Steps towards Costing

Naila Parveen
Jinnah University for Women
Karachi, Pakistan
Email: nailaparwin@gmail.com

Mohammad Ayub Latif
College of Computing and Information Systems
Pafkiet University
Karachi, Pakistan
Email: malatif@pafkiet.edu.pk

***Abstract*—Accurate software estimation is a challenging task from decades while managing a project. A lot of work is done to estimate with maximum accuracy but increasing complexity and uncertain risks made it more difficult to achieve. Although its difficult but not impossible. As estimates play an important role for the success of software project, therefore requires significant attention in the early stages of software development activities. This paper presents the important activities that need to be performed while estimating. The aim is to provide a life cycle of activities including estimating project size, effort, uncertainties, and complexities and so on. Here a line of action is presented to show a step by step solution which provides ease as well as accuracy in estimation.**

***Keywords*—software estimation; estimation process; steps of estimation; estimation life cycle**

I. Introduction

Estimation is a significant activity while developing software. It provides foundation for managing and planning a project. The success or failure of software project depends upon its estimation. The more accurate estimate is, the more chances to success. Estimates are usually made by experienced and knowledgeable persons. To develop any software they express their opinions about how much will it cost based on their experience. But no one can predict the future, what problems can arise? what will happen during development? what is the probability that every thing will go right?
So the target is to achieve estimates that are closer to the actual figures. Obviously no one can get hundred percent accurate estimates.

A. Estimation an Umbrella Activity

It is actually a continuous process starts with the early stage of software development process and remains continue till the completion. Estimates are often required early, in the beginning of software development process, within limited time period so that managers can plan and manage projects. Another reason for early estimate is to be able to do a contract with the customer. After providing a baseline, they are often updated due to various reasons. some resulting from customer side and some from developers side. Some of them are

- Change in requirements
- Adding more features
- Unpredictable risk
- Inaccurate baseline estimates
- Unclear scope

B. How do we estimate?

We estimate using our previous experience and historical information; therefore there is a significant improvement in our estimates with the passage of time. If a manager works on same type of projects, with similar technology and complexity, he can predict easily and accurately. But the world is getting change rapidly. The same case is in software industry. More experienced persons can make more accurate estimates but rapid changes in hardware and software technology and evolving nature of software make it more difficult and challenging to estimate accurately. The factors that affect estimation accuracy are

- Project Complexity: the more complex a project is, the more difficult to estimate.However it is relative to past experience. A person working on e-commerce application consider an accounting project more complex but a person working on 2-3 accounting projects would consider it easy to estimate.
- Project Size: As the size increases interdependency among various elements also increases. Decomposition techniques can be used to break a project into smaller components.
- Structural Uncertainty: Difficulty to compartmentalize functions and the degree to which requirements have been solidified.
- Availability of historical information: With the help of metrics, defined using past projects, estimates can be more accurate [1]

some other factors affecting accuracy are

- Risk: Degree of Uncertainty involve in the project. uncertainty increase if scope of software is poorly defined
- Constraints: Internal or external constraints affect estimation accuracy
- Team Skills: Team skills also have an influence on estimates.
- Team Experience: A team having no previous experience in domain, technology or platform can affect estimation accuracy.
- Quality: Estimates get affected with increase in quality expectations.

C. Conservative vs. Aggressive Estimates

One of the main factors that affect estimation accuracy is, our estimates are often aggressive or conservative. It Depends on human nature, how a person thinks about certain things in life. If we ask a question "how much will it take to reach from one building to another" to two different persons. Each will have a different answer. Individual Characteristics can affect estimates. The fact is that if we use single point estimates, then there are more chances to inaccuracies. To avoid this instead of using single point estimates always use three points, best case, most likely and worst case estimates.[2]

D. What we get through estimation?

Estimation results in three dimensions, the effort, cost and schedule. To do this most of the techniques focus on estimating project size and effort. And by using effort find out required resources ,over all cost of the project and schedule to complete the system. Therefore determining correct project size and convert it into effort accurately leads to successful estimate. But unfortunately there is no fixed amount of ratio in project size and effort required. Project size can be measured using direct measures like LOC(Line of Code) or by indirect measures like FP(Function Points). Many tools are available to convert size into effort. Effort can be measured in person months.[3]

E. Why do estimates go wrong?

Some reasons of software failures are

- Ambiguous or Incomplete requirements
- Inappropriate size or effort estimation
- Lack of skills
- Off the cuff estimates

Software fails in both cases whether estimates are overrun or under run. Why do software projects over or under run? Both have different causes but usually software fails due to overrun of estimates. The penalty associated with over estimation is linear while with under estimation is non linear. So under-estimation can be more disastrous than overestimation. Some reasons for over estimation are

- Unclear requirements Specification.
- Schedule is not updated with requirements change.
- Aggressive development schedules.
- Insufficient resources.

Some reasons for under estimation are

- to avoid Parkinson's Law
- Youth Optimism
- Subjectivity and Bias

Category	Suitable opportune Estimation approach	Suitable implementation of estimation approach
Formal estimation model	Analogy-based estimation	ANGEL, Weighted Micro Function Points
	Parametric models	COCOMO, SLIM, SEER-SEM
	Size-based estimation model	Function Point Analysis, Use Case Analysis, SSU (Software Size Unit), Story points-based estimation in Agile software development
Expert estimation	Group estimation	Planning poker, Wideband Delphi
Combination -based estimation	Mechanical combination	Average of an analogy-based and a Work breakdown structure-based effort estimate
	Judgmental combination	Expert judgment based on estimates from a parametric model and group estimation

Fig. 1. Estimation Approaches!

F. Needs of Periodic Estimates

There are some limitations in achieving estimation accuracy. The uncertainty in software estimates can be reduced till a particular level at a specific time period. In the beginning the uncertainty is high due to variability in input parameters. Later it is due to variability in estimation models. Periodic estimates are really important throughout the development period of software projects; so that problems can be identified in advance and some corrective action can be done.

At the initial stage, everything is in ambiguity. The description about project is vague. Although the requirements cannot be completely understood but the objectives of software project is completely understood. At this level, the maximum possible accuracy is plus minus 50 percent. Informal techniques (i.e., historical comparisons, group consensus, and so on) can be used at this stage

When requirements have been confirmed, an estimate that is function-oriented may be made to estimate project. At this level, for an experienced estimator the possible accuracy is plus minus 25 percent.

Finally, before coding begins, when detail design is done, an implementation-oriented estimate may be made. The accuracy of this estimate is plus minus 10 percent.

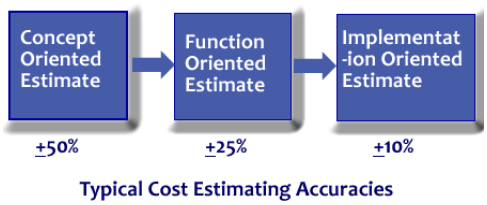

Fig. 2. Estimation Accuracy!

II. Effort Estimation Approaches

There are different estimation approaches. Any of the approach can be used from top down or bottom up. Each has its own advantages and disadvantages. In-fact two or more approaches are used simultaneously to estimate effort. The reason is to get more reliable and accurate estimates. They can be categorized as follows:

- Expert estimation: In this step estimation is formed on the basis of processes judgement.
- Formal estimation model: In this step estimation formed on the basis of mechanical processes, (the use of formula derived from historical data.)
- Combination-based estimation: In this step estimation is formed on the basis of judgemental or mechanical combination of estimates from different sources.

Different tools are available to assist project managers in effort estimation. They worked on different characteristics of projects like size, domain, complexity etc. In past, estimation based on only size of software. Managers analyze existing data, apply some formula on it to match with previous data. Obviously this sort of estimation is not so much accurate. Now different parametric models are also used as well as expert opinions to increase accuracy rate [4]

A. *Project Attributes affecting Estimates*

After the software size has been estimated, the baseline of estimates can be achieved based on some certain attributes of the project. There are a variety of attributes that can increase or decrease result of the estimate including the following:

1. The rate at which project requirements may change

2. The experience of the development team with this kind of project

3. The process or methods used to develop the project ranging from Agile to Waterfall

4. The specific activities that will be performed during development

5. The number of increments, iterations, or sprints that will be used

6. The programming language or languages utilized

7. The presence or absence of reusable artifacts

8. The development tool suites used to develop the project

9. The environment or ergonomics of the office work space

10. The geographic separation of the team across multiple locations

11. The schedule pressure put on the team by clients or executives

12. Contractual obligations in terms of costs, dates, defects, or features

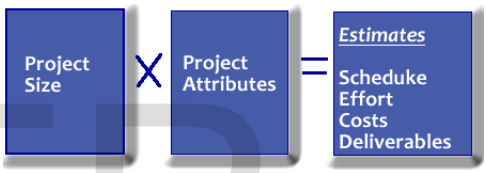

Fig. 3. Software Estimating Principles

Using commercial estimating tools, these project attributes can either be supplied by the user or inherited from similar projects already stored in the estimating tools knowledge base. In a sense estimating tools share some of the characteristics of the object-oriented paradigm in that they allow inheritance [5]

B. *Software Cost Components*

Software Cost include all of the following factors.

1. Hardware and software costs.

2. Travel and training costs.

3. Effort costs (the dominant factor in most projects)

a. The salaries of engineers involved in the project;
b. Social and insurance costs.

4. Effort costs must take overheads into account

a. Costs of building, heating, lighting.
b. Costs of networking and communications.
c. Costs of shared facilities (e.g library, staff restaurant, etc.).

III. Estimating Steps

The proposed Estimation process consists of following steps

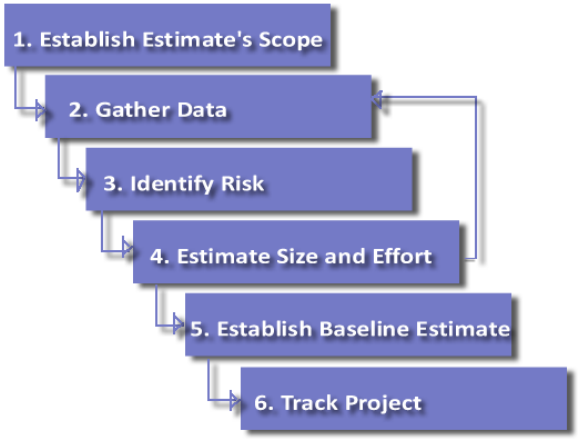

Fig. 4. Estimation Steps!

1. Establish Estimates Scope:

Define the purpose of the estimation, what to estimate, the degree up to which information is required, the persons involve in estimation process, the time frame required to estimate, the process to follow. The aim is to share information between all stakeholders and clearly assigned tasks and roles to each of them. Document all necessary information including top level specification about software, technical requirements, business requirements, dependencies and so on. Include as much detail as possible. At the early stage the information is preliminary and not that much accurate but it will be refine as the estimation process will go to next step.

1	Establish Scope Determine	The estimate's purpose ; The level of detail required; Who will receive the estimate; Estimating Team; Timeline; The overall scope of the estimate

2. Gather data:

Find data sources, gather data about software characteristics, technology involve, constraints, necessary assumptions, limitations, dependencies, inspect similar legacy software for historical data, create a document contains all necessary information. Reusable code or Use of COTS should be mentioned also. Collect as much detail as possible. Interviews, meetings, questionnaires, previous documents can be used to gather detailed knowledge about the software functionality. At the early stage the estimates contains more uncertainty, because of unreliable information. Try to put more focus on information accuracy

2	Gather Data Determine	Find possible data sources; Assess all functional and non functional requirements; Investigate all data sources to collect data about software; Software characteristics; Technology involve; Similar projects to get historical estimates; Document all Features

3. Identify Risk:

Anything that can go wrong during software development is a risk. Some are known in advance are some encountered suddenly. A risk when occurs results in some form of loss. Usually it affects our estimates. Their impact can be negligible, marginal, catastrophic or critical. Catastrophic impact can be disastrous. A risk can be technical, project or business risk. The goal is to identify the risks, find their types, impacts and prepare a line of action to manage them. A person using domain experience can find risk easily, assess its impact and can make a plan to manage it.

3	Risk Identification Determine	Determine each type of risk associated; Analyze risk severity and its probability to occur; Determine the risk-adjusted estimate ; Prepare a plan to manage risk; Document all necessary information

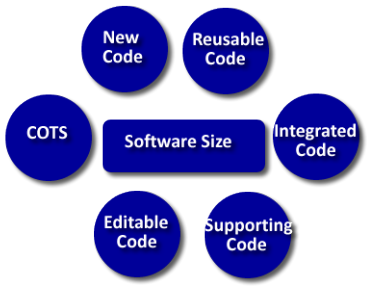

Fig. 5. Elements to Size Software

4. Estimate Size and Effort:

Size is the most important driver for estimation. Most of the time during estimation is dedicated to software sizing. Software size include amount of new code as well as amount

of existing code and other code generated using automated tools. To avoid uncertainty the size is estimated using three points whether expressed in LOC or FP. Another important task is to convert size into appropriate efforts. This can be done using historical data. Again this needs careful attention. Only experienced and trained people must perform sizing and effort conversion tasks. Any automated tool can be used to measure size and effort software. Usually two or three different techniques are used to determine accurate size.

4	Software Size Determine	Work breakdown structure; Estimate each element of WBS independently; Cross check for cost and schedule; Develop an estimating checklist;

5. Establish Baseline Estimate:

Estimates result in budget and schedule. Inaccuracy in estimates leads inaccuracy in budget and schedule. Different approaches can be used to make estimates like top-down approach, bottom-up approach, expert judgement and cost models. Document all these estimates. cost estimates contains fixed and variable amount of expenses. Risk estimates should also include in cost estimates. A reasonable amount of management reserve is added to estimates. Schedule estimates should have a reasonable time line to complete the software within given period. Add also appropriate contingency reserve for risks having high probability and impacts.

5	Baseline Estimate Determine	Document the purpose, team and approval committee of estimates; define rules and assumptions; describe detailed steps of estimation; Define a baseline for each estimate

6. Track Project:

Update estimates to reflect scope changes. Compare estimates with actual schedule and cost to find how much realistic estimates are and update lesson learned database. Track during development and after the completion of project. periodic information is often more effective to find bugs and improve skills of estimators

IV. Conclusion

The paper discusses importance of accurate estimation with several reasons that causes inaccuracies in estimations. Different categories of effort estimation techniques are also

6	Track Project Determine	Document estimate, costs, parameters, drivers and changes that occurred that affected the estimate

provided to help in assessing effort required. Mainly the paper provides a six point strategy to help in estimation. The most important activity during estimation is sizing , more time dedicated to sizing leads successful estimate. . This six point process will help managers to estimate every type of software project with more accuracy. The primary conclusion is that no single technique is best for all situations, and that a careful comparison of the results of several approaches is most likely to produce realistic estimates. Another important factor is to assess risk and quality of project and add a contingency reserve for these.

Acknowledgment

The authors would like to thank Mr. Ayub Latif to his guidance throughout the research activities and all of other persons who helped in any manner to complete this paper.

References

[1] W. Roetzheim, "Estimating software costs," *cost Expert Group*, vol. 1, p. 6, 2010.

[2] R. Pressman, "Software Engineering," *A Practioners Guide*, vol. 1, p. 6, 2010.

[3] D. Galorath, "The 10 Step Software Estimation Process For Successful Software Planning, Measurement and Control," *A Practioners Guide*, vol. 1, p. 6, 2010.

[4] Kardile Vilas Vasantrao, "Understanding of Software effort Estimation at the early Software Development of the life cycle - A literature View," *International Journal of Engineering Research and Applications*, vol. 2, p. 5, 2012.

[5] "Introduction to Software Cost Estimation," *Chapter 1*, vol. 1, p. 22, 2012.

[6] Vijayakumar, S. , "Use of Historical Data in Software Cost Estimation," *Computing and Control Engineering Journal*, vol. 1, p. 22, 1997.

[7] Ali Bou Nassif 1, Luiz Fernando Capretz2, Danny Ho3 , "Use of Historical Data in Software Cost Estimation," *International Conference on Emerging Trends in Computer Science, Communications and Information Technology*, p. 9, 2009.

[8] Barry Boehma, Chris Abts a and Sunita Chulani. , "Software development cost estimation approaches A survey," *Annals of Software Engineering 10 (2000) 177205*, vol. 1, p. 29, 1997.

[9] Prof. David A. Penny , "An Estimation-Based Management Framework for Enhancive Maintenance in Commercial Software Products," *A Release Plan*, vol. 1, p. 14.

[10] Magne Jrgensen, Simula Research Laboratory Martin Shepperd, Brunel University , "A Systematic Review of Software Development Cost Estimation Studies," *A Review*, vol. 1, p. 55.

[11] 1Vahid Khatibi, 2Dayang N. A. Jawawi , "Software Cost Estimation Methods: A Review," *Journal of Emerging Trends in Computing and Information Sciences*, vol. 2, p. 9, 2010-2011.

[12] David Hulett , "Integrated Cost Schedule Risk Analysis," *Chapter 1*, vol. 1, p. 15.