ISSN: 2278 – 1323
International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)
Volume 1, Issue 7, September 2012

Survey of Various Cost Estimation Techniques

Ms. Shruti Jain

Abstract: Cost Estimation is the process of predicting the total cost required to develop software. It is the most difficult and challenging phase in software engineering. The objective of this paper is to present different models available for the effort estimation and describe their advantages and disadvantages.

Key Terms: Software cost, effort, estimation, cost models

INTRODUCTION

Before developing software, it is desirable to know how much the software will cost and how much time it will take. The bulk of the cost of software development is due to the human effort, and most cost estimation methods focus on this aspect and give estimates in terms of person-months. There are different models available for effort estimation.

WIDEBAND DELPHI TECHNIQUE

The wideband Delphi Technique has been used in a large number of cost estimation activities. It is an expert judgment method. It basically consists of two meetings. First meeting is called as Kickoff meeting and the second one is called as Estimation session.

Kickoff Meeting: The objective of this meeting is to prepare the members for estimation session. Before the kickoff meeting starts, every estimator is given all the details of the project that are required to perform estimation.

Steps:

1. If any team member has not yet read the documents then the moderator reviews it with the team.
2. The moderator reviews the goal of the estimation session with the team, and checks that each team member is sufficiently knowledgeable to contribute.
3. The team discusses the software under development and brainstorms any assumptions.
4. The team generates a task list consisting of 10–20 major tasks. These tasks represent the top level of the work breakdown structure. This high-level task list is the basis for the estimates that are going to be created
5. The team agrees on the units of estimation (days, weeks, pages, etc.).

Estimation Session: Moderator provides each estimator with a specification and a form to record the estimates.

1. Moderator calls a meeting of all the estimators in which the estimators discuss various issues related to estimation with the moderator and each other. They can ask questions from moderator.
2. Estimators write down the reviews on the form.
3. Moderator summarizes all the forms given by estimators on an iteration form.
4. Moderator calls a group meeting, specially focusing on having the estimators discuss points where their estimates varied widely.
5. Estimators fill out forms, again anonymously, and steps 4 and 6 are iterated for as many rounds as appropriate.

Advantages:

- The estimators can find out differences between past project and requirements of the proposed project.
- The estimators can factor in project impacts caused by new technologies, architectures, applications and languages involved in the future project.

Disadvantages:

- This method cannot be quantified.
- It is hard to document the factors used by the estimators or estimators-group.
- Estimator may be biased, optimistic, and pessimistic.

ESTIMATING BY ANALOGY

In this method, proposed project is compared with the previous similar projects.

The steps used in estimating by analogy are:

1. Characterizing the proposed project.
2. Choosing the most similar completed projects whose features have been stored in the database.
3. Calculating the estimate for the proposed project from the chosen completed projects by analogy.

Advantages:

1. The estimations are based on actual project characteristic data.
2. The estimator's past experience and knowledge can be used which is not easy to be quantified.
3. The differences between the completed and the proposed project can be identified and impacts estimated.

Disadvantages:

1. With this method, estimators have to choose attributes to best describe the project. These attributes are restricted to the information available at the time of prediction.
2. Estimators have to provide weight age to each attribute so as to establish proper analogy.
3. This method is used with only those project which have analogy in past and cannot be used with new projects.

TOP-DOWN APPROACH

Top-down Approach is also called Macro Model. Using this approach, an overall cost estimation for the project is derived from the global properties of the software project. The cost is then divided into various low-level components. This method is generally suitable in the early phase of the software development. The best example of this approach is Putnam Model.

Advantages:

- It focuses on system-level activities such as integration, documentation, configuration management, etc., many of which may be ignored in other estimating methods and it will not miss the cost of system-level functions.
- It requires minimal project detail, and it is usually faster, easier to implement.

Disadvantages:

- It often does not identify difficult low-level problems that are likely to raise costs and sometime tends to overlook low-level components.
- It provides no detailed basis for justifying decisions or estimates.

BOTTOM-UP APPROACH

This approach is completely opposite from top down approach. With this approach, the cost of each software component is estimated and then combines the results to find out an estimated cost of entire project. The requirement for this approach is that an initial design must be in place that indicates how the system is decomposed into different components.

Advantages:

- It permits the software group to handle an estimate in an almost traditional fashion and to handle estimate components for which the group has a feel.
- It is more stable because the estimation errors in the various components have a chance to balance out.

Disadvantages:

- It will not consider many of the system-level costs (integration, configuration management, quality assurance, etc.) associated with software development.
- It may be inaccurate because the necessary information may not available in the early phase.
- It tends to be more time-consuming.

PRICE-TO-WIN

The estimation is based on the customer's budget instead of the software functionality. The software cost is estimated to be the best price to win the project. For example, if a reasonable estimation for a project costs 100 person-months but the customer can only afford 60 person-months, it is common that the estimator is asked to modify the estimation to fit 60 person-months effort in order to win the project.

Advantages:

- Estimation Cost is according to the customer’s choice.

Disadvantages:

- It is very likely to cause a delay in delivery or force the development team to work overtime.
- Developers may suffer loss.

PROBE

Proxy Based Estimating (PROBE), is the estimation method introduced by Watts as part of the Personal Software Process. The PROBE technique inherently relies upon a computer science principle called separations of concerns. It is based on the previous developments in similar application domains. PROBE is based on the idea that if a developer is building a component similar to one he built previously, then it will take about the same effort as it did in the past.

In the PROBE method, individual developer use a database to keep track of the size and effort of all of the work that they do, developing a history of the effort they have put into their past projects, broken into individual components. Each component in the database is assigned a type (“calculation,” “data,” “logic,” etc.) and a size (from “very small” to “very large”).When making a new estimate, the developer will base his or her estimate on the type of concern being implemented, and the corresponding relative size.

For example, suppose Deborah is a C# developer. She has been tasked to develop a program to read the current location of a train from a database, and then update a display screen based upon the data. After separating the concerns of data access and screen update, and then enumerating the number of methods needed for each class, Deborah concludes that she will need three methods to access the database, and seven methods to update the screen. Consulting her PROBE relative size table for C#, Deborah finds that the average size of a data access method is 8 lines of code, while the average size of an animation method is 19 lines of code. Deborah then uses the PROBE procedure to apply some statistics based upon her past estimation accuracy, and arrives at an estimated program size, including a confidence interval for the estimate.

COCOMO II

The Constructive Cost Model (COCOMO) is a software cost and schedule estimating method developed by Barry Boehm in the early 1980s. Boehm developed COCOMO empirically by running a study of 63software development projects and statistically analyzing their results. COCOMO II was developed in the 1990s as an updated version for modern development life cycles, and it is based on a broader set of data. The COCOMO calculation incorporates 15 cost drivers, variables that must be provided as input for a model that is based on the results of those studied projects. These variables cover software, computer, personnel, and project attributes. The output of the model is a set of size and effort estimates that can be developed into a project schedule.

Advantages:

1. It is able to generate repeatable estimations.
2. It is easy to modify input data, refine and customize formulas.
3. It is efficient and able to support a family of estimations or a sensitivity analysis.
4. It is objectively calibrated to previous experience.

Disadvantages:

1. It is unable to deal with exceptional conditions, such as exceptional personnel in any software cost estimating exercises, exceptional teamwork, and an exceptional match between skill-levels and tasks.
2. It is a size based estimation method.
3. Poor sizing inputs and inaccurate cost driver rating will result in inaccurate estimation.
4. Some experience and factors cannot be easily quantified.

THE PLANNING GAME

The Planning Game is the software project planning method from Extreme Programming (XP), a light weight development methodology developed by Kent Beck in the 1990s at Chrysler. It is a method used to manage the negotiation between the engineering team (“Development”) and the stakeholders (“Business”). It gains some emotional distance from the planning process by treating it as a game, where the playing pieces are “user stories” written on index cards and the goal is to assign value to stories and put them into production over time. Unlike Delphi, PROBE, and COCOMO, the Planning Game does not require a documented description of the scope of the project to be estimated. Rather, it is a full planning process that combines estimation with identifying the scope of the project and the tasks required to complete the software.

The game is a meeting that occurs once per iteration, typically once a week. The planning process is divided into two parts:

- **Release Planning:** This is focused on determining what requirements are included in which near-term releases, and when they should be delivered. The customers and developers are both part of this. Release Planning consists of three phases:
  - Exploration Phase: The customer will provide a list of high-value requirements for the system. These will be written down on user story cards.
  - Commitment Phase: Business and developers will commit themselves to the functionality that will be included and the date of the next release.
  - Steering Phase: The plan can be adjusted, new requirements can be added and/or

existing requirements can be changed or removed.

- **Iteration Planning:** This plans the activities and tasks of the developers. In this process the customer is not involved. Iteration Planning also consists of three phases:
- Exploration Phase: Requirement will be translated to different tasks. The tasks are recorded on task cards.
    - Commitment Phase: The tasks will be assigned to the programmers and the time it takes to complete will be estimated.
    - Steering Phase: The tasks are performed and the end result is matched with the original user story.

CONCLUSION

From last many decades a large no. of cost estimation models has been developed. But as it is clear from the above discussion that there is no one method that can give the results with high accuracy. So developer must use the combination of these methods for cost estimation in order to get reliable and accurate results. Moreover the cost must be re-estimated at regular intervals so as to get the correct status of the project.

REFERENCES

- Bernard l. "cost estimation for software development", addision_wesley, 1987
- Shaw, m l.g. " lecture notes on software cost estimation model"
- Liming,“the comparison of the software cost estimating methods” http://www.compapp.dcu.ie/~renaat/ca421/lwu1.html
- Jovan, maksimović, medić, “methods of effort estimation in software engineering “ http://www.tfzr.uns.ac.rs/emc2012/emc2011/files/f%2003.pdf
- Hareton leung, zhang fan, “software cost estimation” ftp://cs.pitt.edu/chang/handbook/42b.pdf
- Bogdan stepien, “software development cost estimation methods and research trends”, http://www.csci.agh.edu.pl/50/1/cs2003-05.pdf

Shruti Jain (MCA):Assistant Professor,Department of Computer Science, Punjab College of Technical Education, Baddowal, Ludhiana ,Punjab