Article

Cost estimation of feature oriented software development: Statistical approach

Jawad Khan[1], **Faisal Bhadur**[1], **Muhammad Naeem**[2], **Muhammad Javed**[1], **Naveed Jan**[3]

[1]Department of Information Technology, Hazara University, Mansehra, Pakistan

[2]Department of Information Technology, Abbottabad University of Science and Technology, Havelian, Pakistan

[3]Phine Hills Institute of Business Studies, Abbottabad, Pakistan

E-mail: jawadzjadoon@gmail.com, msosfaisal@gmail.com, naeem@aust.edu.pk, mjavedgohar@hotmail.com

Received 26 May 2016; *Accepted 5 July 2016*; *Published 1 December 2016*

Abstract

Software cost estimation is the important part of software development and used to predict the effort which is mandatory part for development of software system. Feature Model is a set of products that represents set of feature in feature model. Software cost estimation is about predicting amount of cost in software development cycle. SCE give analysis both to user as well customer regarding to budging planning. In this paper we have represent a statistical approach for finding cost estimation of feature oriented software development. We have used CPM approach from which we will find commonality as well cost of each feature which will be deploy in feature oriented software development.

Keywords Feature Model; software cost estimation; capability profile matrix; feature oriented domain analysis.

Selforganizology
ISSN 2410-0080
URL: http://www.iaees.org/publications/journals/selforganizology/online-version.asp
RSS: http://www.iaees.org/publications/journals/selforganizology/rss.xml
E-mail: selforganizology@iaees.org
Editor-in-Chief: WenJun Zhang
Publisher: International Academy of Ecology and Environmental Sciences

1 Introduction

Producing things in large amount require standardized processes, especially for the similar products. Companies are organizing their production in large amount of products (Benavides et al., 2010; Zhao and Zhang, 2013; Zhang, 2016). To reuse existing systems in a systematic way, service oriented systems resemble supply chain where products manufactured from supplied parts. Same case is for complex service-oriented systems, which needs third party services (Thomas, 2008). For example, car producer offer variation on a model with variable engines, gearboxes, audio and entertainment systems. Similarly, increasing number of software systems with almost similar requirements guide us to Software Product Line (SPL) (Böckle and Linden, 2005). SPL engineering helps in the development within application domain by considering their commonalities and variability. Features represent the aspects of these software (Kang et al., 1990). To get a valid combination of these features we use feature model which depicts the relationships of these features and constraints on them (Batory, 2005). Usually feature models are tree like structures describing successive

refinement of the variability in a product-line. Feature models were proposed back in 1990 as feature oriented domain analysis (FODA) (Kang et al., 1990).

Software product line has benefits at multiple levels. The main constraints involve in SPL is reusability. While using this constraint we can develop feature oriented software. Feature oriented software development mainly focus on three aspects, i.e., structure, reuse and variation. Developers use the concept of a feature to structure design and code of a software system, features are the primary units of reuse in FOSD, and the variants of a software system vary in the features they provide (Apel and Kastner, 2009). FOSD provide us a way toward customization. Customer demands more and more in-order to customized software product (Klaus et al, 2005; Shannon et al., 2014). Many companies are now moving toward this approach because of its great benefits besides mass customization, such as reduced effort for new product, save time, decreased maintenance effort, increased quality and improved cost estimations (Pohl et al., 2005).

Software cost estimation is about predicting amount of cost in software development cycle. SCE give analysis both to user as well customer regarding to budging planning. Cost Estimation is an important factor for both developer and user. Different techniques are being used to estimate the software cost. Each technique has its own merit and demerit. If a customer knows the cost of selected feature at design phase, it will give great importance to gain satisfaction. For example there exists online car configurator. User can select features of its car on their own finally he gets cost of selected feature. Here our focus is to determined development cost of any software that is based on feature oriented. For finding the development cost many companies may get benefit like user satisfaction along with terms they know the cost of both for user which we can say in terms of money and development as well. Prediction the development cost at design phase of any software may lead to get satisfaction for any organization. Many companies are moving toward customization to get benefit of this term in feature oriented we can predict the cost of selected feature by using statistical technique along with COCOMO II used formulas.

COCOMO is well-studied and accepted cost and effort estimation model and was proposed by Boehm (R.K.D. Black et al., 1970).COCOMO II model is composed of three variants Application composition model, Early design model, and Post architecture model. This is an extension of intermediate COCOMO model (Y. Singh., 2007). In our research we have used different equations of COCOMO II model in order to predict the cost estimation of features. We have also introduced a statistical analysis approach that will be helpful to predict cost estimation of feature oriented software at design phase. Statistical analysis is the new emerging term that is used for collecting, analyzing and making inference from data. Statistical methods and analyses are often used to communicate research findings and to support hypotheses and give credibility to research methodology and conclusions. It is important for researchers and also consumers of research to understand statistics so that they can be informed, evaluate the credibility and usefulness of information, and make appropriate decisions.

The paper is structured as follows: Section 2 review the back ground related to feature model, CPM, approach for software effort estimation, cost, COCOMO II model, function point and finally describe related work. Section 3 illustrates the case study. Section 4 shows the result and section 5 conclude the conclusion.

2 Background and Related Work

In this section, we give an overview of FMs. A FMs is prominent characteristic of product (Kang et al., 1990), firstly coined by Kang in the form of technical report on FODA in 1990. FMs is now one of important techniques use for capturing and managing commonalities and variability in product lines throughout all stages of product lines engineering (Czarnecki et al., 2005). In FMs features are arranged in tree-like structure with additional cross-tree constraints (Benavides et al., 2010). Fig. 1 depicts a simplified feature model inspired by

the mobile phone industry (Benavides et al., 2010). There are different types of features allowed in features models and are discuss here

Mandatory: the feature must be included in the description of its parent feature (Benavides et al., 2010), For instance, every mobile phone system in our example must provide support for calls.
Optional: the feature may or may not be included in its parent description given the situation (Benavides et al., 2010), in the example, software for mobile phones may optionally include support for GPS.
Alternative feature group: one and only one of the features from the feature group can be included in the parent description (Benavides et al., 2010) in the example, mobile phones may include support for a basic, color or high resolution screen but only one of them.
Or feature group: one or more features from the feature group can be included in the description of the parent feature (Benavides et al., 2010). In Fig. 1, whenever Media is selected, Camera, MP3 or both can be selected.
FMs also contain the cross-tree relationship that specifies incompatibility between features. These are typically in the form:
Requires: In case feature A requires a feature B, the inclusion of A in a product implies the inclusion of B in such product (Benavides et al., 2010), represented by empty circle at the end of edge. Mobile phones including a camera must include support for a high resolution screen.
Excludes: In case feature A excludes a feature B, both features cannot be part of the same product (Benavides et al., 2010), represented by filled arc. GPS and basic screen are incompatible features.

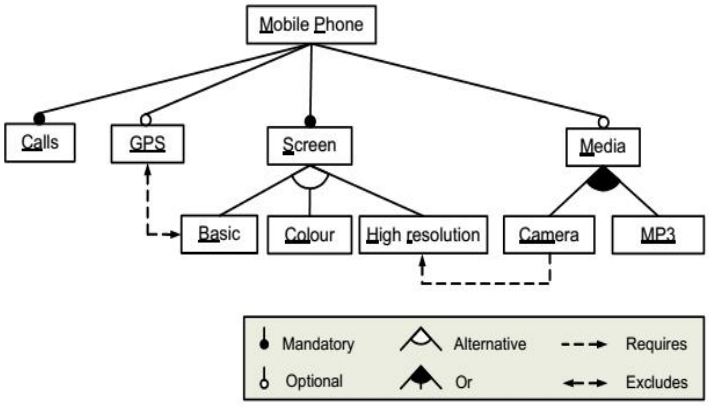

Fig. 1 A feature model of a mobile phone (Benavides et al., 2010).

The role of software becomes increasingly critical for business as well as for human lives (Jones, 1996). Problem caused in software product is because of late or our budgeting. Developing reliable software on time and within budget represents a difficult endeavor for many organizations (Shah, 2008). The most crucial task and still open challenge is estimation; accuracy of any project depends on many factors like business plan, recourses required, resources used, customer expectations, impact of changes and replanning (Leung and Fan, 2002).

The term Software cost estimation is use to predict the amount of effort and development time required to build the software system. It is one of the important factors that is use to in software industries to effectively manage their software development process. Different approaches are carried out for cost estimation techniques and listed here as Model Based -SLIM, checkpoint, SEER, COCOMO. Expertise Based-Delphi,

Rule-Based., Dynamics-Based-Abdel-Hamid Madnick, Learning Oriented Neural, Case based. Regression Based-OLS, Robust. Composite Bayesian-COCOCMO-II.

2.1 CPM

The competitive profile matrix (CPM) isolates firm's major competitors and its particular strengths and weaknesses in relation to a sample firm's strategic position (Da˘gdeviren and Yüksel., 2008). According to (Zimmerer et al., 2008) define Competitor Profile Matrix (CPM) as a tool which helps the companies assess themselves against their major competitors using the critical success factors for that industry. A CPM creates a powerful visual catch point, it conveys information regarding your competitive advantage and basis for your company's strategic and is a useful tool to communicate those attribute and show how the competition is addressing them (Bygrave et al., 2010).

Table 1 Competitive profile matrix of Company A (Zimmerer et al., 2008).

Key SuccessFactors	Weight	Company A		Competitor 1		Competitor 2	
		Score	Weighted Score	Score	Weighted Score	Score	Weighted Score
Innovation	0.25	4	1.00	4	1.00	3	0.75
Advertising	0.20	2	0.40	3	0.60	4	0.80
Brand Name	0.20	1	0.20	4	0.80	2	0.40
Product Quality	0.15	4	0.60	2	0.30	2	0.30
Customer Service	0.10	3	0.30	2	0.20	1	0.10
Price Competitiveness	0.05	3	0.15	3	0.15	4	0.20
Technological Competence	0.05	3	0.15	1	0.05	2	0.10
Total	1.00		2.80		3.10		2.65

Table 1 shows a sample of CPM which is use to analyze data.

2.2 COCOMO II model

COCOMO II model is composed of three variants Application composition model, early design model, and Post architecture model. This is an extension of intermediate COCOMO model (Singh, 2007).

While using COCOMO II model the software cost (SC) is computed as follows:

SC = Effort estimate RELY TIME STOR TOOL EXP $15,000 (1)

Above equation represent effort estimate for any project which is based in person month, RELY represents reliability for any reliable system. The relevant multipliers are based on storage and execution time constraints (TIME and STOR), the availability of tool support (cross-compilers, etc.) for the development system (TOOL), and development team's experience platform experience (LTEX). $15000 represents average for one person-month.

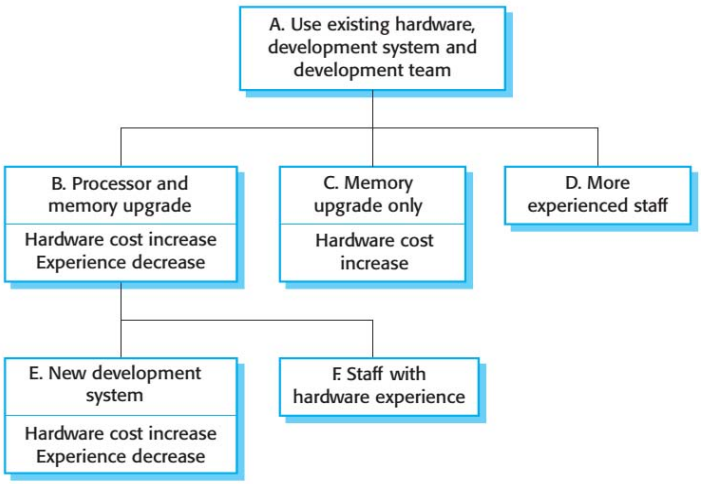

Fig. 2 Management option.

Table 2 Cost of Management options.

Option	RELY	STOR	TIME	TOOLS	LTEX	Total effort	Software cost	Hardware cost	Total cost
A	1.39	1.06	1.11	0.86	1	63	949393	100000	1049393
B	1.39	1	1	1.12	1.22	88	1313550	120000	1402025
C	1.39	1	1.11	0.86	1	60	895653	105000	1000653
D	*1.39*	*1.06*	*1.11*	*0.86*	*0.84*	*51*	*769008*	*100000*	*897490*
E	1.39	1	1	0.72	1.22	56	844425	220000	1044159
F	1.39	1	1	1.12	0.84	57	851180	120000	1002706

Table 2 shows some possible options for any project. These include spending more on target hardware to reduce software costs or investing in better development tools. Table 2 shows the hardware, software and total costs for the options A–F shown in Fig. 1. Option A represents the cost of building the system with existing support and staff. Option B shows that upgrading hardware does not necessarily reduce costs. Option C (upgrade memory) shows a lower cost saving but very low risk. Option D appears to offer the lowest costs for all basic estimates.

From Table 2 it seems that option D cost is less than all other estimations. It appears lower costs for all elementary estimates. According to Fig. 1 here no additional hardware expenditure is involved but new staff must be recruited onto the project. LTEX value will be low because of experienced staff. If we have already available in the company, it will probably be the best option to choose. Option A represents the cost of building the system with existing development team. It represents a base line for comparison. Option B represents that the upgrading does not necessarily reduce cost. The lack of staff experience (LTEX) and tool experience (TOOLS) with new hardware negates the reduction in the STOR and TIME multiplier. Option C (Upgrade memory) has a lower cost saving but very low risk. Option Ex represents new development system with existing staff.

*Code size = AVC * Number of function points* (2)

The code size is calculated from the number of function points. Using historical data analysis, the average number of lines of code, AVC, in a particular language required to implement a function point can be estimated. Values of AVC vary from 2 to 40 LOC/FP for a database programming language such as SQL.

2.2.1 Function points

This is a measurement based on the functionality of the program and was first introduced by Albrecht (Albrecht and Gaffney., 1983). Function points are calculated by counting the following software characteristics: 1. User-input types: data or control user-input types 2. User-output types: output data types to the user that leaves the system 3. Inquiry types: interactive inputs requiring a response 4. Internal file types: files (logical groups of information) that are used and shared inside the system 5. External file types: files that are passed or shared between the system and other systems.

The total number of EIs, EOs, EQs, ILFs, and EIFs, after applying the weights corresponding to the ratings (Low, Average, and High) will give the Unadjusted Function Point count (UFP).

The value adjustment factor (VAF) is calculated based on 14 General System Characteristics that rate the general functionality of the application being counted. The GSCs 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 sites, and Facilitate change. The degree of influence of each characteristic has to be determined as a rating on a scale of 0 to 5 as defined below (Souvik Daw et al., 2011)

0: Not present, or no influence
1: Incidental influence
2: Moderate influence
3: Average influence
4: Significant influence
5: Strong influence throughout

Once all the GSCs have been rated, Total Degrees of Influence (TDI) is obtained by summing up all the ratings. Now, Value Adjustment Factor is calculated using the formula:

$$VAF = 0.65 + TDI/100 \quad (3)$$

2.2.2 Final FP count

After determining the Unadjusted Function Point count (UFP) out of transactional and data function types, and calculating the Value Adjustment Factor (VAF) by rating the general system characteristics, the final Function Point count can be calculated using the formula: (Souvik Daw et al., 2011)

$$FP = Unadjusted\ Function\ Point\ count\ (UFP) * Value\ Adjustment\ Factor\ (VAF) \quad (4)$$

2.3 Related work

To the best of our knowledge no attempt has been made in the literature for cost estimation of feature oriented software development at design phase. Researchers are still working to solve the challenges and issues of software cost estimation, some related work is presented below.

2.3.1 REC

Nikoloas and Lefteris (Mittas and Angelis, 2010) presented Regression Error Characteristics (REC) analysis based virtual tool for software cost estimation. This method comprises geometrical properties along with simple inspection graphs to compare and validate different proposed models. It also uses cumulative distribution function for measurement of prediction error. REC practices collective distribution function for the measurement of prediction error. This method provides better management of projects as well as helps out to find the effects of errors by identifying the types of errors that affects the software cost.

2.3.2 Fuzzy clustering

In order to calculate software cost researcher has shown that sometimes parametric models are used instead of universal mathematical expression .In this regard, Javier et al presented a segmented model based on fuzzy cluster for software cost estimation. The use of fuzzy clustering permits obtaining different mathematical models for each cluster and also permits the items of a project database to contribute to more than one cluster, while conserving constant time execution of the estimation process (Aroba et al., 2008).

2.3.3 Estimation by analogy

Qin and Fang (Qin and Fang, 2011) debated three most popular methods that are being used for software cost estimation such as Scientists determined method, Analogy method and parameter model method. It is analyze that Parameter model COCOMO is most extensively used because of its estimation mathematical equation that provide flexible and dependable input and output factor for scheduling and work load during the project development. Assumption that history will repeat, mathematical equation is easy to be misunderstood in accuracy.

2.3.4 Multi objective

Prasad Reddy et al (Reddy et al., 2011) proposed a model for software cost estimation using Multi Objective (MO) Particle Swarm Optimization. In order to tune the parameters the multi objective particle swarm optimization methodology algorithm is applied It was experiential that the model gives better results when compared with the standard COCOMO model.

2.3.5 Support Vector Regression

Sweta and Pushkar (Kumari and Pushkar, 2013) make available a relative study on support vector regression (SVR), Intermediate COCOMO and Multiple Objective Particle Swarm Optimization (MOPSO) model. This model is used for estimation of software project effort. It has been observed through simulation that SVR gives better result in terms of accuracy and error rate in comparison of other estimating techniques.

2.3.6 Data mining techniques

Different data mining techniques were introduced by Zeynab and Farhad. For efficiency of software cost estimation they implied and tested their technique by NASA'S PROJECT. COCOMO succeeded in improving SCE. While using this technique they also proved that artificial intelligence methods are more reliable then algorithmic methods. On comparing Support Vector Regression (SVR), K-Nearest Neighbors (KNN), Linear Regression (LR) and Artificial Neural Network (ANN), they found that ANN and SVR are more efficient methods (Khalifelu et al., 2012)

In our research we have introduced another technique which is based on statistical analysis and used to find the cost estimation problem in feature model. Here we have introduce one of the most common factor of statistical analysis i.e. CPM. As discuss earlier CPM have major factor like weight, score and weighed score. In our research we have used weight as cost because physically software doesn't have any weight so here we have applied constraint of cost which is used to find the cost of different features which are to be selected. For score we have used constraint of commonality because in feature model through commonality we can judge the score of selected feature.

3 Case Study

In this we have selected E shop software product line that will discuss our terms and methodology that we have adopted in calculated estimation of cost of feature model. E shop is customer access software product line that uses in online shopping store. It's comprises of various features that are selected according to its different types of feature model like mandatory, optional, alternative, etc.

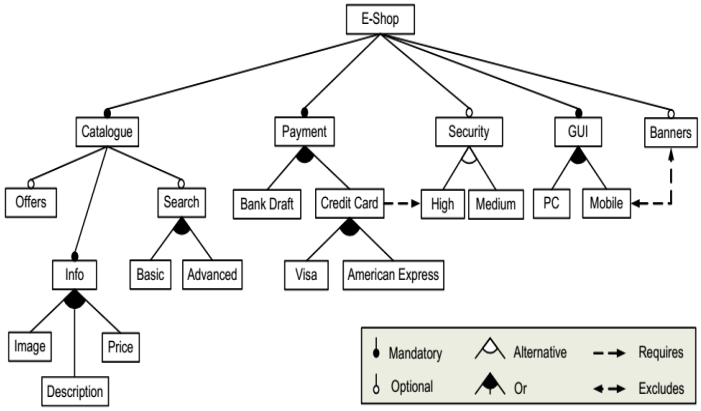

Fig. 3 A sample feature model (Segura et al., 2010).

3.1 Factor for finding function points in proposed methods

Catalogue: A catalog contains the goods and / or services that an e-shop compromises; therefore, an e-shop must support the use of a catalog. The catalog provides a outline to establish the information for goods and services, which can significantly influence the navigability and usability of the e-shop. In catalogue generally information comes from outsource as well from inner source. For example company information and product information given on web source, on the basis of this factor EIF value will be average(0.4), ILF will be high (1), EQ value will be high (1), EO will be high (1) and EI value will be high(1).

Offers: Offers is an optional feature that is provided by catalog. If customer wants different offers for goods or services than this feature is used. In this external data is being used for displaying offers against good or services for customer. Using this factor EIF value will be high (1), ILF will be average (0.5), EQ will be high (1), EO will be average (0.5) and EI will be high (1).

Search: Search is an optional feature that is used in e-shop for searching operation of any product or an items. It further contains two sub features Basic and Advance that lie in Or group. In this internal data as well external data is being process for searching against each query on customer as well user request. Using this factor EIF value will be high (0.8) ILF will be high (0.8), EQ will be high (1), EO will be high (1) and EI will be high (1).

Basic: Basic feature is used for basic searching process. E.g. if customer wants searching of an item just by name then this feature is selected. In this again internal data as well external data is being process for searching against each query on customer as well user request. Using this factor EIF value will be high (1), ILF will be average (0.5), EQ will be high (1), EO will be high (0.8) and EI will be high (0.8).

Advance: Advance feature is selected by customer for advance searching i.e. if customers want searching to be done by name as well by date than this feature is selected. In this internal data as well external data is being process against customer request. Using this factor EIF value will be average (0.7), ILF will be high (1), EQ will be high (1), EO will be high (1) and EI will be high (1).

Information: Information features describe all of the attributes that are recorded for a product in software product line. This information may comprise image, description, prices or all of them. In this external data is highly involved because customer may approaches to get any information against each item or good. So using this factor we can say that EIF will be average (0.5), ILF will be average (0.5), EQ will be high (1), EO will be high(1) and EI will be high (1). Same factor will be treated to image, description and prices.

Payment: This provides different modules of payment like Bank Draft, Credit card or both of them. For bank draft all banks requirement are in tags with it. User can also select credits card for easy usage but this also requires high security as well. Credits card can also be used in Visa, American express or both of them. In this

bank as a third party is being involved so we can say that internal as well external highly intake here. So on view of above factor EIF will be high (1), ILF will be high (1), EQ will be high(1), EO will be high(1) and EI will be average(0.7). Same factor will be used for Visa as well American express.

Security: E-shop implements a high or medium security policy (choose one).In security internal data is highly involved for checking process. So on this factor basis EIF will be average (0.5), ILF will be high (1), EQ will be average (0.5), EO will be low (0) and EI will be average (0.5). In case of high feature selection security its requires for other features so in this terms EIF will be high (1), ILF will be high (1), EQ will be average(0.5), EO will be low(0) and EI will be average(0.5). For medium feature same factor will be used as for security.

E-shop provides an interface for PC, Mobile or both of them can be used. For mobile it also excludes a banner. In GUI external process of feature is highly involved. Data here is being process against requirement of customer. So EIF will be high (1), ILF will be average (0.4), EQ will be average (0.5), EO will be high (1) and EI will be high (1). Same factor will be adopted for PC as well Mobile selection.

Banners: E-shop implements banner feature that is optional. For this displaying banner in different apps internal as well external data is involved. So EIF will be high (1), ILF will be high (1), EQ will be low(0), EO will be high(1) and EO will be average(0.5), EI will be low(0.2).

Now analyzing our technique. For this we will calculate cost of each feature with our techniques separately. Using Eq. 1 to find the software cost. In Eq. 1 we have effort estimate which we have calculated from code size while using Eq. 2. To find value for AVC in Eq. 2 for any data base used in feature model, we have categorize there range shown in Table 3.

Table 3 AVC range.

Values of AVC	Range of Features
2-10	10-15 features
10-15	16-35 features
16-20	36-60 features
21-25	61-75 features
26-30	76-95 features
31-35	96-110 features
36-40	111-125 features

As in our proposed feature model 22 features are being used so AVC would be 12 for this scenario.

In Eq. 4 we have UAF and VAF, UAF is calculated from sum of total EIs, EOs, EQs, ILFs, and EIFs. After applying weight for each features corresponding gives UAF. These values for each selected feature have been calculated as above.

In Eq.3 we have TDI Total Degrees of Influence (TDI) and is obtain in case of feature model there rating is calculated through following criteria, i.e.

Table 4 TDI value range.

TDI rating (value)	Feature types
1	Optional Feature
2	Or Group Feature
3	Or Alternative Feature
4	Mandatory

Now calculating values of Eq. 3, putting in Eq.4 we will have FP. Putting obtained values from Eq. 4 in Eq.2 we come to find code size used in Eq. 2. Having this value in Eq. 1 with all other values we will have Software cost for single selected feature.

A= Use existing hardware, development system and development team, B =Processor and memory upgrade, C = Memory upgrade only, D = More experienced staff, E= New development system, F. Staff with hardware experience. Using Eq. 1 we have calculated the cost of each feature.

4 Result Analysis of Feature Oriented Software Development

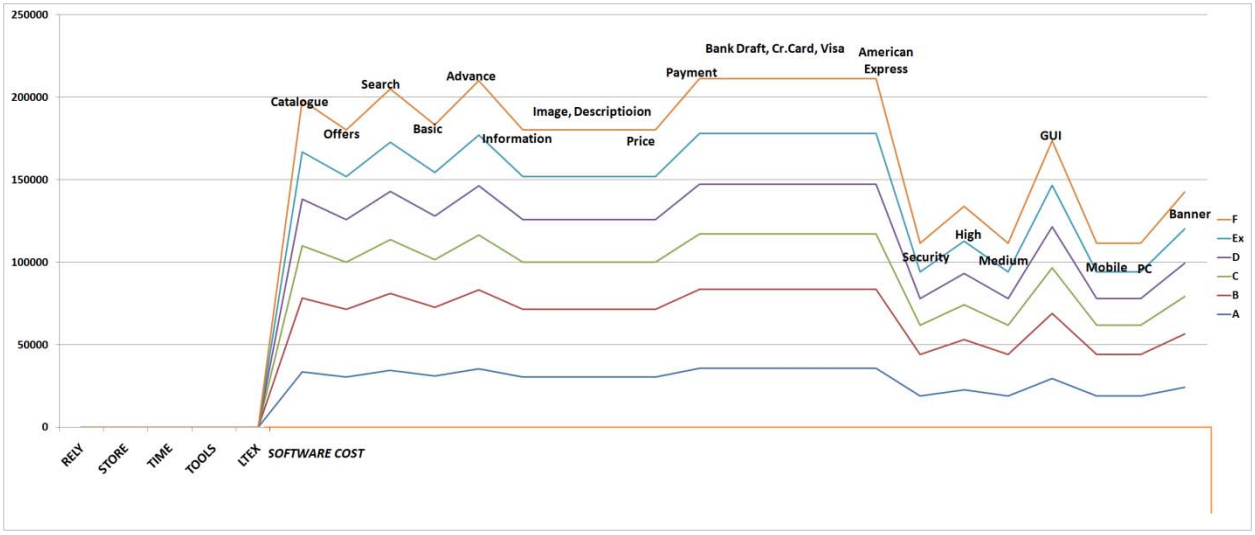

Fig. 4 Cost comparison of features selected from Feature Model depicted in Fig. 3.

Commonality: Commonality is an important factor which is used in FM. Commonalty informs us about the percentage of products in which an input feature appears. In our proposed method of CPM we calculate commonality of each feature from following formula.

Commonality = Number of instance in which feature appear/Total number of Instances (5)

After calculating commonality and software cost of each feature we will apply in CPM technique as show in Table 5.

For Example:

Product1: {Catalogue, Offers, Search, Basic, Info, Image, Price, Payment, Bank Draft, Security, Medium, GUI, PC, Banners}

Weighted score of Product1 :{ 33,360, 12,147, 24,157, 12,354, 30,369, 30,369, 12,147, 12,147, 35,603, 21,361, 15,046, 1808, 29,334, 20,533, 4,810}

Total Estimated cost of Product1= 265176.

Table 5 CPM of option A.

Feature	Commonality	Cost	Weighted Score
Catalogue	1	33,360	33,360
Offers	0.4	30,369	12,147
Search	0.7	34,510	24,157
Info	1	30,369	30,369
Image	0.4	30,369	12,147
Description	0.4	30,369	12,147
Price	0.4	30,369	12,147
Basic	0.4	30,887	12,354
Advance	0.4	35,373	14,149
Payment	1	35,603	35,603
Bank Draft	0.6	35,603	21,361
Credit Card	0.6	35,603	21,361
Visa	0.4	35,603	14,241
American Express	0.4	35.603	14,241
Security	0.8	18,808	15,046
High	0.7	22,547	15,782
Medium	0.1	18,808	1808
GUI	1	29334	29,334
PC	0.7	29334	20,533
Mobile	1	29,334	29,334
Banners	0.2	24,051	4,810

5 Conclusions

In this paper we used CPM a statistical analysis technique to find cost estimation based on feature oriented software development. For CPM two important factors are used, i.e., commonality and software cost. We calculated the cost of each feature separately by using formulas of COCOMO II cost analysis. The calculated cost is with different options available for the software development. Then we calculated the commonality of each feature. Finally we used both values to calculate the weighted score of each feature.

References

Albrecht AJ, Gaffney JE. 1983. Software function, source lines of codes, and development effort prediction: a software science validation. IEEE Transactions on Software Engineering, SE-9: 639-648

Al-Msie'deen R, et al. 2013. Mining features from the object-oriented source code of software variants by combining lexical and structural similarity. In: International Conference on Information Reuse and Integration. 586-593, IEEE, USA

Apel S, Kästner C. 2009. An overview of feature-oriented software development. Journal of Object Technology, 8(5): 49-84

Aroba J, et al. 2008. Segmented software cost estimation models based on fuzzy clustering, Journal of Systems and Software, 81(11): 1944-1950

Batory D. 2005. Feature Models, Grammars, and Propositional Formulas. 7-20, Springer, Berlin, Heidelberg

Benavides D, Segura S, Ruiz-Cortés A. 2010. Automated analysis of feature models 20 years later: A literature review. Information Systems, 35(6): 615-636

Black RKD, Curnow RP, Katz R, Gray MD. 1977. BCS Software Production Data. Final Technical Report, RADC-TR-77-116, Boeing Computer Services, USA

Bygrave William D, Zacharakis A. 2010. Entrepreneurship. John Wiley and Sons, NJ, USA

Chen Y, Gerald C, Gannod, James S. 2006. Collofello. A Software Product Line Process Simulator. Software Process: Improvement and Practice, 11(4): 385-409

Czarnecki K, Helsen S, Eisenecker UW. 2005. Staged configuration through specialization and multi-level configuration of feature models. Software Process Improvement and Practice, 10(2): 143-169

Dagdeviren M, Yüksel. 2008. Developing a fuzzy analytic hierarchy process (AHP) model for behavior-based safety management. Information Science, 178: 1717-1733

Daw S, Das A, Paul P. 2011. Annotation of function point model over size estimation. International Journal of Computer Science and Communication Networks, 2(2): 224-230

Fischer S, Linsbauer L, Roberto E. Herrejon L, Egyed A. 2014. Enhancing clone-and-own with systematic reuse for developing software Variants. In: International Conference on Software Maintenance and Evolution. 391-400, IEEE, USA

Jones C. 1996. Software Systems - Failure and Success. International Thomson Computer Press, London, UK

Kang K, et al. 1990. Feature-oriented domain analysis (FODA) feasibility study. Technical Report, Carnegie-Mellon University Pittsburg, SEI, USA

Khalifelu ZA, Gharehchopogh FS. 2012. Comparison and evaluation of data mining techniques with algorithmic models in software cost estimation. Procedia Technology, 1: 65-71

Knauber P, et al. 2002. Quantifying Product Line Benefits. In: Software Product-Family Engineering (van der Linden FJ, ed). 155-163, Springer, Netherlands

Kumari S, Pushkar S. 2013. Comparison and analysis of different software cost estimation methods, International Journal of Advanced Computer Science and Applications, 4(1)

Leung H, Fan Z. 2002. Software Cost Estimation. Department of Computing, Hong Kong Polytechnic University, Hong Kong

Martinez J, et al. 2015. Bottom-Up Adoption of Software Product Lines: A Generic and Extensible Approach. In: International Conference on Software Product Line. 101-110, ACM, USA

Mittas N, Angelis L. 2010. Visual comparison of software cost estimation models by regression error characteristic analysis. Journal of Systems and Software, 83(4): 621- 637

Pohl K, Bockle K, Linden VD and Frank J. 2005. Software Product Line Engineering: Foundations, Principles and Techniques. Springer, Netherlands

Pohl K, et al. 2005. Software Product Line Engineering (Vol. 10). Springer, Heidelberg, Germany

Qin X, Fang M. 2011.Summarization of Software Cost Estimation. Procedia Engineering, 15: 3027-3031

Reddy P, et al. 2011. Multi objective particle swarm optimization for software cost estimation. International Journal of Computer Applications, 32(3): 13-17

Segura S, et al. 2010. FaMa Test Suite v1.2. Technical Report ISA-10-TR-0. 1-52, Applied Software Engineering Research Group, University of Seville, Spain

Singh Y, Aggarwal KK. 2007. Software Engineering (3rd edition). New Age International Publisher Limited, New Delhi, India

Thomas E. 2008. SOA: Principles of Service Design (Vol. 1). Prentice Hall, USA

Zhang WJ. 2016. Selforganizology: The Science of Self-Organization. World Scientific, Singapore

Zhao Y, Zhang WJ. 2013. Organizational theory: With its applications in biology and ecology. Network Biology, 3(1): 45-53

Zimmerer T, Scarborough NM., Wilson, D. 2008. Essentials of Entrepreneurship and Small Business Management (5th ed). Prentice Hall, NJ, USA