A Review of Comparison among Software Estimation Techniques

Syeda Binish Zahra and Mohsin Nazir

Abstract- Software estimation process is still a complicated procedure for estimators. It is the responsibility of software project manager; that he effectively handles the budget problems and should manage whole work within given time slot. Lot of estimation models, techniques and approaches are used; but none of them be able to provide 100% accuracy in time, in cost or in any other estimation factors. But expert suggests if we use estimation techniques in proper manner than estimation process become easier or reliable. This paper comparing the software estimation approaches in depth, and makes the selection of appropriate estimation model easier.

Index Terms: Line-of-Code (LOC), Function Point (FP), constructive cost model (COCOMO), Unadjusted Function Point (UFP).

I. INTRODUCTION

Estimation means "prediction". It neither is used in plan nor in commitment. Estimation techniques are used to calculateapproximate results; made uncertain or incomplete data become usable. One person might consume a whole day to complete an assignment. Another person just requires some hours to complete that same task. This uncertainty is the cause of discrepancy between the answers of different peoples; when they are supposes to do the same task. So it is difficult to estimate the right time due to variance of statements.

This discrepancy can be solved, when the work is performed in right manner. In daily life, people do compares the quality of work with time. They though quantum of effort is only possible with quantum of time. In today's life this is not a 'measuring standard' of quality. Might be possible that a person who is taking lesser time; he suffer from "learning curve effect" and he can be apply suitable effort on task while using less time.

When we talk about time and effort of a specific task, we can't ignore its cost. Cost estimation is also important.

Although, budgeting is used but cost which is to be incurred is still vague in many situations. The problem of suspension variance highly affects the cost. So, the need is to use better estimator to predict the best estimate.

Syeda Binish Zahra, Mohsin Nazir, Department of Computer Science Lahore College for Women University Pakistan. Mohsin_sage@yahoo.com. Manuscript received April 9, 2012; revised July 30, and November 1, 2012.

II. ESTIMATION TECHNIQUES

In general, there are many techniques that describe the software estimation. But mainly estimation is divided into two forms / group [1]. (See Table 1)

1. Arithmetic 2. Non Arithmetic

These both forms are in use to perform accurate estimation, but extent of accuracy relates with the information collection. Software estimation has been considered as the most important but difficult job for the project managers.

- Complexity of estimation relates with project size and level of details.
- According to Boehm the problem should be indentified at the starting stage of estimation model process otherwise it will become so costly to recover it and restarting the process is also time-consuming [2].

A. Algorithmic Parametric Model

Is expressed in form of mathematical equations and uses theory or historical data. Algorithmic or parametric model based on especial mathematical equation or formula. This model firstly gathers all required data as input and then generates output by using its mathematical equation calculation. Algorithmic model is further categorized into some models and each model has its own mathematical equation to handle estimation. These models are.

1. FUNCTION POINT SIZE
2. SEER – SEM
3. COCOMO
4. COCOMO II
5. Checkpoint
6. Halstead's Metric

These models usually work from complete statistical data that gather from past completed software projects [3]. Validation of algorithmic model is difficult because a huge amount of data is required from last projects. (See Table 2, 3 & 4).

a. Function Point Size

Function point size method is introduced by Albrecht and in 1983 to calculate the functionality of software project [1]. The technique needs data from user as input and also for output it uses logical files, interfaces and inquires.

This estimation technique uses an equation that is a mixture of two things:

 ISSN – 1999-4974

1. Complexity
2. Weight

The complexity of project is measured by some indicators, like 1 for simple, 2 for medium and 3 for complex. UFP (Unadjusted function point) = complexity x weight [1].TFC is a mixture of following factors: data communication, change, complex interfaces, backup & recovery and online data entry [1].

To calculate function point, two factors made this equation FP= UFP × TFC [1]

The more recent technique of FP is object points, in which one counts objects instead of functions.

Advantages [4]

- Little detail of information is required.
- Easy in use
- Accurate in estimation, as other techniques
- Changing in data is allow
- Government and industry commonly used it

Disadvantage

- Input size is poor
- Cannot handle exceptional requirements

b. SEER – SEM: System Evaluation and Estimation of Resources --- Software Estimation Model

SEER-SEM model is proposed by Galorath in 1980 and mostly business projects are based on SEER-SEM estimation model. Defense industry and aerospace use it as software tool [4]. SEER-SEM uses size of software as input. This technique considers efficient because it handle all steps of project life cycle and application configuration like graphics, client- server, distributed and stand-alone, etc [5]. SEER-SEM takes size, complexity, constraints as input and effort cost, risk, schedule, maintenance and reliability as output [5]. This model is used for maintenance and development of software project [2] and considers being best in accuracy of estimation when effort is concerned.

The main drawback of this approach that it takes many parameters as input which complexes it and output as well. It is considered as a perfect tool for project planning and control.

c. Slim

Software life cycle model was developed by Putnam in 1970's to estimate project schedule, effort and rate of defect. To calculate estimation is used a model known as Rayleigh curve model. Firstly it take line of code as input of project then use Rayleigh curve model to generate estimation in effort [2, 7]. The big advantage of that model is that support size estimating method and drawback is that it is sensitive about estimates [5]. SLIM can analyze its data with previously finished projects. If previously data is not available for guideline then equation is used.

SLIM = Environmental Factor × (Manpower × Software Delivery Time) The problem of SLIM is that it is sensitive in nature and software project size must be define in early stage it is also not appropriate for small projects. The big advantage of this technique is that it uses linear programming, review techniques and statistical imitation [6].

d. COCOMO

The constructive cost model (COCOMO) is developed by Barry &Boehm in 1981 to calculate size and schedule, effort and duration. In this model, software size is used as input. The process of COCOMO completes in three steps [2] basic, intermediate and complete and COCOMO divided into three categories organic, semidetached and embedded [6].

Basic COCOMO calculate effort as E = a × (size) b where a and b are constant [7], size in FP, LOC, E=estimated effort. The successor of COCOMO is COCOMO II .The main point of that model is to be user (developer) friendly, and it makes it more reliable and acceptable so gives more accurate results. These equations used to calculate number of persons – months require in the software project [6] and size is estimated by line of code (LOC).

Organic: This mode is used in projects handled by experienced team, and also suitable for small projects.

Semi-Detached: "Semi" means in between. That mode is used when project somewhere in between.

Embedded: That mode is suitable for large projects and when developing team is unfamiliar with project requirements.

COCOMO is incremental model and according to incremental nature, it has a big challenge of degree of change [8]

e. COCOMO II

COCOMO II was introduced in 1994 and published in 1995 in annuals of Software Engineering. The basic intuition of COCOMO is that when size of project increases then put more effort to grow project in reliable manner. COCOMO II takes line of code, function point as project input [6]. This model is further categorized in three sub-models [5, 9].

Applications composition

- Used to estimate schedule and effort of project by using ICASE tools.
- Based on object

Early design

- Involves concepts and architected of system

 ISSN – 1999-4974

- Based on function points, 5 scales factors and 7 effort multipliers

Post Architecture

When top level design is finished that mo is used that model is useful when these two conditions meet.

1. Top level design is finished.
2. Complete information is available.

Means; the architecture of software is well established.

f. Check point

Check point was developed by capes Jones in 1997 as software project tool for estimation used by SPR (software productivity research). It takes function points as input which calculates functionality of software. It mainly focuses on three capabilities o software [5].

Measurement: Capture project material

Estimation: estimate effort, defects, cost, schedule and resources

Assessment: done comparison between estimated and actual performance.

It may work with source lines of code (SLOC) that make it complex. But it generate accurate results of estimates because it uses function points rather them LOC.

g. Halstead's Metric

This technique used to estimate effort, difficulty, complexity and program volume. It uses total number of operands and operators of program to calculate complexity of program.

$$\text{Complexity} = \sum (\text{operands} + \text{operators})$$

Also calculate time and effort of project by using effort equation, which is $E = V / L$

$$\text{Effort} = \text{program Volume} / \text{program Level}.$$

Benefit is that it is very helpful as identifying errors and limitation is that its equation is facing the controversy [2].

B. Non-Algorithmic

These estimate models work with the usage of previous data, which are similar in requirement. (See in table 2, 3, and 4)

a. Top Down

To down method concerned with complete characteristics of the system that we want to developed. Means overall estimation of project is calculated. When you have complete estimate for your project then you are able to sub- divide it into small modules. Because of that feature it is move reliable and accurate method. Benefit of that method is we improve accuracy at each step. The inputs of that approach are cost of documentation, configuration management and account integration [6].

a (i). Estimacs

This technique was developed by Howard Rubin in 1970's as sequential model. Input based on function point and because of its sequential nature, output of one phase becomes the input of next phase. It used staff development and size, cost, resource, effort, risk for development of project. Overall model based on interactive approach as top-down approach further divided into small modules. ESTIMACS is a top down approach, so it divided into fine sub-parts [5].

System Development Effort Estimation:

In this part total effort is calculated hours. It starts by the knowledge of following factors: complexity level, size, and developer knowledge about particular application area.

Staffing and Cost Estimation:

This part takes effort as input and also information of staff salaries to estimate proper cost of project. Output is staff distribution, number of team members and cost of all phases.

Hardware Configuration Estimates:

This part deals with which type of hardware is required for system (that fully supports the system). Inputs are types of applications and operating system. Output is power of required processor and storage medium.

Risk Estimates:

This sub part shows the complete picture of particular system with the help of answer that given at early 4 stages.

Advantage

- It uses function points as inputs and error rate is much less. [2]

Limitation

- Each stage not clearly translate the effort [10]

a (ii). Use Case Points (UCP)

This estimation technique was introduced by Kerner in 1993. It is version of Function Point but totally base on 'use cases' [9] It use functional requirements, determine number of use cases and estimate efforts. It produced 20 %actual effort at early stage than any other estimation technique [9].

Limitation

- It is not much accepted when we compare it with other estimation technique.

ISSN – 1999-4974

- We are not able to estimate until all use cases are in defined form.

Advantage

- It is reliable approach when it compared with FP and COCOMO [2]
- It is based on standardized notation of use case model[11]

a(iii). Predictive Object Point (POP)

This technique was developed by Minkiewicz in 1998. Later PRICE system has developed metric of POP to estimate effort. POP based on function points and its metrics based on object- oriented concept Advantage:

- Process can be automated [2]

b. Machine Learning

Machine learning demonstrating the accurate estimates. The accuracy is done by the "Knowledge" of datasets of finished projects.

b(i). Neutral Nets

Neural nets consist of layers of neurons that make a net because they are interconnected. Each layer get an input and neural net use sum of these layer as input, and produce output accordingly. The most outer layer used to represent set of input and output layer show actual effort of complete project. This approach is comparable and better them any statistical model. The limitation of that model is that there is no clear guidance for the design of neural nets and accuracy depends on training set [12].

c. Case-Based Reasoning (CBR)

Case-based reasoning also known as Estimation by analogy [13], is an estimation approach that use past cases to make new project reliable. This approach is reliable and beneficial because its performance is much better than algorithmic model and extremely near the approach of expert. First CBR system stored cases in a database. This database provide extreme base of knowledge that is useful for the developer of projects. Developers use this knowledge base to retrieve the most similar case with their new project. Another advantage is that results of CBR systems are more accurate than any

Expert Judgment approach [12]. CBR system depends on four steps (also known as CBR-cycle) [14].
Retrieve: get similar cases from database
Reuse: solution recommended with these similar cases.
Revise: fit into new case among participants. Team member's works individually and show their work at meeting that conducted in rounds.

Firstly all work is assigned to team members individually. In first round, results are gathered in tabular form from each participant and then these results returned to each participant for second round. New problem arise from first collected data and participants try to answer all the problems that arise at previous round. The round system will continue until all results are accurate and problem is shoveled [5].

This technique is useful only when participants are experts in their decisions and able to give more "expert's opinion" about the problem. A person is said to be "experienced" in particular field when he spend many of years in that field, but also it gives no guarantee of future knowledge because new needs and requirements after and may be expert can be wrong in this expertise.

Advantages

- Relatively simply process and useful in the absence of historical data.
- Communication among participants at each stage that ensure estimate are not over works so every point is discuss and final output in much accurate instead to individual estimation [2]
- Remove politics estimation is not based.

Limitations

- Depends on required management co-operation
- Team member should be agreed on agreement
- Team members should be experienced and able to share their ideas clearly
- Time- consuming as many participate involved[12]
- Useless when any of participant is absent.
- People that no interested, was forcedly because they consider being experienced.
- Expensive method

Retain: new complete case put into database.

Advantages

- Requirement of expert is null
- Provide akin to thinking of human
- Accurate prediction to handle cases even failure
- Provide efficient reasoning [15]
- Allowing faster knowledge acquisition [15]
- Provide unique explanation capability

Limitation

- Case data hard to gather
- Limited prediction about cases

C. Expert Judgment

This technique involves one or more experts (but not more than 20) that use their knowledge or experience for software estimation [16]. This approach strictly relies on the experience, knowledge. That knowledge may be drive

 ISSN – 1999-4974

from historical database. Historical database must be maintained with completed projects. The past study show that developer use their memories rather than a maintain database, which make this approach more risky. The usages of past experiences make new project more reliable and error free, which is main beneficial point of Expert Judgment. But estimation can be based [12, 17].

c(i). Delphi

Delphi technique was developed at Rand Corporation in 1969 and in 1981 Barry Boehm refines and renews the concepts of Delphi as Wide-band Delphi. This technique estimate schedule, effort and plan. Specifications are used as input and assumption about estimation, detailed task list and effort given as output. In this technique a team of experts is selected for software development. This team consists of 3 to 7 member with a person who plays the role of moderator [2]. The whole technique is divided into 6 step and these steps are conducted in a sequence. Team members interact with each other; because Wide-band Delphi incorporates much interaction and communication among participants. Team member's works individually and show their work at meeting that conducted in rounds.

Firstly all work is assigned to team members individually. In first round, results are gathered in tabular form from each participant and then these results returned to each participant for second round. New problem arise from first collected data and participants try to answer all the problems that arise at previous round. The round system will continue until all results are accurate and problem is shoveled [5].

This technique is useful only when participants are experts in their decisions and able to give more "expert's opinion" about the problem. A person is said to be "experienced" in particular field when he spend many of years in that field, but also it gives no guarantee of future knowledge because new needs and requirements after and may be expert can be wrong in this expertise.

Advantages

- Relatively simply process and useful in the absence of historical data.
- Communication among participants at each stage that ensure estimate are not over works so every point is discuss and final output in much accurate instead to individual estimation [2]
- Remove politics estimation is not based.

Limitations

- Depends on required management co-operation
- Team member should be agreed on agreement
- Team members should be experienced and able to share their ideas clearly
- Time- consuming as many participate involved[12]
- Useless when any of participant is absent.
- People that no interested, was forcedly because they consider being experienced.
- Expensive method
- Sometime difficult to coordinate and motivate a group of experts with diverse interest and busy schedules.

c(ii). Parkinson's Law (Parkinson's Principle)

Parkinson's Law proposed by UK historian and author Northcote Parkinson in 1955. In 1958 He wrote a book to describe human behavior in many situations. For software estimation this means that the estimation is determined by given resources. He stated that this is obvious if a person complete earlier his work,

The best point of this approach is it correlates with experience [14] and helping in distribution of project among team members [2].Reinforces poor practice [14] is the main drawback of this approach. It seems to be fine but not realist.

TABLE 1
CLASSIFICATION OF ESTIMATION TECHNIQUES

Software Estimation Techniques			
Algorithmic	**Non- Algorithmic**		
Function Point	**Expert Judgment**	**Machine Learning**	**Top Down**
SEER-SEM	Parkinson's Law	Neural Nets	ESTIMACS
SLIM	Delphi	Case-based Reasoning	Use Case Point
COCOMO	PROBE		Predictive Object Point
COCOMO II			
Checkpoint			
Halstead's Metric			

TABLE 2
COMPARISON BETWEEN ALGORITHMIC AND NON-ALGORITHMIC MODELS

	Algorithmic / Parametric [3]	**Expert Judgment**	**Machine Learning**	**Top-down**
Based on	Especial Algorithm	Known follow up of all the past project	System that effectively "learn" how to estimate from training set of completed projects	Overall characteristic of the system to be developed
Used	Mathematical equations, theory or historical data	Project size, total effort, breakdown on activities	Training rules of estimation	Integration, configuration management, documentation costs
Useful in		When lack in the scope of the project	When data-base is maintain about past projects	
Techniques	size, SEER-SEM, COCOMO, SLIM, Checkpoint	Parkinson's Law, Delphi, PROBE	Case-Base Reasoning, Neural Nets	ESTIMACS, Use Case Point –UCP, Predictive Object Point - POP

ISSN – 1999-4974

III. SUMMARY

The whole work is summarized into these tables.
In general estimation techniques divided into two parts

TABLE 3
FURTHER CLASSIFICATION AND COMPARISON OF ESTIMATION TECHNIQUES

Sr. No	Methods	Accuracy	What can estimate	Applicable system	Available Tools	Input
Expert-Based Estimation Methods						
1	**Parkinson's Law**	Low	Effort	Heuristic approach of development is required for system	3-point estimating	Existing resources
2	**Delphi**	Depend on estimator's skills	Work break down structure, Effort	Agile development	Delphi freeware	Vision and scope documents
3	**PROBE**	Good	Size, Effort	Data-base system	PSP (personal software process)	?
Algorithmic Estimation Methods						
4	**Function Point**	Low	Size, Effort		Mark II FPA	?
5	**SEER-SEM**	Low	Effort, defects, staffing and duration	All types of projects	SEER of software Version 7.3, SEER for IT	Size metrics, complexity

TABLE 4
COMPARISON AMONG ESTIMATION TECHNIQUES

6	**SLIM**	Good	Effort, Time	Agile development, service-oriented architecture	SLIM estimator, COSMOS	?
7	**COCOMO**	Good [11]	Effort, Time	All types of projects	COSTAR 7.0	Size (primarily LOC / Subsequently using FP)
8	**COCOMO II**	Good	Effort, size, cost, schedule	COSTAR 7.0	COSTAR, SOFT-CALC, WICOMO	?
Sr. No	**Methods**	**Accuracy**	**What can estimate**	**Applicable system**	**Available Tools**	**Input**
9	**Checkpoint**	Good	Effort, cost, schedule, staffing	All types of projects	CHECKPOINT	Function Point, Feature Point, Source Code
10	**Halstead's Metric**	Good	Effort, Time, complexity	All types of projects	Metric Advisor	Program complexity
Top Down Estimation Methods						
11	**ESTIMACS**	Relatively good	Cost, Effort, Risk	Object-oriented system	Estimacs	Customer complexity, Function Point
12	**Use Case Point (UCP)**	Good [11]	Development effort, Test effort,	Object-oriented system	EZEstimate, EstimatorPal, UCPPal,	Use cases (size)
13	**Predictive Object Point (POP)**	Good	Effort	Object-oriented system	EstimatorPal, SOFT-CALC	Complexity, size
Machine Learning Estimation Method						
14	**Neural Nets**	Low	Effort	?	?	Project Size
15	**Case-Base Reasoning**	Good	Effort	Data-Base, CBR-DSS, SQUAD	CASPIAN, Remind 1.3 [1]	Set of Cases

ISSN – 1999-4974

IV. CONCLUSION

Software estimation is an important part of the process of software development. The selection of appropriate estimation model in software project management is still difficult task. This paper gives comparison among different estimation models or techniques in detail and provides more efficient and straightforward way for estimator to select right model to estimate the software. In the near future I decide to explore the estimation techniques in more depth to obtain the right model for estimating that gives accurate results.

REFERANCES

[1] Vahid Khatibi, Dayang N. A. Jawawi, "Software Cost Estimation Methods: A Review", In proceeding of Journal of Emerging Trends in Computing and Information Sciences, Vol.2, No. 1, ISSN: 2079 8407.

[2] Samaresh Mishra, Kabita Hazra, Rajib Mall, "A survey of Metrics for Software Development Effort Estimation" In proceeding of International Journal of Research and Reviews in Computer Science, Vol. 2, No. 5, October 2011, ISSN: 2079-2557.

[3] Sarah Jane Delany, Padraig Cuningham, "The Application of Case-Based Reasoning to Early Software Project Cost Estimation and Risk Assessment", TCD-CS-2000-10

[4] Anatol Siamionau,"Requirements estimation guideline for IT projects at Scania Networks AB", Master of Science Thesis, Stockholm, Seden 2011, TRITA-ICT-EX-2011:168

[5] Dr Barry Boehm, Sunita Chulani, "Software Development Cost Estimation Approaches" A Survey, 1998, PhD Thesis, Computer Science department at USC, university of Southern California

[6] Mehwish Nazir, "A survey of Software Estimation Techniques and Project Planning Practices", In proceeding of IEEE 7th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and parallel/ Distributed Computing (SNPD'06), 2006

[7] Yang Lanrong, "ACase-Based Reasoning Decision Support System", in proceeding of Canadian Social Science, Vol. 1 No. 3 November.

[8] Parastoo Mohagheghi, Bente Anda, Reidar Conradi, "Effort Setimation of Use Cases for Incremental Large-Scale Software Development" in Proceedings of the 27th international conference on Software engineering (ICSE'05), ACM, May 15-21, 2005, St. Louis Missouri, USA. (INDIACom-2011), Computing for Nation Development, New Delhi March 10 – 11, 2011.

[9] Stephen G. MacDonell, Martin J. Shepperd, "Combining techniques to optimize effort predictions in software project management", In proceeding of Journal of System and Software 66 (2003) 91-98

[10] Pascal Ziegler, "Manual Techniques, Rules of Thumb", seminar on software cost estimation, WS 2002 / 03

[11] Li D. Xu, "Case-Based Reasoning: A major paradigm

[12] of artificial intelligence", In proceeding of IEEE, 1994 [16]. Hareton Leung, Zhang Fan,"Software Cost Estimation", IN: HANDBOOK OF SOFTWARE ENGINEERING AND KNOWLEDGE ENGINEERING, WORLD SCIENTIFIC PUB. CO, RIVER EDGE, NJ, 2002

[13] David Cok, "Active Learning for Effort Estimation", In proceeding of Journal of IEEE Transactions on Software Engineering, Vol. X, No. Y, Somemonth 201Z. [9]. Jongmoon Baik, "The Effects of Case Tools on Software Development Effort", PhD Thesis, Faculty of the Granduate School, University of Southern California, December 2000.

[14] FJ Heemstra,"Software Cost Estimation"' Information and Software Technology, Vol. 34 No.10, October 1992, Butterworth-Heinemann Ltd. pp.627- 639

[15] MelDamondaran, Aqua Netta E. Washington, "Estimation Using Use Case Points", in proceeding of ISECON 2002, San Antonio, 2002

[16] Abdulbasit S. Banga,"Software Estimation Techniques", In proceeding of National Confer