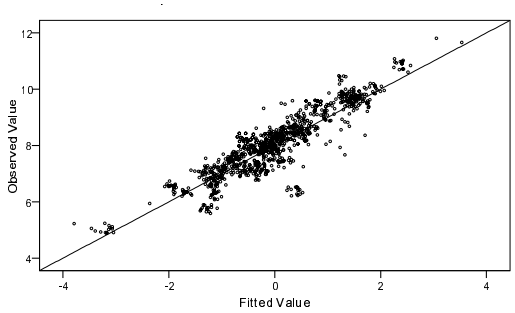


Fig. 4. Scatter plot of observed values versus fitted values

3.3 Model Validation

The model's underlying assumptions need to be checked before the final model obtained through the above steps.

- Assumption 1: In a well-fitted model, there should be no pattern to the errors (residuals) plotted against the fitted values.
- Assumption 2: The errors in the model should be randomly and normally distributed with mean zero.

In our model, "Fitted Value" here refers to the *leffort* predicted, and Fig. 5, where the residual versus fitted value graph is shown, indicates no obvious pattern. In addition, Fig. 6 shows the distribution of residuals which is normality with mean zero. Therefore, the assumption of normality of the residuals can be checked and confirmed.

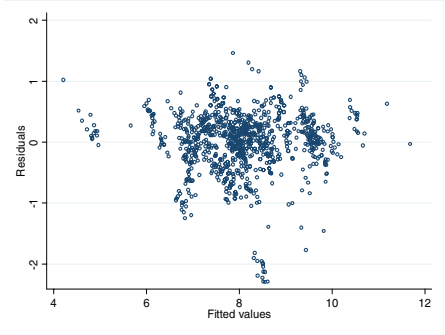


Fig. 5. Diagnostic plot of the residuals versus the fitted values

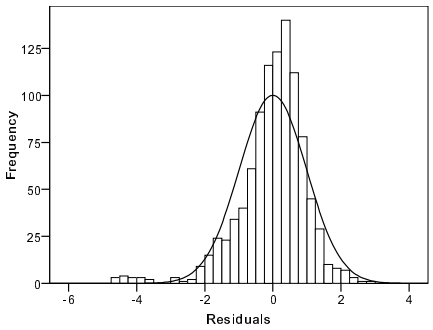


Fig. 6. Histogram of the residuals to check its normal assumption

4 Discussions

As shown in Table 3, on the basis of p-value < 5%, the final sets of factors that are significant to software development effort can be identified: Project Size, Duration, (maximum) Team Size, Development Type, (primary) Programming Language and Business Area.

Project Size (RQ1): The regression coefficient of effort on Project Size after log transformation is 0.38. It illustrates that Project Size is positively related to effort. While productivity is defined as size over effort, it also shows that productivity will increase with increasing Project Size. The result confirms the finding in [15], which compared the median productivity of different project size groups. Interestingly, the phenomenon of Economies of Scale for our dataset is consistent to some others' research [3], [6], but opposite to [14]. For the phenomenon of Economies of Scale, Agrawal et al. [6] explained that is due to the high maturity (CMM 5 level) for organizations in their study. However, this might be not the case in terms of our data due to the lack of supporting information. Another possible explanation is that small-sized projects came from low

productivity organizations and the large ones from high productivity ones, but these all need to be further investigated with more evidence in the future data collection and analysis. In addition, while adding size alone, the explanatory power of the fitted model is high at R2 = 51.8%, which indicates that project size is indeed an intrinsic driver of software development effort. This result is also in agreement with many classic effort estimation researches which identify software Project Size as a fundamental factor in dealing with software development effort or cost [14], [2], [16].

Team Size (RQ2): The regression coefficient of effort on Team Size after log transformation is 0.50, which indicates more effort need to be spent for larger team size while other attributes' values do not change. This result is consistent with the finding in [15]. It is quite frequent that some managers are used to adding new personnel for a challenging project. However, adding personnel is not always a wise decision since organizations have to pay more attention and effort to maintain their process control, personnel coordination and resource harmony for an increased team size.

Duration (RQ3): The regression coefficient of effort on project Duration after log transformation is 0.55, the positive value implies that increasing project duration is very likely to lead to a decrease in productivity. In other words, to implement the same size of software, increasing project calendar time will increase total effort. Sometimes, due to the pressure from concurrently developed multiple projects, development teams have to decrease their effort on every single project and postpone their schedule. The result here reminds managers to balance the additional effort caused by schedule slack.

Development Type (RQ4): By modeling analysis, Development Type is confirmed to be another significant factor to influence effort. Table 3 shows that the regression coefficients of re-development and new development are -0.16 and -0.46 respectively, and these values are relative to the coefficient 0 of enhancement as the default development type. This means that given the other attributes with the unchanged values, the enhancement projects may consume the most effort, while re-development may need less effort than enhancement, and new development may consume even less than re-development. In other words, new development projects in the CSBSG dataset show the highest productivity than the other two types, which also confirms the finding in [15]. In contrast with the findings in some other research [3], [7], the possible reasons for the low productivity in enhancement are explored. If the manager often changes the development team or key personnel, it might add the effort in assimilation process. At the same time, in new development, rush to get high productivity with the lack of disciplined documentation may also cause many problems for future maintenance or enhancement work. All of these give project managers a noticeable reminder.

Business Area (RQ5): The diversity of Business Area within the organization or industry that the project/product will support is also confirmed to significantly influence software development effort. With reference to the default manufacturing area (Coef. 0), all the business areas can be ranged in descending order of the number of effort needed: Media (Coef. 0.87), Retail & Inventory (Coef. 0.81), Finance (Coef. 0.48), Health Care (Coef. 0.38), Telecom (Coef. 0.32), Generic (Coef. 0.17), Public

Admin (Coef. 0.123), Energy (Coef. 0.120), Manufacturing (Coef. 0) and Transport (Coef. -0.13). By fixing the other attributes' values, projects in such business areas like Media, Finance and Retail & Inventory may cost more effort, while other areas like Public Admin, Energy and Manufacturing may cost less, in other words, they are more productive. Compared to productivity ascending order shown in [15], the consistency is that software development in Energy, Manufacturing and Public Admin was more productive, and the Finance and Retail & Inventory areas were less productive. There is an inconsistency for Telecom area, based on the modeling analysis, Telecom was not as inefficient as described in [15].

With interests in this inconsistency, 171 projects from Telecom area are further examined. From the aspect of Development Type, only 28% projects are new development; from the Project Size, 87% of them are smaller than 64KLOC, and 57% are even smaller than 16KLOC. In addition, as shown in Fig. 7, C++ and Java are two languages dominated the Telecom projects, while they show relative low productivity as discussed later, that could be a possible reason for the low productivity in this Telecom subset from CSBSG.

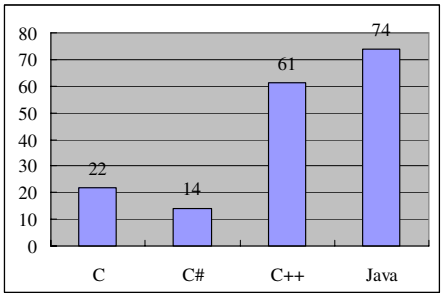


Fig. 7. Language application state for Telecom projects

Because of the diversity of Business Area, software development is affected by multiple aspects. As to Finance area, there exists some other research reporting its low productivity [17]. Since financial software requires real-time, excessive data exchange, vast data processing, high level security and other complex technologies, the productivity is easier to decrease than other business sectors. Meanwhile, due to considerations on confidential information, some banks or investment companies insist implementing internal software development regardless the low productivity.

On the other hand, for the cost-effective areas, such as Public Admin, Energy and Manufacturing, one possible explanation might be that most of the projects in these areas have comparatively less complexity and relatively stable requirements. Also, formal public bidding institutionalization in Chinese government contributes to guarantee for the quality and efficiency of the entrusted software development companies in the recent years [18]. Generally speaking, the market competition, requirement of functionality, evolution and complexity of techniques, integration extent of hardware, and other issues influence the software development in each business area.

Programming Language (RQ6): Programming Language is the last but not least influencing factor to development effort. By comparing the coefficients of each type of language, projects using ASP costed the least effort, and then followed by Cobol, C#, C, Java, C++, and Visual Basic. In contrast to the comparison result in [15], there exist inconsistencies for Cobol and VB. Among 24 projects using Cobol, 23 of them are enhancement and from Finance or Retail & Inventory areas with relatively complex requirements. This might result in Cobol's low productivity level when compared in the whole dataset. On the contrary, for the 48 projects using VB, 87.5% are for Manufacturing area whose system functions were relatively stable, and they were all new development. Hence, the relatively high productivity could be explained by the factors other than language alone. In previous studies, specific language was seldom used to discuss the influence of language on software development effort; instead, language generations, called 2GL, 3GL, 4GL etc., have been considered by some researchers [9], [3]. However, almost all the languages presented in CSBSG are 3GL, and it is difficult to compare our result with the others that classified languages by their generations.

5 Conclusions and Future Work

A better understanding the factors influencing development effort/cost can enable software project practitioners to achieve more reasonable and realistic resource estimation and allocation solutions. As a matter of fact, many researchers tried to contribute in this direction. However, due to the lack of support of relatively large datasets, in-depth studies on the basis of real projects in software industry, particularly in China, were limited. This study analyzes the data of 999 projects from 140 software organizations in China to revisit the factors that significantly influence software development effort, and to figure out how they influence in this context.

As a result, the set of factors that are significant to Chinese software development effort are prioritized: Project Size, Duration, (maximum) Team Size, Development Type, (primary) Programming Language, and Business Area. In terms of the analysis results, we can confirm some findings from the previous related researches, and also conclude the answers to the research questions (Section 2), some of which seem to be counter-intuitive somehow.

1) The effort increased while software (project) size increased, and this dataset reveals the phenomenon of Economies of Scale.
2) More effort were needed for larger team size while other factors maintained the same.
3) Extending the deadline of projects might cause additional development effort.
4) Given the other attributes with the same values, enhancement projects consumed the most effort, while re-development required less effort than enhancement, and new development took even less than re-development.
5) Without changing the other attributes' values, projects in the business areas like Media, Finance and Retail & Inventory costed more effort than in the other sectors like Public Admin, Energy and Manufacturing, where projects were observed more productive.

6) Projects using ASP costed the least effort, which was followed by Cobol, C#, C, Java, C++, and Visual Basic in ascending order.

However, as the limitation of some missing or ignored information in the current dataset results in a difficulty in further examining the exact reasons, we only present some preliminary reason analysis at the current stage. These analyses can provide the project managers some empirical suggestions in real word project management.

For the future work, we plan to add more factors while modeling cost estimation for some type of projects, for example, focusing on one specific business area. Moreover, to construct a cost prediction model for some type of projects is also an important subject in the future research.

Acknowledgments

This work is supported by the National Natural Science Foundation of China under Grant Nos. 90718042 and 60873072; the National Hi-Tech R&D Plan of China under Grant No. 2007AA010303; the National Basic Research Program (973 program) under Grant No. 2007CB310802.

NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Centre of Excellence program. This work was also supported, in part, by Science Foundation Ireland grant 03/CE2/I303 1 to Lero - the Irish Software Engineering Research Centre (www.lero.ie).

References

1. Boehm, B.W., Abts, C., Chulani, S.: Software Development Cost Estimation Approaches - A Survey. Annals of Software Engineering 10(1-4), 177–205 (2000)
2. Li, M., He, M., Yang, D., Shu, F., Wang, Q.: Software Cost Estimation Method and Application. Journal of Software 18(10), 775–795 (2007)
3. Jiang, Z., Naudé, P.: An examination of the factors influencing software development effort. International Journal of Computer, Information, and Systems Sciences, and Engineering 1(3), 182–191 (2007)
4. Maxwell, K.D., Wassenhove, L.V., Dutta, S.: Software Development Productivity of European Space, Military, and Industrial Applications. IEEE Transactions on Software Engineering 22(10), 706–718 (1996)
5. Jiang, Z., Naudé, P., Comstock, C.: An investigation on the variation of software development productivity. International Journal of Computer, Information, and Systems Sciences, and Engineering 1(2), 72–81 (2007)
6. Agrawal, M., Chari, K.: Software development effort, Quality and Cycle Time: A Study of CMM Level 5 Projects. IEEE Transactions on Software Engineering 33(3), 145–156 (2007)
7. Kemerer, C.F., Slaughter, S.: Determinants of software maintenance profiles: an empirical investigation. Journal of Software Maintenance 9, 235–251 (1997)
8. Premraj, R., Shepperd, M., Kitchenham, B.A., Forselius, P.: An Empirical Analysis of Software Productivity over Time. In: IEEE METRICS 2005, p. 37 (2005)
9. ISBSG Benchmark Release 8, `http://www.isbsg.org`

10. Premraj, R., Twala, B., Mair, C., Forselius, P.: Productivity of Software Projects by Business Sector: An Empirical Analysis of Trends. In: 10th IEEE International Software Metrics Symposium (Late Break-in Papers) (September 2004)
11. SPR programming languages table (2003), http://www.spr.com/
12. Maxwell, K.D.: Applied statistics for software managers. Prentice Hall, New Jersey (2002)
13. Green, S.A.: How many subjects does it take to do a multiple regression analysis? Multivariate Behavioral Research 26, 499–510 (1991)
14. Boehm, B.W.: Software Engineering Economics. Prentice-Hall, Englewood Cliffs (1981)
15. He, M., Li, M., Wang, Q., Yang, Y., Ye, K.: An Investigation of Software Development Productivity in China. In: Wang, Q., Pfahl, D., Raffo, D.M. (eds.) ICSP 2008. LNCS, vol. 5007, pp. 381–394. Springer, Heidelberg (2008)
16. Pfleeger, S.L.: Software Cost Estimation and Sizing Methods: Issues, and Guidelines. Rand Corp. (2005)
17. Maxwell, K.D., Forselius, P.: Benchmarking Software Development Productivity. IEEE Software, 80–88 (January/February 2000)
18. He, M., Yang, Y., Wang, Q., Li, M.: Cost Estimation and Analysis for Government Contract Pricing in China. In: Wang, Q., Pfahl, D., Raffo, D.M. (eds.) ICSP 2007. LNCS, vol. 4470, pp. 134–146. Springer, Heidelberg (2007)
19. http://www.stata.com