Chapter 4

Literature Review

The purpose of this chapter is to identify and analyze the gap that exists between the publications of software cost estimation methods. The intention is to discuss the advantages and limitations of expert judgment, especially in case of the Wideband Delphi method as compared to formal estimation methods and models. In further sections, the focus will shift from cost estimation to software quality as quality is a very important part of software development and also influences the software’s cost.

4.1 Reviews on Software Cost Estimation Methods

The paper titled “Systematic Review of Software Development Cost Estimation Studies” by Jorgensen and Shepperd [34] is one of the best sources of information on software cost estimation as it provides in-depth details of 76 journals wherein 304 software cost estimation papers published.

This paper classified the papers in different categories based on estimation approach, topic, research method, and experiment data. The classification is regularly updated and available through the online catalog BESTWeb9. This paper was the most valuable one as I found almost all the details related to journals, articles and the researchers under one heading.

Another paper that I found interesting was “Software Development Effort Estimation: Formal Models or Expert Judgment?” By Jorgensen & Boehm[35]. In this paper they have compared two methods i.e. expert estimation and algorithmic approach and have concluded that formal estimation methods are no longer in use. Expert judgment is better in situations where experts with highly specific knowledge are present. They also stated that companies mostly use a combination of different estimation techniques and have generated better results.
In another paper titled “A Review of Surveys on Software Effort Estimation” by Jorgensen and Molokken[36], performance of different software cost estimation techniques has been analyzed. This paper suggests methods that are currently being used in the industry and how accurate are they in predicting the software’s cost. The authors have concluded that expert estimation is the most popular and frequently used method. They said “Project overruns are frequent, but most projects don’t suffer from large overruns. Managers state that accurate estimation is a problem. The reasons for overruns are complex, manifold and not addressed in an appropriate way in many studies” [36].

From the above discussion it is clear that Expert estimation is the most popular and dominant software cost estimation technique prevailing in the market.

In context to this, Jorgensen gave some practical guidelines in one of his papers [37]. According to him, it would be better to separate planning and bidding from effort estimation. He further stated some effective and ineffective combinations of software cost estimation methods. Some effective combinations are top-down & bottom up, analogy and linear regression, expert judgments and formal methods etc. Ineffective combinations are those where cost is estimated with experts having similar experience or same educational background.

Till now the literature reviewed has outlined the different methods cost estimation prevailing in the software industry. It has also listed the advantages and disadvantages of different methods. Then it covered the comparison of parametric and non-parametric approaches wherein non-parametric proved to be better especially expert estimation. After this, literature focused on the importance of hybrid approach. It provided details around the use of this approach in the software industry and level of satisfaction attained.
4.2 Reviews on Software Quality

Software quality, which is another important factor of software development, will now be considered for literature review. There exist various papers on software quality; however, none is able to establish its relation with influencing the cost and effort of a project.

Some of the papers I found are listed as follows:

Quality is a very complicated topic to understand [38]. It cannot be measured directly; instead, different quality attributes like maintainability, reusability, portability, usability, flexibility, security etc. are considered and referred for calculating the software quality value. These values are mostly interdependent and support each other. Before I move further to know how quality influences the effort and cost of the project, it is necessary to understand how to measure the software quality.

The paper titled “Software quality measurement” by Jorgensen [40] stated that there is a need to standardize the software quality measurement as it is always measured indirectly. This paper basically provided the preconditions for software quality measurement. He also stated that without a formula, it is impossible to measure quality as increasing one quality attribute might decrease another.

Another paper on this topic was presented by Boehm et al. [39], wherein he specified the framework for defining software quality and further subdivided it into different quality attributes.

In another paper titled “Body of Knowledge for Software Quality Measurement [41], Schneidewind discussed nine issues related to quality that will help software developers and engineers to understand software quality and how to measure it. He also mentioned the risks associated with software quality. In addition to product quality measurement, he covered the concept of process quality and discussed its inferences.
There is additional literature available on software quality methods, specifically around measurement of quality in terms of products and processes. The various software quality models existing in the market operate on individual quality attributes. Therefore, there is need for a model that combines all quality attributes and generates outcomes that are quality assured.

Ravinder, Kiran and Arvind in one of their paper titled “Measuring Software Reliability: A Fuzzy Model" defines reliability as a non-functional requirement and further divided it into three NFR’s (Non-Functional Requirements) that are availability, failure probability and recoverability. They used fuzzy logic approach to find the reliability of the system with the help of MATLAB simulation [42].

[43] “Measuring reliability of an Aspect Oriented Software" - In this paper, they used 4 NFRs (Maturity, Fault Tolerance, Reliability Compliance and Recoverability) by using Fuzzy Logic approach by taking an assumption for making a library software. As conclusion they developed a library software using Aspect Oriented Approach, which has high level fault tolerance, good recoverability and high reliability compliance standard.

S. W. A. Rizvi and R. A. Khan proposed a maintainability estimation model for object oriented software in design phase [44]. In another study Hayes and Zhao [45], proposed a Maintainability model that categorizes software modules as ‘easy to maintain’ and ‘not easy to maintain’. This model helps the developers to identify modules that are not easy to maintain, before integrating them.

Avadhesh Kumar, Rajesh Kumar and PS Grover proposed a model to evaluate maintainability of Aspect Oriented System [46]. They also proposed A Fuzzy Logic Approach to Measure Complexity of Generic Aspect-Oriented Systems [48]. To make system more reliable Datsun Park, Sungwon Kang and Jihyun Lee proposed a model Design Phase Analysis of Software Qualities Using Aspect-Oriented Programming, which is quite useful if we see in terms of software quality [47].
All the above models take single attribute of quality as input to predict the quality of software. This method, however, is incomplete in itself as there are other factors that also affect the quality of software.