Effort Estimations Based on Lines of Code and Function Points in Software Project Management

Mr. K Koteswara Rao †,Dr. G.S.V.P Raju††, Mr. T.V.Madhusudhana Rao†††,

M.Sumender Roy†††† Mr. Siruvuru Siva Sankar Sharath †††††

† Sr.Asst. Professor, CSE, GMRIT, RAJAM, Srikakulam, 532127 India
†† Department of CS & ST Andhra University Visakhapatnam, ,AP,Indaia
†††HOD Dept of IT,TPIST, Bobbili,Vizanagaram ,AP,Indaia
†††† Associate professor,CSE.GIET,Rajahmundry,AP,INDIA
††††† 4/4 B.Tech, GVPCOE, Visakhapatnam 530003 India

Summary
The project is a collection of inter related activities that are executed in a certain order. Project management is a discipline of defining and achieving the targets while optimizing the use of resources over the course duration of a project. One of the most important problems that are faced by project manager is to see the project is going to be completed in stipulated time, and to see the company does not suffer from cost and time overruns. For this purpose project manager able to monitor the resources and generate the reports on performance and by providing the accurate forecast of effort estimations. Estimations can be done in two ways; among them one is estimating the size of the project. In the traditional systems most of the software experts can claim the sizing of a project by using LOC (Lines of Code). In this technique there are so many problems, which can be avoided by measuring the total functionality of the project instead of size of the project which leads to the concept called function point analysis (FPA). Function point analysis was proposed to help measure the size of a computerized business information system. It is widely used in actual software development. However it has been reported that since function point counting involves judgment on the part of the counter. In this paper we proposed how the estimations can be done in conventional software project management and modern software management for the large, complex, distributed projects.

Key words:
Effort Estimations , Software Project Management.

Introduction

Project is a collection of interrelated activities that are executed in a certain order. Software engineering is the establishment and use of the sound engineering principles in order to achieve the software that is reliable and economically efficient. Software engineering is a framework that encompasses process, set of methods and array of tools. Project Management is the discipline of defining and achieving targets while optimizing (or just allocating) the use of resources (time, money, people, materials, energy, space, provisions, communication, etc.) over the course of a project (a set of activities of finite duration).

Manuscript received June 5, 2008.
Manuscript revised June 20, 2008.

Project Management Activities

Project management is composed of several different types of activities such as:

1. Planning the work.
2. Assessing and mitigating risk.
3. Estimating resources.
4. Allocating of resources.
5. Organizing the work.
6. Acquiring human and material resources. 7.

Assigning tasks.

8. Directing activities.
9. Controlling project execution.
10. Reporting progress.
11. Analyzing the results based in the facts achieved. 12.

Analysis & Design.

Project control variables are: Time, Cost, Quality, Scope and Risk.

Conventional software project management

Conventional software management practice is mostly sound in theory, but practice is still tied to archaic technology and techniques. Conventional software management performance:

1. Finding and fixing a software problem after delivery costs 100 times more than finding and fixing the problem in early design phases.
2. You can compress software development schedules 25% of nominal, but no more.
3. For every $1 you spend on development, you will spend $2 on maintenance.
4. Software maintenance and development costs are primarily a function of the number of source lines of code.
5. Variation among people account for biggest differences in software productivity.
6. The overall ratio of software to hardware is still growing. Only about 15% of software development effort is devoted to programming.
7. Software system and products typically cost 3 times as much per SLOC as individual software programs.
8. Walkthroughs catch 60% of the errors.
9. 80% of the contribution comes from 20% of the contributions.

Modern Software Project Management

Although the current software management principles are improved on conventional techniques, they still do not emphasize the modern principles on which this project is based. Building on Davis's format, there are top ten principles of modern software management.

1. Architecture first approach.
2. Iterative life cycle process.
3. Component based development.
4. Change management environment.
5. Round trip engineering.
6. Model based notation.
7. Objective Quality control.
8. Demonstration based approach.
9. Evolving levels of details.
10. Configurable process.

Hierarchies of the peers in industry

In industry, we organize the peers into 3 groups:

1. **Higher level**: Higher level people are nothing but who plays the crucial role in administration. In this level, CEO, President, Vice President may exist.
2. **Middle level:** This level contains intermediatery people between the industry and customer. In this Delivery head may exist. For some companies, Vice President is the delivery head.
3. **Lower Level:** This is the level where the people are involving in the project from production to testing phase. In this level, project member, project leader, project manager can exist.

The role of the peers in the project

Customer can interact with the company, then delivery head can interact and he can collect the problem to be implemented from the customer. Then the delivery head can handover the problem and information to the project manager. Project manager can make the estimations for the project i.e., effort, duration, manpower, phase wise effort, phase wise duration, nominal productivity, staffing and personal plan. These estimations can be done by using LOC and FP. If the project manager not able to claim the size of the project then, it is better to go for estimating the total functionality of the project instead of size. To estimate the functionality the parameters required are Unadjusted Function Point Count (UAFP), Total Complexity Factor (TCF).

UAFP = External Input + External Output + External Inquiry + External File + Internal File

TCF = Multiplication of the cost drivers

Functional Point = UAFP + TCF

On the other hand, estimations can be done by using the LOC. Then, COCOMO model is essential.

Stages of COCOMO

SIZE (in terms of KDLOC) is being calculation using the input table of Size Estimates of different modules.

INITIAL EFFORT is calculated using the formula:

$$E_i = a * SIZE\ b$$

Where a and b are constants depending on the project type.

EAF - Effort Adjustment Factor is calculated using the table of cost drivers.

EAF = multiplication of selected cost drivers.

EFFORT for the project is calculated using the INITIAL EFFORT and EAF.

$$EFFORT = El * EAF$$

Phase wise effort can be calculated by using the formula. %

Effort of that phase=a+ (b/c)*total effort, where a =

intermediate% of effort for that phase

b = (medium KDLOC% of that phase-

Intermediate KDLOC% of that phase)

c = medium KDLOC size-intermediate KDLOC size of that phase

Effort of that phase= % effort of that phase/100* total effort.

DURATION which gives the total number of months needed for development of a project is calculated using the formula:

$$D= 2.5*\text{total effort}^{0.38}$$

Duration for each phase is calculated using the following formula.

% Duration of that phase = a+ (b/c)* total duration, where a = intermediate % of duration for that phase

b = (medium KDLOC % of that phase - intermediate KDLOC % of that phase)

c = medium KDLOC size- intermediate KDLOC size of that phase

Duration of that phase= % duration of that phase/100* total duration.

MAN-POWER calculation is done using effort for each phase and duration for each phase.

Staff for each phase = (Effort for each phase)/(Duration for each phase)

After that, the project manager can assign the work to the project leader and he can assign the work to the project member. The completed work can be submitted from the project manager to project leader, project leader to project manager, project manager to delivery head, deliver head can provide the status of the work.

Function Point

Most of software project management experts claim that length is misleading when trying to size the software product. A better way to generate effort estimations and duration estimations from the product is to estimate the functionality rather than the physical size. Function Points are the one of the major technique to the major functionality of the system. This paper explains how the function points can be measured from the Class and Interaction Diagrams of UML.Function Points were originally introduced by Allan Albrecht over 20 years ago. This paper emphasize how the five components can be identified from the design specifications named.

- External Inputs (EI)
- External Outputs (EO)
- External Inquiries (EQ)
- Internal Files (IF)
- External Files (EF)

Here file means a user identifiable group of data, thus not necessary a traditional physical file implemented in computer system. After EI, EO, EF, IF, EQ is weighted then unadjusted function point count can be calculated by summing the all individual counts. After that adjustment factor can be calculated from the 15 cost drivers based on technical and quality requirements.

Diagrams

To understand the UML it is necessary to form the conceptual model of the UML which includes the learning of 3 concepts.

- Building Blocks
- Rules
- Common Mechanisms

Building blocks are the necessary elements which includes.

- Things
- Relationships
- Diagrams

  Things are nothing but abstractions which are first class citizens can be classified

- Structural (Class, Use Case, Interface, Active Class, Component, Simple Collaboration and Deployment)
- Behavioral Things (Interaction and State Machine)
- Grouping Things (Package)
- Annotational Things (Note)

Relationship is a semantic link between any two elements of the above things which can be classified into Association, Dependency, Generalization and Realization. Diagrams are the graphical representation of elements which are classified into 2 types.

1. Structural Diagrams (Static Aspects)
   - Class Diagram
   - Object Diagram
   - Component Diagram
   - Deployment Diagram
2. Behavioral Diagrams (Dynamic Aspects)
   - Use Case Diagram
   - Activity Diagram
   - Interaction Diagram
     - Sequence Diagram
     - Collaborations Diagram
   - State Chart Diagram

Use case diagram:- use case diagram is a brief functionality of the system. It shows the relationship

between actors and use cases. Use case diagram explains about how the system is going to interact with the outside environment. The components are actor, use case, relationship and package.

Actor is someone or something that is interacting with the system. Use cases are nothing but scenarios of the system. If the abstraction level of the diagram is more enough an estimation of the function point can be derived from it. Different use cases must be consistent and describe the behavior of the system. Here first defining the boundary of system of the application which is essential to find whether the file is internal or external. Here boundary exists between actor and the system. Transaction type can also be identified easily if the use cases are defined properly.

External Input (EI) : external input process data or control information that comes from the outside the application boundary. The EI it self is an elementary process. EI means where data flows from an actor to the system.

External Output (EO) : An EO is an elementary process that generates data or control information sent outside the application's boundary. EO means data can flow from system to actor.

External Inquiry (EQ) : An external inquiry is an elementary process made up of an input-output combination that results in data retrieval. The output side contains no derived data. Here, derived data is data that requires processing other than direct retrieval and editing of information from internal logical files and/or external interface files. No internal logical file is maintained during processing. When an actor retrieves data from an internal file but the input processing does not maintain an internal file and the output does not contain the derived data, two phase process is external query.

Internal File (IF) : Internal file means originator.

External File (EF) : External File means destination.

Unified Library Application System.

In ULAS member can login into system that means actor is providing the data to system so it can be counted as External Input. Actor can visualize the reports in Search and Browse phase it can be counted as External Output. The ULAS is maintaining the catalog to authenticate and to respond on operation, it can be treated as Internal File. The actor can have the availability to see the reservation details and the availability details so it can be treated as External File. When member asked for issue the book then the librarian can reservation, dues etc. it can be treated as External Inquiries. It can be enough for a rough estimation of effort to only count the number of transaction types and get an early estimate of the functional size of the designed system. To get more accurate figures, a base use case diagram is not enough. Explanations on different use case on natural language provide more information on attributes of each process. If these are available, the number of external and internal files can be calculated more easily and also the complexity factor of each use case can be extracted from the texts. The extra information can also become available as the design proceeds .so the results of function point calculations can be updated and adjusted when possible. It is a safe to say that the quality of function point calculation based on use cases is highly dependent on the quality of the use cases. The more detailed description of the use cases are the more accurate function point analysis. Frequently confronted problems with the use case include also the fact that one use case can contain several transactions or one transaction can consist of several use cases.

Interaction Diagrams

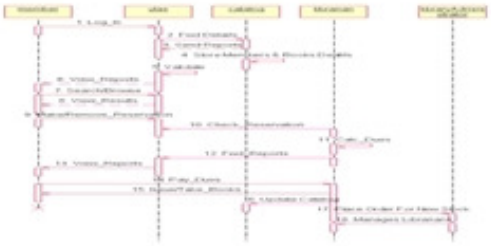
Interaction Diagrams explains about how the sequence of the messages can be exchanged among the objects in order to achieve the piece of functionality of the system. Use case diagrams are behavioral diagrams but they often offer some knowledge of the structure of the system. if we have only behavioral diagrams that describe how the system reacts with the users and internal components. It can be difficult to find all the needed components for function point analysis; on the other hand a mere structural description of a system hides the actual usage of the system. So combination of behavioral and structural ones of the UML sounds promising. Measuring the function points from the interaction diagrams was proposed by Uemura, Kusumoto and Inoue. This method is based on Allan Albrecht function point calculation called IFPUG, which is very similar to the original one. In practice it only defines systematically the step needed to calculate function points and give some static weighting values and accurate defamations for the different components but the basic idea and used components remind the same.

Interaction diagram classified into two types 1.Sequence Diagram 2.Collaboration diagram.

Sequence Diagram

Sequence diagram shows the step by step procedure to accomplish a piece of functionality provided by the system. Sequence diagram is time ordering of messages. The components are:

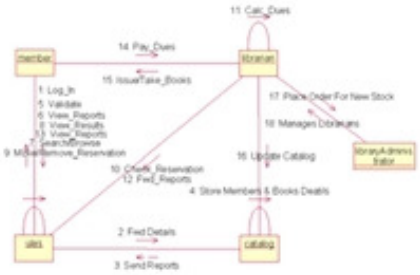
a) Actor object message lifeline
b) Focus of Control Object destructor



Unified Library Application System
Note: Actor also represented as object

Collaboration Diagram

Collaboration diagram displays object interactions around objects and their links to one another**.**



Unified Library Application System

Class diagram

Class diagram shows structure of the software system. The class diagram shows a set of classes, interfaces and relationships.

The components are:

a) Class
b) Relationship:
The forms of relationship are:

1. Association
2. Aggregation
3. Generalization
4. Composition
5. Dependency

Unified Library Application System

Here also as in Use cases the application boundary can be extracted from the sequence diagrams quite easily. Actors are outside the system and other objects are inside the system here data function types can be calculated first. Candidates for the data function are taken from the sequence diagram where each non actor object exchanging data with a non actor object is a candidate. The function types of each of these candidates are determined based on the knowledge we got from the class diagrams.

Internal File: If the candidate has operations that that attributes of other objects in exchanging of data, it is considered to be an internal file. The complexity of the files is determined by counting the number of attributes in the corresponding class and tacking the appropriate values from the table.

From the sequence diagram take each message exchanged by a data function object as a candidate for a transaction function. If the message has no arguments it does not exchange data and can be discarded immediately. The messages are listed as sequences starting from an actor object and ending to the actor or some other object. Each of these sequences is assigned a type based on five patterns and a complexity factor.

External Input : The pattern include a message sent by an actor object to a data function.

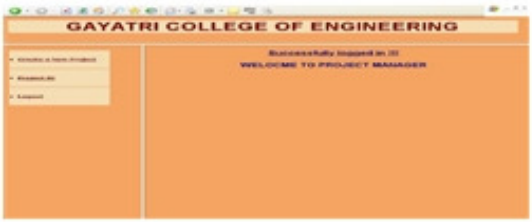
External output : A message sent to an actor object by a data function.

External Inquiries : A message sent to a data function by an actor and a return message and two combinations of these patterns. All of the identified sequences can be constructed from these patterns and given complexity factors based on the number of arguments on the messages and number of involved objects. The success of these types of assignments and complexity measures depend on the usage. The used sequence diagrams must consistent naming systems and all the used arguments must be placed on correct message otherwise systematic approach fails. When the steps described above are completed the final result of the function point calculation can be derived by deciding the technical requirements. The described method is an interesting attempt to automate the calculation process. It has been tested in real life examples and the results have been compared with the function points calculated by experts.

Results

1. Here the project manager can login into system, by entering the user name and password.

2. After the successful login of the Project manager, the welcome and successful logged screen will be displayed as follows.

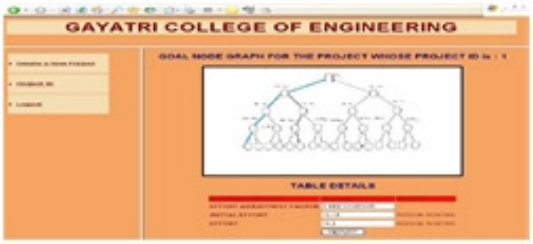


3. Here the project manager can enter the details of the project i.e. project ID, goal identifier, KDLOC of project.

4. This result screen contains details of the project i.e. Pid, KDLOC, and name of the project manager.

5. This result screen shows, how EAF can be calculated, by selecting 15 cost drivers.

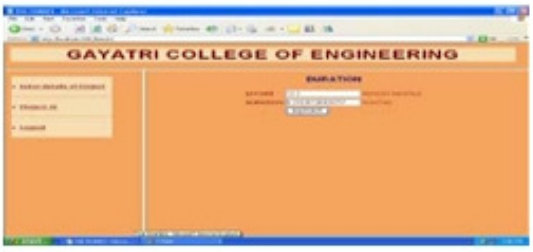
6. This result screen shows the decision tree, Goal node of the project, Bittst-ring, constants a and b. EAF, initial effort estimate, and total effort.



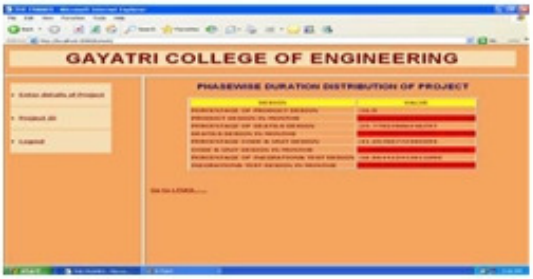
7. This result screen shows the links to the project reports, links are as listed in figure.

8. This result screen shows phase wise effort distribution of the project i.e. percentage of effort for that phase and person months for every phase.

9. This Result screen shows the total duration of the project.

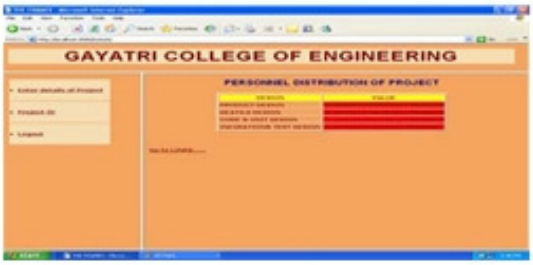


10. This Result screen shows the phase wise distribution of duration of the project.



11. This Result screen shows how nominal productivity of the project can be calculated.

12. This Result screen shows the total number of persons required for every phase of the project.



Conclusion

Complex projects need firm and well defined plans and to present cost and time overruns need effective monitoring and control. This aspect calls for a reliable and accurate forecasts for project estimations like effort Duration, complexity and cost etc. Thus we need an expert system that provides accurate forecasts under imprecision and partial truths. The prototype presented is a fuzzy based expert system for forecasting effort estimation under imprecision. If software experts are efficient to claim the size of the project then it is better to go for COCOMO model to complete the estimation.

References

[1] Pankaj Jalote ," An Integrated Approach to Soft ware Engineering" 2 nd Edition , Narosa Publishing House,2004,Chapter-4 (Planning a Software Project),Pg no.166-170. ISBN - 81-7319-271-5.

[2] Roger S Pressmen, "Software Engineering - a Practitioner's Approch" 6th Eddition Mc Graw Hill international Edition, Pearson education, ISBN 007 - 124083 - 7.

[3] Waman S Jawdekar, "Software Engineering Principles and Practices" Tata Mc graw hill ISBN 0 -07 - 058371 - 4

[4] Walker Royce "Software Project Management - A unified frame work" 2nd Edition, low price Edition, ISBN 81 - 7758 - 378 - 6, pearson education.

[5] Ian somarville, "Software Engineering" 5th Edition low price Edition, International Computer Science Series. [6] Shari, Laurance, Pfleeger, "Software Engineering theory and practies" 2nd Edition, ISBN 81 - 7808 - 4589 - 7, low price edition.

[7] Carlo ghezzi, Mehdi Jazayeri, Dino Mandrioli, "Fundamentals of Software Engineering", PHI, ISBN 81 - 203 - 0865 - 4.

[8] Henrik Legind Larsen - An Approach to Flexible Information Access Systems using Soft Computing -32nd Hawaii International Conference of System Sciences, 1999.

[9] Magnr Jorgesen ,Molokken Ostvold "Reasons for Software Effort Estimation Error: Impact of Respondent Role, Information Collection Approach, and Data Analysis Method " ,IEEE Transactions on Software EngineeringVol. 30, No. 12; December2004, pp. 993-1007.

[10] Pankaj Jalote ," An Integrated Approach to Soft ware Engineering" 2nd Edition , Narosa Publishing House,2004,Chapter-4 (Planning a Software Project),Pg no.166-170. ISBN - 81-7319-271-5.

[11] Zhiping Fan, Jian Ma - An Approach to Multiple Attribute Decision Making Based on Incomplete Information on Alternatives - 32nd Hawaii International Conference of System Sciences, 1999.

[12] Roger S Pressman, "Software Engineering", TMH, New Delhi.

[13] Ian Somerville, "Software Engineering 5th Edition", ICSS, 2005.

[14] Walker A Jawdekar, "Software Engineering Principles and Practices", TMH, New Delhi, 2004 [15] Walker Royce, "Software Project Management", Pearson Education.

Mr. K. Koteswara Rao received the B.Tech degree in CSE from Acharya Nagarjuna University. He received M.Tech degree in CSE from Jawaharlal Nehru Technological University in 2006. At present working as Sr.Assistant Professor in the Dept. of CSE in GMR Institute of Technology, Rajam, Srikakulam. His research interest includes Software Engineering, Object Oriented Systems Development, Project Management, and Data Mining.

Mr. T.V.Madhusudhana Rao received his B.Tech from JNTU Kakinada, M.Tech from JNTU Anantapur. At present he is working as HOD of IT, TPIST, Bobbili. His research interest includes Software Engineering, Data Mining.

Mr. M.Sumender Roy received his B.Tech degree in CSE from Andhra University. He received his M.Tech degree in CSE from JNTU. He worked as Lecturer, Astt.Professor in CSE dept.of SVH College of Engineering, at present he is working as Associate professor in CSE Dept.of GIET.His research areas Include Software engineering, Data Mining, UML, andProject Management.

Mr. G. Sri Sagar Reddy Pursuing the B.Tech in CSE from JNTU. His research interest includes Software Engineering, Data Mining.

Mr. S.S.S Sharath Pursuing the B.Tech in CSE from JNTU. His research interest includes Software Engineering, Data Mining Operating system.

Mr. Vaka Venkat received the B.Tech degree in CSE from Jawaharlal Nehru Technological University in 2006. He is presently pursing M.Tech degree in CSE from Jawaharlal Nehru Technological University. His research interest includes Software Engineering, Data Mining and Software Project Management.

Mr. S.Anil Kumar received his M.Tech degree from Acharya Nagarjuna University in 2008. At present he is working as HOD of IT, GITAS, Bobbili. His research areas are Software Engineering, Data Mining.