

Type of the Paper (Article)

Effort Estimation Model for Developing Web Applications Based Fuzzy and Practical Models

Dinesh Kumar Sain^{1*}, Jabar H Yusif ²

¹ FCIT, Sohar University Oman 1; dinesh@soharuni.edu.om

² FCIT, Sohar University Oman 2; jyousif@soharuni.edu.om

Abstract: Objective: This paper aims to build an Effort Estimation Model for design, coding and testing Web Applications Based Fuzzy and Practical Models, which will help in optimizing the efforts in software development. Methods/ Analysis: Soft computing approach is adopted and applied in the effort estimation and then compared with practical efforts in the development process with interpreting the historical data available for the existing functionalities. Findings: The effort estimation model presented in this paper focuses on the first level estimates published by Project Managers and the second level estimates presented by Project Leaders or Developers for any new requirement or enhancement for a web application built on 3-tier architecture using Microsoft technologies. The model considers the classification of each task as either Low or Medium or High complexity. These tasks pertain to the lowest level parts in bottom-up estimation. Efforts are estimated for designing, coding and unit testing of these tasks and the efforts are summed up to get the effort estimation for the higher level which is a feature to be implemented. Novelty/Improvement: The paper also discusses about the application of the effort estimation model by taking a new requirement as a case study. The first level estimates calculated using the effort estimation model has a variance of about 25% when compared with the actual effort. This variance is very much acceptable considering the fact that the first level estimates can be tolerable up to 35%. The proposed effort estimation tool would help the project managers to efficiently control the project, manage the resources effectively, and improve the software development process and also trade off analyses among schedule, performance, quality and functionality. Fuzzy logic is used to verify the claims made in efforts estimation. It is proposed a new relation between the number of data and efforts value membership for actual data. And converts it into crisp value in the range [0...1] which helps to classify the complexity of the task and subtask in the design, coding and testing phases.

Keywords: Effort, Estimation, Design, Coding, Unit Testing, Fuzzy model.

1. Introduction

The effort estimation is considered as one of the important activities in software development project management. Several researchers have been discussed and modeled the association between the main factors of software development like size and efforts 1,2,3,4. Nerveless, all these efforts, but still many problems need to solve and still. These problems usually happened in the early phases of a project such as inconsistent, uncertain and unclear data 5,6,7. The enhancement of the effort estimation phase is considered as a vital tool for software development planning and forecasting of the time and cost in developing a software project system. The creation of low price software development effectively is considered as an important feature to produce a competitive software 8,9. The relation between the need for reliable and accurate software in one direction, and the cost of predictions software is a challenging mater. Typically, software effort estimation models can be

classified into algorithmic and non-algorithmic models. The main method of computation in algorithmic models are based on the statistical analysis of historical data and Software Life Cycle Management (SLIM). Besides, the COCOMO and Albrecht's Function Point models [10,11]. They focused on create a precise estimation method for determining the values of main factors like the size of software lines of code (LOC), the complexity of each code, the number of interfaces and user screens. The estimation of any software project, be it effort or cost, has always been a challenging task despite the incredible amount of research that has gone into this activity. Some of the main reasons why estimation have gone haywire are [1] including the lack of understanding of the software project scope, the time pressure to complete the project, tight budgets and the lack of quantitative results. This dissertation attempts to build an effort estimation model based on historical data available. The essence of building a good estimation model lies in the interpretation of historical data to estimate the effort for any future activity.

The main focus of this paper is to build an effort estimation model for design, develop and the unit test of a web application either new or enhancement. This model will estimate based on the effort data available for past releases. The web application considered is a 3-tier application with SQL server constituting the back-end (database), COM components are developed using Visual Basic framework for forming business layer. Lastly web pages are developed using Active Server Pages providing the front end (Graphical User Interface). So, the web application discussed is developed using Microsoft Technologies. The set of activities for any new release or enhancement would comprise of a combination of tasks that span across many activities in past releases [12,13,14].

The aim of this work is set out to answer the following questions.

1. How much time is required for design, code and unit test a new functionality to be added to the existing web application?

2. How will the effort be distributed over time?

To answer these questions, this work sets up an estimation model by interpreting the historical data available for the existing functionalities. The data available is based on the actual effort spent in designing, coding and unit testing of each functionality. However, the challenge is that how each task functionality is implemented. Besides, the task functionality it is must be unique by itself and hence it is not possible to interpolate or extrapolate such data to forecast effort. The solution is to break each of the functionality to a set of tasks, prepare an exhaustive list of all such tasks, calculate the effort required to complete each such task, identify the set of such tasks that need to be performed for any new functionality and then estimate the total effort required to implement the functionality.

The model identifies the effort required to complete design, coding and unit testing for one unit of the task, e.g. adding a user control on a web page, and derives a multiplication factor to estimate the effort required for the completion of multiple units of the same task. The design activity considered involves conceptual design, impact analysis for any new requirement, and detailed design with the pseudo code, mapping diagrams and report layouts wherever required.

2. The Web Application

The web application considered in this work is based on Microsoft technologies. It is built on a 3-tiered architecture as described below.

1. Front End – Provides Graphical User Interface for the application – Built using Active Server Pages (ASP) version 2.0 that are deployed in Microsoft Internet Information Server (IIS) version 4.0.

2. Middle Tier – Constitutes COM components developed using Visual Basic (VB) version 6.0. These components implement the business rules for the application and are deployed in the Microsoft Transaction Server (MTS) version 2.0.

3. Back End – Constitutes the Microsoft SQL Server version 7.0 database that is a repository for data storage.

2.1 SQL Server

The MTS IIS is the browser where ASP pages are rendered. The 3-tier architecture where the ASP pages are rendered on the client's (user) browser as Hyper Text Markup Language (HTML)

pages are considered in the paper. The users interact with the application through the graphical interface produced by the HTML pages. The user requests are posted to the IIS that forwards the request to the database server through the business layer in MTS. The database and the business layer serve the client's requests and results are displayed on the browser 15,16,17.

The web application provides various menus to perform the above operations based on the privileges of the user who logs in to the application. The administrators of the web application set these privileges. For the service provider, the application provides the number of allocations made by users to enable billing of the clients. Also, the web application provides facility to post messages to user groups. The web application is enhanced with a set of new features at least twice in a year. The maintenance of the web application it is to be done once per two months based on the problems reported by the customers.

Need for an Effort Estimation Model

The SDLC phases followed in the implementation of new features are

1. Requirement Analysis
2. Technical Design
3. Coding
4. Unit Testing
5. Integrated Testing
6. System Testing
7. Operators Acceptance Testing
8. User Acceptance Testing
9. Deployment

As there are many phases involved, Project Management becomes very complex activity if the estimates are not accurate 18,19,20. This is because, in the absence of accurate estimates, monitoring the progress of the project becomes difficult, as the actual status could be misleading. Predicting the effort required for testing is simpler compared to that for design and development. This is because the effort for testing is determined based on the number of test cases and test scenarios and the complexity of each of them. In the web application, the testing is basically a black box type wherein input is supplied and the testing is done for validating the actual result with the expected value. In the case of development, effort estimation for the design activity is very challenging. Since, the coding is done using three different tools, then it becomes very complicated to classify portion of coding and corresponding tool. This is because of many factors like depending on the skills of the developer, apart from the business rules, other logic pertaining to data processing can be handled in any of the layers. So, it becomes more important to have an effort estimation model for design and development in order to have better control on the project.

3. The Existing Effort Estimation Model

The existing effort estimation model for design and development establishes a pattern that resembles the set of features to be implemented in the new enhancement then the model compares the patterns implemented in earlier releases and finds out a close match. The effort required to implement the matched pattern(s) in the earlier releases is used to predict the future effort. The patterns are matched based on certain metrics like (the number of controls added, Number of files accessed and Nature of database access).

The web application is classified under various modules like:

1. Graphical User Interface Design like adding, modifying and deleting controls
2. Graphical User Interface Functionality
3. Database schema
4. Generation of Reports
5. Data Import
6. Less/more data centric changes

So, for every new enhancement, based on the requirement analysis, details are worked out in terms of the already existing patterns and the estimation are derived. A data repository is already available, which has the efforts of design and development for a variety of patterns.

3.1 Drawbacks of the model

The drawbacks of this model are that it is difficult to measure the similarity levels between the current requirements and those implemented in the past. When the requirements for a feature consists of a small set of tasks like modifying a data element based on a configuration parameter, the estimates were pretty accurate. On the other hand, if the requirements are called for a large number of tasks, then the estimation is made with this model were highly inaccurate. Also, the effort required to arrive at estimation was very high and hence it was considered unproductive. The model provides the estimation in terms of person days, which does not normally reflect the correct effort in terms of person hours.

4. Proposed Effort Estimation Model - Overview

The proposed estimation model involves a more scientific approach. This also makes use of the data repository available for a variety of patterns. Also, it explores the data in the repository and breaks down the work required for a specific kind of requirement. The model is based on the complexity of each task. The model assumes that the estimator would provide the input whether the task is of low or medium or high complexity. Each level of complexity also has 3 sub-levels of complexity. So, there are nine levels of complexity for any task. For each level of complexity is assigned the effort in terms of person hours for design, coding and unit testing.

The effort involves bottom-up approach where a task is broken to a number of subtasks and depending on whether the subtask could be developed using SQL Server programs or VB programs or ASP program or any combination of these, each subtask is assigned an effort. The efforts of all the subtasks are added to get the total effort required for a task. The proposed effort estimation model focuses on design, coding and unit testing where design deals with methodologies to generate the functionality for the application, coding encompasses the actual content and presentation of the application and unit testing involves designing scenarios and testing a module for its intended functionality. The data for the proposed estimation model is obtained from the existing data for each feature implemented in the application and then defined the patterns in terms of front end activity, middle tier programs and back-end procedures. The total effort for design, coding and unit testing of each pattern is calculated for any number of repeated units by simple arithmetic. The metrics used for collecting data are based on complexity, reusability and size.

4.1 Data Collection

The design corresponding each of the features that were implemented in a major releases for the repository holds the data of programs written in Active Server Pages. The commands implemented in branding example are illustrated in Table 1.

5. The Estimation Process

The estimation process aims to measure the attributes of historical projects to arrive at a bottom-up effort estimation model. Bottom-up estimation begins with the lowest level parts of products or tasks to provide estimates for each. Then these estimates are combined together to arrive at the higher level estimates 21,22.

To briefly explain, the historical projects are studied with respect to the features implemented from the design perspective. Then based on the empirical results available in the repository, the effort spent for implementing each task (e.g. implementing a configurable parameter for tax calculation) for design, coding and unit testing is arrived. It is to be noted that the repository contains only the actual effort data pertaining to design, coding and unit testing for implementing a specific feature. The repository does not contain data at each subtask level.

Table 1: The description of commands implemented in branding example

S.No.	Description
1	Adding a new menu to Branding
2	Adding a sub-menu to Branding
3	Adding a mouse event to a menu
4	Adding a graphic to a menu/submenu
5	Modifying a picture in Branding Header
6	Modifying a link in Branding Header
7	Modifying a link in Branding Footer
8	Navigating the user to a specific page based on the user action
9	Adding a java script function to validate user entries
10	Modifying a java script function to validate user entries
11	Modifying display of GUI content
12	Adding a form element
13	Adding a client function to submit the form
14	Adding a server side function for pagination
15	Removing a form element
16	Writing a server side function to render HTML content
17	Adding a method to call a SQL statement using connection object
18	Adding a method to call a Stored Procedure by passing parameters using a
19	Modifying a method to call a SQL statement using connection object
20	Modifying a method to call a Stored Procedure by passing parameters using a
21	Navigating a record-set and displaying values in a form element
22	Adding a method to build a SQL statement and execute using Connection
23	Populating static values in a form element

The effort spent on each subtask is summed the time sheet of each developer who implemented the feature. So, for a given feature, the subtasks were interpreted and a complexity was assigned to each subtask. Each subtask pertains to design and implementation using either SQLServer programs or VB programs or ASP program or any combination of these. The details are provided in Appendix 1.

As an example, let us consider a feature that was implemented to add search criteria to the web application. The requirement was to search all the cardholders belonging to corporate based on their First Name and Last Name. The subtasks to implement this feature are:

1. Add form elements corresponding to first name and last name in an ASP page. (Technically, this means adding two text boxes – one for the First Name and the other for Last Name)

2. Add a Java Script function to validate the user entries in to these form elements (Technically, this means a new function needs to be written in Java Script in the ASP page to validate for entries in mandatory fields and presence of special characters)

3. Modify a Java Script function submit the form (In the existing web application, a Java Script function already exists to submit forms to the server. This function needs to be modified to include this new ASP page)

4. Add new stored procedure in SQL Server to return the set of cardholders based on the input (i.e. First Name and/or Last Name) – Corresponds to writing a query to retrieve results from two tables

5. Modify a VB method to call the above stored procedure by passing the parameters viz. First Name and Last Name, using a Command Object.

6. Adding a server side function in ASP to display the cardholders (Server side functions are written in VB script in the web application discussed)

Each of these subtasks is assigned a complexity as shown below.

SubTask 1 – Low Complexity.

SubTask 2 – Low Complexity.

SubTask 3 – Medium Complexity.

SubTask 4 – Low Complexity.

SubTask 5 – Low Complexity.

SubTask 6 – Medium Complexity.

These complexities are assigned based on the actual effort. Also, a set of guidelines put forth by the author's organization is also followed wherever the time sheets did not reflect the subtasks correctly. After identifying the subtasks belonging to each category. Low, medium and high, the effort required for each subtask is computed from the time sheet. From Appendix 1, it can be seen that to add a new search criteria, the effort taken works out to 20.5 person hours as detailed below.

SubTask 1 – 1 person hour x (2 form elements) = 2 person hours

SubTask 2 – 4 person hours

SubTask 3 – 3 person hours

SubTask 4 – 1.5 person hours

SubTask 5 – 4 person hours

SubTask 6 – 6 person hours

Total Person Hours = 20.5 (for design, coding and unit testing)

This work addresses the issue of non-availability of the subtask details for every new requirement during a high level design stage by building a list of all possible generic features that could be implemented in the web application and also recommending the level of complexity for every feature in the list. This does not limit the estimators to define the complexity perceived based on specific requirement characteristics. So the estimator can rate a feature as "Medium" complexity even if the recommended complexity is "Low" and vice versa. The list of all possible generic features along with the recommended complexity for SQL, VB and ASP programs as appropriate is given in Appendix 2. The complexity for each of these features has been arrived based on the complexity of subtasks. The basis of assigning the overall complexity for each of these features has been decided after consultation with the team of designers and developers.

6. The Estimation Model

Given the diverse nature of the requirements and the different hardware and software technologies used, the classification of each program unit as Low, Medium and High in terms of complexity is very much appropriate¹¹. The program unit and definition varies for each requirement and hence there arises a need to consider the project characteristics and refine the effort estimates. The proposed effort estimation model is constructed by refining the actual effort for each subtask and considering requirement analysis. Besides, it is considered the requirement of specific characteristics like performance, impact on other modules within the application, impact on external modules, impact on the administration of the web application, formatting changes and browser related issues.

The Appendix 1 gives the effort data for design, coding and unit testing for all the features listed in Appendix 2. The effort data for coding has been given for SQL Server programs, VB programs and ASP programs. For each type of program, the effort data has been given for all the three categories Low, Medium and High. The effort data reflect the estimated effort for one unit of implementation. For example, adding one control to the web page can be considered as one unit of implementation. If multiple units need to be added, then a multiplication factor has been given for each feature. The multiplication factor has been arrived considering the reusability of code after the first unit has been implemented. The basis for arriving at the multiplication factor has been the actual effort data. A comparison was made between subtasks where one subtask for a specific feature implement one unit was compared with another subtask that implemented multiple units. For example, in generating reports, defining the number of fields for one unit of implementation varies from one requirement to

another to decide on the complexity. If the estimator defines the generation of reports with 10 fields as complex, and if the report consists of 20 fields, then it should be treated as two units of implementation. But it is not possible to calculate the multiplication factor for each feature in this way. The historical data does not address all possible scenarios. So the multiplication factors were determined after consultation with experienced developers. This is a classic case of expert judgment. The estimation model has been applied to a new release and the comparison between the estimated effort and the actual effort will discuss later in section 9.

Usually, a high level estimate is required during the high level design of application. Project Manager, estimates the need to be made with the help of functional requirements that are derived from the business requirements. The tolerance allowed on this kind of estimate is about 30 to 35 percent. The proposed effort estimation model is easy to use by Project Managers who are not very much aware of the nitty-gritty of the project in terms of design and development. They need to just identify the feature given, and then assume the complexity recommended and arrive at estimates. This model can also be used by project leaders and developers to define the second level of estimates. The effort estimates could change based on the complexity rating and multiplication factor if there were requirement changes between the first level and second level of estimates. But since the second level of estimate is published after the design phase, the tolerance on the estimates is about 20 to 25 percent.

To demonstrate the estimation of effort using this model for the example discussed (i.e. adding new search criteria to the web application to display all the cardholders based on the First Name and Last Name), let us look in to Appendix 2. For adding new search criteria based on one field (assuming one unit of implementation), the recommended complexity is "Medium". So, looking in to the total effort, the estimated value is 17 person hours for one unit of implementation. The multiplication factor defined is 0.5. So for two units of implementation (since the search is based on First Name and Last Name), the total effort is $17 + (0.5 \times 17) = 25.5$ person hours. This value is higher when compared to the value obtained, which was 20.5 person hours. However, the estimate calculated using effort estimation model is much refined and considers requirement specific characteristics.

6.1 Benefits of the Model

Firstly, the most important benefit for Project Managers using this estimation model would be to control the development process efficiently as the estimates derived would act as benchmarks to track progress and check on the effort variance.

Secondly, Project Managers can plan the work for the resources effectively since they have a reliable set of estimates. For the same reason, Project Managers can address the requirement or project specific factors by redefining the development process necessary checkpoints to nullify any risks and also bring down the complexity of the tasks involved. With a first level of estimate within 30% tolerance, Project Managers can negotiate better deals to fixed bid projects or trade off analyses among schedule, quality, performance and functionality.

7. Soft Computing Paradigms

Soft Computing (SC) is a new computing technique for utilizing real world problems and provides lower cost solutions. It is mainly consisting of the following techniques: neural networks, fuzzy systems and evolutionary computation. The three techniques of soft computing are vary from one another in their function and time scales of operation which they embed a priori knowledge. The Neural network implements in a numeric framework, which used to identify their learning and generalization conditions. Fuzzy systems are implemented in a linguistic framework, which is used to handle linguistic information and then performs approximate reasoning. Nevertheless, the evolutionary computation techniques are powerful methods for searching and optimizing the results. Many researchers all over the world contributed essentially in soft computing to discover solutions of various problems in modern scientific society applications. The significant directions of soft computing applications are implemented and performed into knowledge representation, learning

methods, path planning, control, coordination, and decision making²³. Moreover, the SC can be significantly implemented in the following areas: Biometrics systems, Bioinformatics systems, Biomedical systems, Robotics applications, Vulnerability analysis. Furthermore, SC is performed successfully in many applications such as character recognition, data mining, Natural Language Processing (NLP), Image processing, Machine control, Software engineering, Information management, etc.

Despite of all efforts in using the computation in algorithmic models, but it is not reached to the production of efficient and reliable software models. As a result, there was a need to explore new methods for solving the limitations of algorithmic models. The old methods need to be replaced with non-traditional methods of calculation, such as Parkinson's, and experts estimate and Judgment, Price-to-Win and lastly the machine learning methodologies. The uses of techniques such as neural networks and logic fuzzy are more modern techniques for calculating the estimation models which they work on a few and inaccurate data and produced accurate and reliable results. Fuzzy logic is an expert knowledge based approach with powerful linguistic representation for epitomizing imprecision in input and output data sets for model building ²⁴.

8. Fuzzy systems

Fuzzy systems are suitable for the uncertain or approximate reasoning function that their behaviour can be clarified based on fuzzy rules and it can be attuned by tuning the rules. The fuzzy set methods are appropriate for reasoning linguistic modes of natural to human. It is used the concept of crisp sets. The robustness and flexibility of data is implemented by removing the sharp boundary between members and non-members of a group²⁵. Human experiences and preferences are implemented as Fuzzy logic via membership functions and fuzzy rules. The Fuzzy membership functions have different figures based on the preference and experience. The fuzzy set of a membership function of an input variable is mapping a universe of discourse in the interval [0, 1]. Let X is a non-empty set, and then the membership or containment of X in a fuzzy set is "A" which decides an attribute membership function $\mu_A(x) \in [0, 1]$. It is mathematically expressed as:

$$A = \sum_{i=1}^n \left(\frac{\mu_A(x_i)}{x_i} \right)$$

The resolution of a fuzzy set "A" is defined using α -level set. The crisp set " A_α " is containing all the elements of the universal set U. The crisp set is calculated as:

$A_\alpha = \{x_i \in U \mid \mu_A(x_i) \geq \alpha\}, \alpha \in [0, 1]$. And if $A_\alpha = \{x \in U \mid \mu_A(x) > \alpha\}$ then A_α is called a strong α -cut. The level set of the fuzzy set "A" is a set of all levels $\alpha \in [0, 1]$ with distinct α -cuts is illustrated as:

$$\Lambda_A = \{\alpha \mid \mu_A(x) = \alpha, x \in U\}$$

The support $SA \in [0, 1]$, of a fuzzy set A is described as

$$S_A = \left\{ \sum_{i=1}^n \left(\frac{\mu_A(x_i)}{x_i} \right) : x_i \in X \forall \mu_A(x_i) > 0 \right\},$$

All the constituent elements whose membership values are equal to unity is called the core (CA) of a fuzzy set "A" is defined as: $C_A = \{x_i \in U \mid \mu_A(x_i) = 1\}$, while the bandwidth (BWA) α is expressed as $BW_A = \{x_i \in U \mid \mu_A(x_i) \geq 0.5\}$.

9. Implementation of the Estimation Model

This effort estimation model is used to estimate the effort based new requirements the in vogue. The estimated effort has been compared with the actual effort for three of the features implemented. The description of these three features is described below.

1. Branding Changes – corresponds to Graphical User Interface Design changes
2. Modification to Reports – corresponds to more data centric utility
3. Handling of transaction disputes – corresponds to Graphical User Interface Functionality.

As a case study, let us consider the estimation for branding changes.

The requirements for the branding changes are the following:

1. Add 8 new menu items.
2. Add 12 new 1st level sub menu items.
3. Add 9 new 2nd level sub menu items.
4. Add functionalities for Mouse_Over and Mouse_Out events for each menu and sub menu item.

5. Associate a path with every menu and sub menu item.

6. Add graphics to each menu and sub menu item.

Based on Appendix 2, the estimation for the design effort is as given in Table2. The branding changes involved development is worked only in ASP pages, the estimation was done by taking the values listed for ASP pages in each sub task. The Table 3 illustrates a case study – effort estimation for coding based on practical calculations.

The total effort estimated for coding works out to 204.6 person hours. The actual effort was 189 person hours. The variance works out to about 8.2%. The Table 4 depicts the use case of the calculation effort estimation unit in testing phase. The estimated total effort for unit testing works out to 69.5 person hours. The actual effort was 88 hours. The variance works out to about 26.6%. Based on the above comparison between estimated effort and actual effort, it can be concluded that the effort estimation model can be used to estimate with a variance of about 25%. This variance is very much acceptable for first level estimation.

Table 2: Case Studies – Effort Estimation for Design

S.No.	Description	Complexity	Basic Effort	Multiplication Factor	Calculation	Estimated Effort, Person
1	Add 8 new menu	High	2.0	0.25	$2.0+(0.25 \times 2 \times 7)$	5.5
2	Add 12 new 1st level sub menu items	Medium	2.0	0.25	$2.0+(0.25 \times 2 \times 11)$	7.5
3	Add 9 new 2nd	Medium	2.0	0.25	$2.0+(0.25 \times 2 \times 8)$	6.0
4	Add Mouse Over Event for each	Medium	2.0	0.6	$2.0+(0.6 \times 2 \times 28)$	35.6
5	Add Mouse Out Event foreach item	Medium	2.0	0.6	$2.0+(0.6 \times 2 \times 28)$	35.6
6	Add Graphic to	Medium	1.0	0.3	$1.0+(0.3 \times 1 \times 28)$	9.4
7	Associate	Medium	2.0	0.5	$2.0+(0.5 \times 2 \times 28)$	30.0

Table 3: Case Studies – Effort Estimation for Coding

S.N	Description	Complexity	Basic Effort	Multiplication Factor	Calculation	Estimate Effort,
1	Add 8	High	8.0	0.25	$8.0+(0.25 \times 8 \times 7)$	22.0
2	Add 12 new 1st level	Medium	4.0	0.25	$4.0+(0.25 \times 4 \times 11)$	15.0
3	Add 9 new 2nd level	Medium	4.0	0.25	$4.0+(0.25 \times 4 \times 8)$	12.0
4	Add Mouse Over	Medium	3.0	0.6	$3.0+(0.6 \times 3 \times 28)$	53.4
5	Add Mouse Out	Medium	3.0	0.6	$3.0+(0.6 \times 3 \times 28)$	53.4
6	Add	Medium	2.0	0.3	$2.0+(0.3 \times 2 \times 28)$	18.8
7	Associate path with	Medium	2.0	0.5	$2.0+(0.5 \times 2 \times 28)$	30.0

Table 4: Case study – Effort Estimation for Unit Testing

S.N	Description	Complexity	Basic Effort	Multiplication Factor	Calculation	Estimated Effort, Person
1	Add 8 new menu items	High	1.0	0.25	$1.0 + (0.25 \times 1 \times 7)$	2.75
2	Add 12 new 1st level menu items	Medium	1.0	0.25	$1.0 + (0.25 \times 1 \times 11)$	3.75
3	Add 9 new 2nd level menu items	Medium	1.0	0.25	$1.0 + (0.25 \times 1 \times 8)$	3.0
4	Add Mouse Over	Medium	1.0	0.6	$1.0 + (0.6 \times 1 \times 28)$	17.8
5	Add Mouse Out	Medium	1.0	0.6	$1.0 + (0.6 \times 1 \times 28)$	17.8
6	Add path with each	Medium	1.0	0.3	$1.0 + (0.3 \times 1 \times 28)$	9.4
7	Associate path with each	Medium	1.0	0.5	$1.0 + (0.5 \times 1 \times 28)$	15.0

The practical and estimation calculation shows that the complexity of branding changes is only medium and high. But, this is not reflect the real case of the efforts spend in deferent stages. For example, if the user wants to add 8 new menu items, then the estimated effort is 5.5 (the first row in Table 2), and complexity is high. While if the user wants to add 12 new sub menu items in the first , then the estimated effort is 7.5 (the second row in Table 2), and complexity is medium. This will make unclear understanding of the meaning of complexity values. Now, the estimation of efforts will be calculated based on the Fuzzy model.

The implementation of the fuzzy model is discussed and the case study of the effort estimation for design in Table 1 is implemented as a fuzzy model as depicted in Figure 1. The relations between the number of data in the application & basic estimation efforts for the number of external input (EI), external output (EO), external enquiry (EQ), external interfaces files (EIF) and internal logical files (LIF) as shows in Table 5.

Table 5: Complexity matrix for external input (EI)

FTR (File Type References)	1-5	5-15	>15
0-1	Low	Low	Medium
2	Low	Medium	High
3 or more	Medium	High	High

Usually, any fuzzy system consists of three main phases commonly referred as, Fuzzification, rule Evaluation, and defuzzification. However, Fuzzifier converts the crisp input to a linguistic variable using the membership functions stored in the fuzzy knowledge base. Then using If-Then type fuzzy rules converts the fuzzy input to the fuzzy output. Therefore, the values with 0 or 1 in the file type references can be translated in the meaning of rules for external input (EI) with 0 or 1 file type references as follows:

R1: If the number of data in the range [0...4], then complexity is a low.

R2: If the number of data in the range [5...15], then complexity is a low.

R3: If the number of data is greater 15, then complexity is a medium.

Also, the values with 2 file type references can be translated in the meaning of rules for external input (EI) as follows:

R4: If the number of data in the range [0...4], then complexity is a low.

R5: If the number of data in the range [5...15], then complexity is a medium.

R6: If the number of data is greater 15, then complexity is a high.

Lastly, the values with 3 and more file type references can be translated in the meaning of rules for external input (EI) as follows:

- R7: If the number of data in the range [0...4], then complexity is a medium.
- R8: If the number of data in the range [5...15], then complexity is a high.
- R9: If the number of data is greater 15, then complexity is a high. Defuzzifier converts the fuzzy output of the inference engine to crisp using membership functions analogous to the ones used by the fuzzifier. The membership function must be computed and transferred as a crisp function in the range of [0...1] as depicted in Table 6.

Table 6: The crisp values of each estimated effort

Complexity	Basic estimated effort	Crisp function
3	5.5	0.154494382
2	7.5	0.210674157
2	6	0.168539326
2	35.6	1
2	35.6	1
2	9.4	0.264044944
2	30	0.842696629

Based on the result in the Figure 2, the high complexity can be gained only if any data number reach a value of 1 in crisp function. And the complexity medium is cached if the value of a crisp function in the range of [0.264044944 ...0.842696629]. Finally, the complexity low is acquired if the value of a crisp function in the range of [0... 0.264044944].

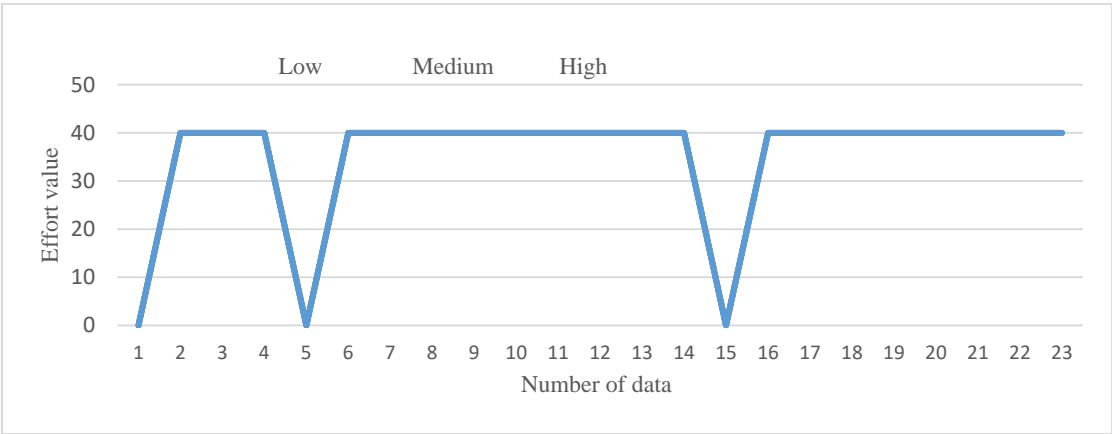


Figure1: the relations between the number of data & efforts value

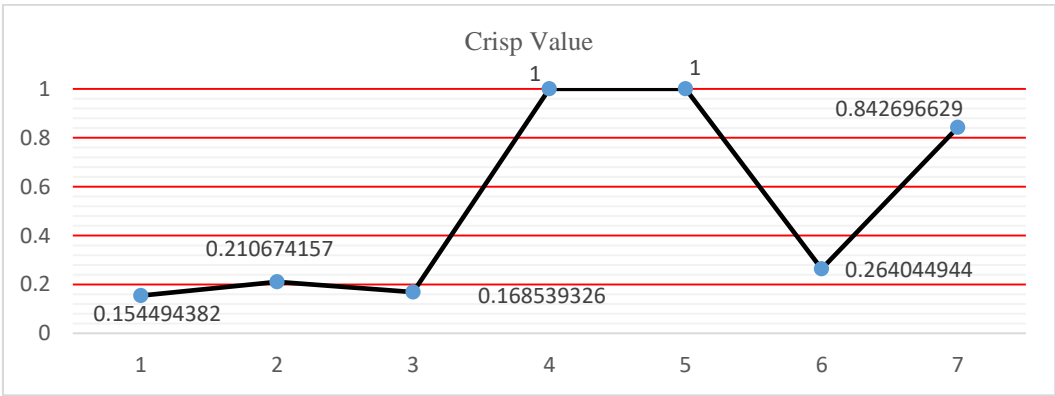


Figure 2 presents the relations between the complexity & crisp values

Figure 3 shows the MSE for training and cross validation for the Fuzzy logic model. It is clearly give evidence that the output of the fuzzy model is closely fit the desired data. The data sets are divided into three categories (60 % as training data sets, 20% as a cross validation data sets and 20% as a testing data sets). The model achieved a final MSE of (0.064708287) in the training phases, and it is achieved minimum MSE of (0.001772667) as summarized in Table 7.

The fuzzy model will be exchanged the rules [R1 to R9], into a new form based on the crisp function values. Therefore, the rules will be as follows:

- R1: If the crisp value is 1, then complexity is a high.
 - R2: If the crisp value in the range [0.264044944 ...0.842696629], then complexity is a high.
 - R3: If the crisp value in the range of [0... 0.264044944], then complexity is a low.
- The same transformation will be implemented for the other rules for coding, design and testing.

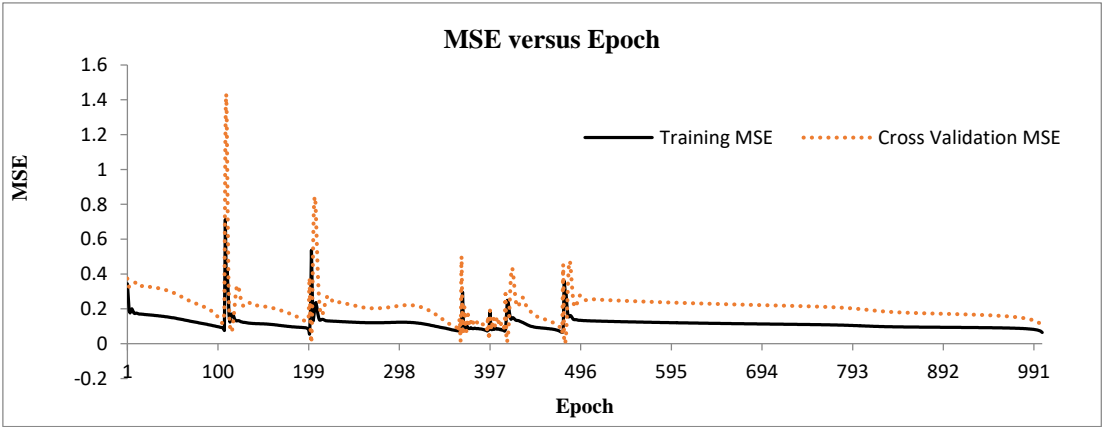


Figure 3: The MSE for training and cross validation for Fuzzy logic model.

Table 7: The results of fuzzy model and Final MSE

Best Networks	Training	Cross Validation
Epoch #	200	478
Minimum MSE	0.054452516	0.001772667
Final MSE	0.064708287	0.114842201

10. Conclusion and Recommendations

The effort estimation model discussed in this paper focuses on estimating the effort of design, coding and unit testing, and the first level estimates have been found to be within a variance of about 25%. The variance would improve for second level estimates and is expected to be about 15%. Estimation must be as closer as possible.

Also the results can be compared with estimates calculated using function points and other methods that use regression. The first level estimates calculated using the effort estimation model has a variance of about 25% when compared with the actual effort. This variance is very much acceptable considering the fact that the first level estimates can be tolerable up to 35%. The proposed effort estimation tool would help the project managers to efficiently control the project, manage the resources effectively, and improve the software development process and also trade off analyses among schedule, performance, quality and functionality. A Soft Computing approach is implemented and results are verified using the Fuzzy Logic models, which helps to compute the complexity accurately and easily. A serious problem in the effort estimation models that some subtasks aren't used before, so they haven't any complexity yet. By using the fuzzy models, it can easily compute the complexity of these tasks and then estimate the effort needed to build the application. Besides, using a fuzzy model helps to compute the complexity of a different type of tasks like design, coding and testing.

References

1. Aloka, S., Singh, P., Rakshit, G. and Srivastava, P.R. (2011) 'Test effort estimation – particle swarm optimization based approach', International Conference on Contemporary Computing IC3, Vol. 168, pp.463–474, India.
2. Thamarai, S. Murugavalli, "An Evolutionary Computation Approach for Project Selection in Analogy based Software Effort Estimation", Indian Journal of Science and Technology, 2016 June, 9(21), Doi no:10.17485/ijst/2016/v9i21/95286
3. S. Malathi, S. Sridhar, "Efficient Estimation of Effort Using Machine-learning Technique for Software Cost", Indian Journal of Science and Technology, 2012 Aug, 5(7), Doi no:10.17485/ijst/2012/v5i8/30539.
4. Ferrucci F, Gravino C, Oliveto R, Sarro F. Using tabu search to estimate software development effort. In Software Process and Product Measurement 2009 Nov 4 (pp. 307-320). Springer Berlin Heidelberg.
5. Shilpee Chamoli, Gil Tenne, Sanjay Bhatia, "Analysing Software Metrics for Accurate Dynamic Defect Prediction Models", Indian Journal of Science and Technology, 2015 Feb, 8(S4), Doi no: 10.17485/ijst/2015/v8iS4/63111.
6. Shaik Mohammad Shahabuddin, Y. Prasanth, "Integration Testing Prior to Unit Testing: A Paradigm Shift in Object Oriented Software Testing of Agile Software Engineering", Indian Journal of Science and Technology, 2016 May, 9(20), Doi no:10.17485/ijst/2016/v9i20/91223.
7. Frei R, Di Marzo Serugendo G. Concepts in complexity engineering. International Journal of Bio-Inspired Computation. 2011 Jan 1;3(2):123-39.
8. Kim Hee-Cheul, Kim Kyung-Soo, "Software Development Cost Model based on NHPP Gompertz Distribution", Indian Journal of Science and Technology, 2015 June, 8(12), Doi no:10.17485/ijst/2015/v8i12/68332.
9. Saini DK, Ahmad M. Return on Investment and Effort Expenditure in the Software Development Environment. International Journal of Applied Information Systems. 2012 Dec;4(7):35-41.
10. Srivastava PR, Varshney A, Nama P, Yang XS. Software test effort estimation: a model based on cuckoo search. International Journal of Bio-Inspired Computation. 2012 Jan 1;4(5):278-85.

11. R. Sathyaraj, S. Prabu, "An Approach for Software Fault Prediction to Measure the Quality of Different Prediction Methodologies using Software Metrics", Indian Journal of Science and Technology, 2015 Dec, 8(35), Doi no: 10.17485/ijst/2015/v8i35/73717.
12. R. Arunachalam, "Innovative Approach of Testing in Event Driven Software (EDS)", Indian Journal of Science and Technology, 2014 June, 7(S5), Doi no: 10.17485/ijst/2014/v7iS5/50295.
13. Saini DK. Prioritizing Test Case Generation for Software Testing in High Level Programming Development Environment. International Journal of Computer Applications. 2013 Jan 1;82(14).
14. O. M. Alshareet, "An Empirical Study to Develop a Decision Support System (DSS) for Measuring the Impact of Quality Measurements over Agile Software Development (ASD)", Indian Journal of Science and Technology, 2015 July, 8(15), Doi no: 10.17485/ijst/2015/v8i15/70774.
15. Saravanan, M. S. Irfan Ahmed, S. Sathya Bama, "Policy Approval Engine - A Framework for Securing Web Applications and Web User ", Indian Journal of Science and Technology, 2016 Jan, 9(4), Doi no: 10.17485/ijst/2016/v9i4/84341.
16. R. Nagarajan, A. Velanganni Joseph, S. Sujatha, "Behavioural Aspects of Software Project Management- In- House Software Development", Indian Journal of Science and Technology, 2015 Feb, 8(S3), Doi no: 10.17485/ijst/2015/v8iS3/58771.
17. Zahra Asheghi Dizaji, Farhad Soleimanian Gharehchopogh, "A Hybrid of Ant Colony Optimization and Chaos Optimization Algorithms Approach for Software Cost Estimation", Indian Journal of Science and Technology, 2015 Jan, 8(2), Doi no: 10.17485/ijst/2015/v8i2/57776.
18. P. Monika, D. Venkatesan, "DI-ANN Clustering Algorithm for Pruning in MLP Neural Network", Indian Journal of Science and Technology, 2015 July, 8(16), Doi no: 10.17485/ijst/2015/v8i16/62540.
19. Cui Z, Fan S, Zeng J, Shi Z. APOA with parabola model for directing orbits of chaotic systems. International Journal of Bio-Inspired Computation. 2013 Jan 1;5(1):67-72.
20. Chintala, A., Pavan, K.V., Harish, V. and Ranjan, S.P., 'Test effort estimation using neural network', Journal of Software Engineering Application (JSEA), 2010. Vol. 3, No. 4, pp.331-340.
21. Farhad Soleimanian Gharehchopogh, Laya Ebrahimi, Isa Maleki, Saman Joudati Gourabi, "A Novel PSO Based Approach with Hybrid of Fuzzy C-means and Learning Automata in Software Cost Estimation", Indian Journal of Science and Technology, 2014 Jan, 7(6), Doi no: 10.17485/ijst/2014/v7i6/46481.
22. Ali Badizadeh, Sohrab Khanmohammadi, "Developing a Fuzzy Model for Assessment and Selection of the best Idea of New Product Development", Indian Journal of Science and Technology, 2011 Dec, 4(12), Doi no: 10.17485/ijst/2011/v4i12/30322.
23. P. Sekar, "On Fuzzy Information Theory", Indian Journal of Science and Technology, 2010 Sep, 3(9), Doi no: 10.17485/ijst/2010/v3i9/29882.
24. Zadeh, L.A. (2001). Applied Soft Computing. Applied Soft Computing 1, 1-2.
25. Uraibi, H.S., Midi, H., A Talib, B. and Yousif, J.H., 2009. Linear regression model selection based on robust bootstrapping technique. American Journal of Applied Sciences, 6(6), pp.1191-1198.

582 **Appendix 1: List of Sub Tasks in ASP Program**

ASP Items	Sub Task Description	Complexity	Effort Person
1	Adding a new menu to Branding	Medium	6
2	Adding a sub-menu to Branding	Medium	4
3	Adding a mouse event to a menu	Medium	3
4	Adding a graphic to a menu/submenu	Medium	1.5
5	Modifying a picture in Branding Header	Medium	1.5
6	Modifying a link in Branding Header	Low	1.5
7	Modifying a link in Branding Footer	Low	1.5
8	Navigating the user to a specific page based on the user	Low	1.5
9	Adding a java script function to validate user entries	Medium	4
10	Modifying a java script function to validate user entries	Low	2
11	Modifying display of GUI content	Low	2.5
12	Adding a form element	Low	1
13	Adding a client function to submit the form	Low	1
14	Adding a server side function for pagination	Medium	4
15	Removing a form element	Low	1
16	Writing a server side function to render HTML content	Medium	6
17	Adding a method to call a SQL statement using	Low	1.5
18	Adding a method to call a Stored Procedure by passing parameters using a Command object	Medium	4
19	Modifying a method to call a SQL statement using	Low	2
20	Modifying a method to call a Stored Procedure by passing parameters using a Command object	Low	3
21	Navigating a record set and displaying values in a form	Medium	3
22	Adding a method to build a SQL statement and execute using Connection object	Medium	4
23	Populating static values in a form element	Low	3

583

584

585

Appendix 2 : List of Features with Recommended Complexity

Description	SQL	VB	ASP
Check for the existence of a configuration parameter	Low	Low	Low
Identify values in the database based on input	Low		
Update values in the database based on input	Medium		
Insert new values in the database based on input	Medium		
Update related values in the database based on the	Medium		
Loop through a set of values and perform updates	High		
Add filters	Medium		
Establish referential integrity among entities	Medium		
Delete value(s) based on input	Low		
Delete value(s) based on input and also establish	Medium		
Write a new field into the file		Med	
Update a field in the file		Med	
Select output based on input/filters	Low		
Select output based on complex logic	High		
Format the output		Med	Med
Add a new field to the entity	Low		
Validate a file		Med	

1	Map source field and destination field		Med	
1	Identify the group to which a set of values belong	Mediu		
2	Identify entities dynamically based on input	High		
2	Generate Reports		Med	
2	Validate business rules	Mediu	Med	Med
2	Validate Data Types		Med	
2	Menu Design - Add a new menu			High
2	Add a new sub-menu			Med
2	Add a mouse event to a menu			Med
2	Add a graphic to a menu/sub- menu			Med
2	Modify Branding Header			Med
2	Modify Branding Footer			Med
3	Read a value from registry		Low	Low
3	Write a value into registry		Low	Low
3	Add a new search criteria		Med	Med
3	Navigate the user to a specific page based on the user			Med
3	Validate user entries			Med
3	Formatting GUI displays			Med
3	Add a new user input field			Med
3	Implement Pagination feature		Low	Med
3	Removing a user input field			Low
3	Displaying results in the form		Med	Med
4	Maintain user state across pages		Med	Med
4	Save user's search criteria	Mediu	Med	Med
4	Adding privileges to user groups	Mediu	Low	Low
4	Computing values based on user input	Mediu	Med	Low
4	Generate Reports in Complex Formats	Mediu	High	
4	Tune Performance	High	Med	Med
4	Login Process	Mediu	High	Med
4	Validate Login Privilege	High	Med	Low
4	Purge Obsolete Data	High	Med	
4	Handle encrypted customer data	Mediu	High	Med
5	Add a new entity	Mediu		
5	Import Data - design a utility	High	High	
5	Print Reports	Mediu	Med	High
5	Add a new menu based on user's privilege	Mediu	Med	High
5	Add a sub-menu based on user's privilege	Mediu	Med	Med
5	Validate user entries based on business rules	Mediu	Med	Med