Journal of Computer Science 2 (3): 276-282, 2006
ISSN 1549-3636

Software Cost Estimation Model Based on Integration of Multi-agent and Case-Based Reasoning

Hasan Al-Sakran
Information Technology College, MIS Department, Yarmouk University, Irbed, Jordan

Abstract: Accurate software cost estimation is a vital task that affects the firm's software investment decisions before committing required resources to that project or bidding for a contract. This study proposes an improved Case-Based Reasoning (CBR) approach integrated with multi-agent technology to retrieve similar projects from multi-organizational distributed datasets. The study explores the possibility of building a software cost estimation model by collecting software cost data from distributed predefined project cost databases. The model applying CBR method to find similar projects in historical data derived from measured software projects developed by different organizations.

Key words: Mobile agent, COCOMO, CBR, magnitude of relative error

INTRODUCTION

Software becomes increasingly expensive to develop and is a major cost factor in any information system budget. The accuracy of estimation of software project cost has a direct and significant impact on the quality of the firm's software investment decisions. Management carefully considers costs and benefits of software before committing the required resources to that project or bidding for a contract. Accurately estimating a new software project is still a goal of every project manager. Unfortunately such preliminary estimation is difficult to measure because there is little information about the project at an early stage.

Over- or under-estimation of software costs may result in costly errors such as projects are rejected as too expensive; projects may omit important features; projects are abandoned. Accurate project estimation can reduce these unnecessary costs and increase the organization's efficiency and effectiveness.

Most of today's software cost estimation models are built on using data from projects of single organization. Using such data has well known benefits such as ease of understanding and controlling of collected data. But different researchers have reported contradictory results using different software cost estimation modeling techniques. Myrtveit and Srensrud[1] state that it is still difficult to generalize many of the obtain results. This is due to the characteristics of the datasets being used and datasets' small size. The fact, that many studies rely on using organization-specific datasets, makes the results more biased because the data at hand are specific to a given organization. Briand[2] found that cost estimation models using single-company dataset do not perform significantly better than models using multi-companies dataset. Characteristics of the datasets being used play a major role.

It has been established that relying on organization-specific datasets leads to poor software cost predictions due to the following problems[3, 4]:

* It's too expensive to collect data on previous projects from single organization.
* Information about older projects may no longer be valid or appropriate due to the new technologies that organization is using.
* It's difficult to ensure consistency of the collected data.

Massively collected data about software projects present an interesting aspect of software cost estimation. One purpose of this research is to address the issues of the dataset characteristics and usage of a large number of datasets. This study is based on selecting and using a large number of datasets coming from distributed software project databases of different organizations of comparable domains. This approach supports the fast construction of cost estimation models, it also helps organizations, who do not have their own data or expertise, to access external data come from similar types of projects to build their own cost estimation model; provides larger and up to date project datasets.

Recent research has demonstrated the potential of the use of Artificial Intelligence (AI) methodology to estimate the cost of software to provide both consistency and more accurate estimates. This study presents an alternative approach of software cost estimation based upon an AI methodology, namely Case Based Reasoning (CBR), similar to the one applied by Shepperd *et al*.[5], combined with mobile agent technology. The CBR approach makes use of previous experience to solve newly encountered problems. The past experience is recorded in a case

Corresponding Author: Hasan Al-Sakran, Information Technology College/MIS Department, Yarmouk University/Irbed /Jordan, Tel: 962-2-7211111, Fax: 962-2-7274725

base database. When a new problem emerges, the CBR system retrieves projects from the database to find similar cases to the current problem and the closest match is modified to fit the new problem. The modified case also will be stored in the case base as a learned case to save the experience and can be reused in the future.

In CBR problem-solving is seen as a process, which involves the retrieval of similar prior cases from case bases using mobile agent methodology and the adaptation of retrieved cases' solutions to fit the new problem's requirements.

Related work: Several different directions in the research on the estimation of software development cost emerged during the last two decades. Some potential solutions of the above problem have been developed based on algorithmic models (e.g. Constructive Cost Model (COCOMO, COCOMO II[6, 7], Function Points, Price-to-win and SLIM[8]), expert judgment and estimation by analogy. Most of the algorithmic software estimation models are based on analytical methods and derived from the statistical or numerical analysis of historical projects data.

The general form of equation used by COCOMO and Function Points methods can be represented as:

$$E = xS^{y},$$

Where, E is effort, S is size measured as number of lines of code or function points, x is a productivity parameter and y is economics of scale parameter. COCOMO model provides three equations according to the project development mode (embedded, semi-detached and organic). Their parameters need to be adjusted to local circumstances.

The released version of COCOMO II has been used to conduct empirical analysis of the model. The general form of equation used by COCOMO II is:

$$E = X \times [S]^{B+\sum SF} \times \prod EM,$$

Where, X is baseline multiplicative constant, B is baseline exponential constant, SF are scale factors (understanding product objectives, flexibility, team coherence, etc.), EM are effort multipliers (software reliability, database size, reusability, complexity, etc.)

None of mentioned above methods have been shown to be convincing or consistent in solving the problem. Some of these algorithmic methods may lead to relative errors as high as 600%[9]. The prediction accuracy is measured based on standard metrics such as Magnitude of Relative Error (MRE). MRE is defined as:

$$MRE = \frac{Effort_{actual} - Effort_{estimated}}{Effort_{actual}}$$

If the value of MRE is large, then the model over-estimates the cost, while a large negative value would indicate, that the model under-estimates the software cost.

Researchers have begun to turn their attention to non-algorithmic methods and in particular, to a set of approaches based on expert judgment, rule based, neural networks and case based reasoning.

Expert judgment methods rely on the use of human expertise to estimate software cost[10]. These techniques are useful in the absence of quantified empirical data and are based on prior knowledge of experts in the field. Instead of starting estimating software cost from scratch, software managers rely on their past experiences and understanding of the problem. They attempt to find past cases similar to the new project and to adapt old estimations to fit the new situation. However, this human-based approach lacks a consistent and systematic procedure for cost estimation and as a result, might lead to over- or under-estimation of the cost of the software project. The major drawback of this method is that an estimate is only as good as the expert's opinion.

The rule based systems can be used for estimation when no further rules are fired up from known or new facts. This technique has been adopted from the Artificial Intelligence domain where a known fact fires up rules, which in turn may assert new facts. Kellner and etc. Kellner, Madachy and Raffo [11] developed a rule based system to estimate the cost of software.

In the last decade, significant effort has been put into the development of software estimation models using neural networks[12]. Neural networks are based on the principle of learning from example; no prior information is specified. Neural network estimation models must be trained by providing them with historical project data input values (project size, complexity, skill levels, etc) and automatically adjusting their algorithmic parameter values until it is very good at predicting results for the training data set. These models suffer from the same kinds of statistical problems with the training data as the algorithmic techniques. Very large data sets are needed to accurately train neural networks.

Estimation by analogy based on the comparison of the software under consideration with similar projects.

There is no single best software cost estimation model, but CBR method is rated among the best methods in a variety of circumstances[13]. Experiment showed that CBR approach provides better accuracy than algorithmic methods. CBR systems deal only with those problems that occur in practice, while algorithmic system must handle all possible problems. CBR solutions are derived from form of reasoning which close to the human problem solving as opposed to rule based or neural nets. CBR can operate in circumstances where it is not possible to generate an algorithmic

model (no statistically significant relationships could be found).

Case based reasoning: Case Based Reasoning (CBR) has been attracting much attention recently as a paradigm with a wide variety of applications. In this study, issues related to construction of a cost estimation model and composition of a case, where subcases are distributed across different distributed databases, are discussed.

CBR is an AI methodology combined with database of cases related to the topics under consideration for re-using past experience. In this approach, a reasoner tries to remember previous cases similar to the current one and uses them to solve the current problem. The CBR systems store therefore numerous cases related to the matters considered. Often the past experiences provide important clues or direct answers to the current problem.

CBR technique was described by Aamodt and Plaza[14] as combination of the following four processes and shown in Fig. 1.

* Retrieve previously experienced case or cases related to the current problem.
* Re-use this or these case(s) in one way or another.
* Revise the solution based on re-using previous cases.
* Retain the new solution (as a new case) by adding it into the existing case-based database. In such a way, a CBR system will gradually grow larger and become a precious resource.

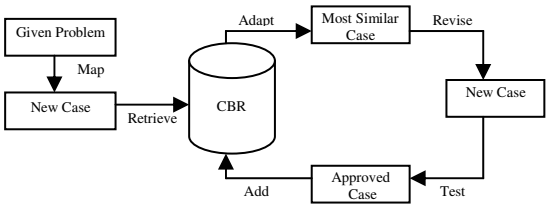

Fig.1: The CBR cycle

Case-based reasoning has several advantages:

* Many early studies showed that CBR presented better prediction accuracy than other models.
* CBR method reflects the same method that human experts use when making estimates by applying analogical reasoning.
* CBR can handle both quantitative and qualitative data
* CBR systems can use existing solution and adapt it to the current situation.
* CBR systems can be implemented very quickly.
* CBR is simple and flexible, compared to algorithmic models.
* To add new knowledge to CBR system, a user only needs to add new cases to the system.
* CBR can effectively support all the steps in the software cost estimation process from storing past cases, retrieving similar cases to adapting the retrieved case for the new project.
* CBR approach takes advantages of expert prior knowledge
* CBR systems can handle failed cases. (Identify potentially high risk situations.

Recent research by Shepperd's group[5] has created a new automated approach to use CBR in software cost estimation. The approach is very successful in providing accurate estimates.

In order to find a case from a large number of cases, the similarity of cases should be analyzed. The establishing similarity of cases is the basis of CBR and case searching. Similarity of cases is influenced by a set of attributes which make the case different from others. These attributes are the key attributes of a case. The cases which have one or more similar key attributes are similar. Every key attribute of cases represents the cases from a specific perspective. Project size, target platform, quality of system requirements are some of the attributes that can act as key attributes. One of the more likely used key attributes is the project size which represents the number of lines of code the project will have. It can be estimated using different techniques such as Genetic Programming and Neural Networks[15]. Some attributes, such as development environment, application type, business area type and others, can act as sub-key attributes.

Case searching model is used to compare and filter the cases from the case base to find similar cases. The case searching model is based on the key and sub-key attributes. To make the case searching model more effective, case index reflecting the main feature of the cases is build. This index is recommended especially when the volume of the case base is large.

Mobile agent: Mobile agents can be defined as autonomous, problem-solving computational entities capable of effectively performing operations in dynamic unpredictable environments. Such environments are known as multi-agent systems. Agents interact and maybe cooperate with other agents. They are capable of exercising control over their actions and interactions. Using mobile agent technology solves the problem of heterogeneity of networks, low bandwidth of communication channels, reduces network traffic by processing data locally instead of transmitting the data over a network. It could accelerate development with agent components and enhance modularity, reusability, flexibility and reliability.

A mobile agent consists of two different parts: the code itself, which composes of the instructions that

define the behavior of the agent and its intelligence and the current state of execution of the agent. At least three major requirements have to be fulfilled for a mobile agent to perform its job. They are common execution language across heterogeneous networks, for example Java; transference of agents across networks through a communication mechanism, for example MAP, TCP/IP, HTTP, or SMTP; protection of agents against hostile server and agent server from malicious agent. Agent may protect their data and information by using encryption/decryption techniques.

A multi-agent system is composed of intelligent agents working towards finding most similar cases. Agents access case bases to retrieve the best matching case. In such a system, each of the agents may not be individually capable of finding the best similar case. Each agent may retrieve the best local cases, which, when assembled, may not result in the best overall case in terms of global measures. But cooperation among them may lead to achievement of the final goals of finding the most similar case or cases. That means the cost prediction of a project does not just rely on few projects stored locally, but affected by larger size of data (distributed datasets).

The main characteristics of Intelligent Agents within CBR environment are:

* Autonomy: the ability of agents to make independent decision;
* Ability to autonomously learn from experience;
* Goal-driven: that is the provision of detailed knowledge so that goals can be achieved;
* Mobility: it allows the routing of agents through a distributed system;
* Reactivity: reacting to changes in the environment;
* Ability to cooperate: a group of agents work together to achieve a common goal;
* Ability to communicate: the agents must to be able to communicate with other agents and/or user.

Agents require knowledge of the current situation, skills to accomplish tasks and make decisions on how to act.

Each client agent will search in a local case base. And each one is associated with a set of constraints representing the requirements of the software cost estimation model. Client agents use a centralized search mechanism to find an optimal or partially optimal projects to a given problem instance. Candidate projects are stored at the data structure of the agent.

The client agents can be simply divided into three types of agents:

* Interface agents: to communicate with the client
* Mobile information agents: to collect information from distributed information resources.
* Task agents: to solve the problem by selecting the best solution from the accumulated information collected by the mobile information agents.

Adaptation of CBR and mobile agent approaches to cost estimation model: The complementary properties of CBR and mobile agent can be advantageously combined to solve the software cost estimation problem, where any single technique fails to provide a satisfactory solution. The mobile agent can be effective in addressing the problem of getting data from different companies. Within this approach, CBR software cost estimation process consists of the following components:

* Formal case representation: past cases should be classified by their attributes and focus on a specific group of cases relevant to the current situation.
* Identification of project attributes for which an estimate is required. These attributes are used as basis for finding similar past projects of known costs.
* Hierarchical case indexing: construction of an efficient case indexing technique to reduce search time for retrieving similar cases. The closeness between a past case and a new one is assessed based on similarity metric for accurate case matching.
* Knowledge - based cost estimate adaptation: search for the relevant knowledge in case and if there is no complete match between the retrieved case(s) and the new one, revision of the existing solution to fit new problem.

The most important software cost factors (attributes) to be considered in CBR method are:

* Project size (the size of source code measured in number of lines of codes, number of HTML pages, or functions points)
* Organization type (Manufacturing, banking, services, administration);
* Target platform (mainframe, Network, PC, etc.);
* Quality of system requirements;
* Development type (new development, redevelopment, enhancement);
* Business area type (engineering, sales, legal, inventory);
* Application type (transaction system, office automation, management information system, executive information system);
* Project security (need for security);
* Complexity of the software;
* Staff experience, availability and skills;
* Development environment;
* Others (the volume of documentation, the number of developers, the number of different files created, the number of bugs reported and so on).

Case representation scheme is dependent on the case size and the complexity of the attributes describing the case. A case of the software cost estimation system

consists mainly of three parts: the description part, solution parts and the relationship part. The description part contains the attributes values describing the behaviors of the case, while the solution part contains the solutions. The relationship part describes the relationship information among cases. Multiple cases can be use to represent a single problem.

When the number of cases in case-base is very large, it is important to formulate indexing technique that helps locate similar cases close to each other efficiently. David W. Patterson and others in[16] propose two efficient indexing schemes designed for use in CBR systems. The first one is based on a matrix of cases indexed by their attribute values. The second one is an extension of the first one by combining the matrix with an additional tree-like indexing structure. The strength of these techniques lies in its ability to improve retrieval efficiency over time by reusing previously encountered solutions.

CBR solves the software cost estimation problem in the following way. The attributes or features of the current problem (project) are identified. Then the current problem is matched against the cases (projects) in the case base (using the most important attributes) and most similar cases (with known cost) are ranked. The most similar case from these ranked cases is retrieved. Searching for similar case is not only by the features in the description part but also by the relation among cases.

If the retrieved case completely matches the current problem, it is used to suggest a solution, which is reused and tested for success. If partial match occurs, then the proposed solution is revised and adapted to fit new needs[17]. Retrieving a case starts with identifying a set of relevant descriptors (cost factors), such as software size (number of lines of code), function points, security needs, use of software tools, etc. and ends when a best matching case has been selected. The final solution becomes a new case in the case base library.

The degree of similarity in CBR is assessed by means of a matching function such as the Nearest Neighbor (NN) matching function[18]:

$$Similarity \quad (N, P) = \frac{\sum_{i=1}^{n} f(N_a, P_a) * w_a}{\sum_{i=1}^{n} w_a},$$

Where N is the new case (new project), P is the previous case (previous project), n is the number of features in each case, i is an individual feature from 1 to n, f is a match function for attribute a in cases N and P and w_a is the weight of the a-th attribute which reflects the relative importance of that attribute.

It is possible that more than one case will have the same value of similarity coefficient or the values of similarity coefficients for different cases can be very close. To select the most suitable case from these candidate cases, the system, through the task agent, will suggest the best case to choose.

The overall framework of the proposed system is presented in Fig. 2. It is composed of three different major components: front end user machine, back end server and the software cost estimation servers on the web. The system has a number of agents. Each agent is designed to represent a specific functional unit. This requires three different agent types, one mobile and two static (interface agent, task agent and mobile information agent).

The client at the front end user machine interacts with the system through a web browser. The back end server has a CBR database storing the previous projects, task manager agent and mobile information agent. The mobile information agent will visit the software cost estimation servers on the web. Each time a client conduct a search, searching criteria will be generated at the back end server and sent by task agent as a data, stored in the mobile information agent, into the web. The mobile information agent will roam the web searching for the required information based on the given criteria. When information is found, the mobile information agent will send it back to the task agent at the back end server where it will be filtered and then presented to the user. The information mobile agent searches case base for the most similar project according to the similarity metric and uses it as a candidate project.

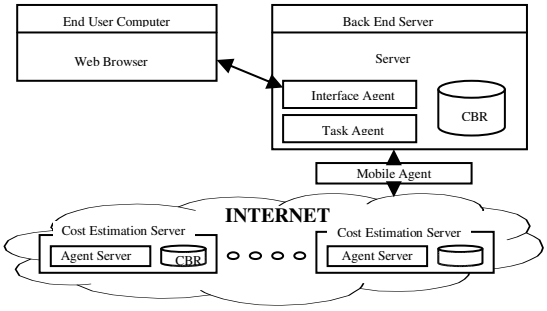

Fig. 2: Architecture of the software cost estimation model

The process of estimation of software cost by CBR and multi-agent consists of the following steps:

After the client communicates with the system through interface agent, each client agents should execute the following algorithms:

Each mobile information agent executes:

Do local retrieval: Select projects of the same types, similar application domains, size, etc. (to be defined by the client) from CBR database.

The task agent executes:

* Receive candidate projects from each mobile information agent;
* Merge candidate projects;
* Choose best project(s).

If a project is retrieved, then the attributes that do not match the attributes of the current project will be grouped as a new project and these unmatched attributes are used to do the retrieval again. This process will be repeated until all attributes are collected and the task manager will merge the collected projects. If there are many projects which have the same amount of attributes, the one which is the most similar (the nearest neighbor)[19] to the current project will be selected by the task agent.

A prototype of the software cost estimation system is implemented using Java so that the system can run on heterogeneous platforms and a CBR inference engine. The implementation consists of the server side and the client side connected through the Internet. The server side consists of the server agents and CBR database. The client side consists of a browser that has support for XML and Java applets. Internet information server and servlets are used for the web servers (Java Servlet Class running in server). MS-SQL is used for database programming. Communication between agents established through Java Agent Development. The software agents communicate with each other in XML messages.

Request for similar projects is constructed at the client interface agent. The main attributes of the new project are entered. The task agent of the back end server will do data analysis for this request though conducting case identification using CBR database. The system is connected to the database by JDBC to access the data for initial reference of similar cases. The mobile information agents will carry the main attributes and their values and then search for similar projects in different web servers. The response results will be forwarded back to the task agent to choose the best matching project. The final result will be presented to the client though his agent browser.

CONCLUSION

Software cost estimation is an important and hard management task. This is due to the lack of information on making decisions in the early phases of the project development.

In this study, a new hybrid software cost estimation model, which integrates case-based reasoning and multi-agent technology, has been presented. The study described the application of cased-based reasoning to estimating the cost for developing software project using multi-organization databases integrated with mobile agent technology. The major property of CBR is saving the previous experience into case base and re-use past solved problems in order to propose solutions to new problems later. The experience of a solved problem can be stored into the case base.

Large collected software cost data from different sources present an interesting aspect of cost estimation model, which may behave better than models developed on projects coming from single database. The proposed system may be used to produce estimates for new projects by software organizations that do not have historical projects cost data or just starting up their software business.

Future work primarily involves conducting experiments on sensitive empirical data coming from different sources using the proposed integrated approach of CBR and multi-agent techniques.

REFERENCES

1. Myrtveit and E. Srensrud, 1999. A controlled experiment to assess the benefits of estimating with analogy and regression models. IEEE Trans. Software Engg., 25: 510-525.
2. Braind, L., E.L.K. Emam and K. Maxwell, 1999. An assessment and comparison of common software cost estimation modeling techniques. Intl. Conf. Software Engg., Los Angeles, CA, pp: 313-322.
3. Briand, L.C., T. Langley and I. Wieczorek, 2000. A replicated assessment of common software cost estimation techniques. Proc. 22nd Intl. Conf. Software Engg, ICSE, pp: 377-386.
4. Mendes, E. and B. Kitchenham, 2004. Further comparison of cross-company and within-company effort estimation models for web applications. Proc. 10th Intl. Symp. Software Metrics (METRICS'04), pp: 348-357.
5. Shepperd, M. and C. Schofield, 1997. Estimating software project effort using analogies. IEEE Trans. Software Engg., 23: 12.
6. Boehm, B. and E. Horowitz *et al.,* 2000. Software Cost Estimation with COCOMO II. Prentice-Hall.
7. Clark, B., S. Chulani and B. Boehm, 1998. Calibrating the COCOMO II Post-Architecture Model. Proc. Intl. Conf. Software Engg., pp: 477-480.
8. Chulani, S., B. Boehm and B. Steece, 1999. Bayesian analysis of empirical software engineering cost models. IEEE Trans. Software Engg., 25: 573-583.
9. Kemerer, C., 1987. An empirical validation of software cost estimation models. Commun. ACM, pp: 416-429.
10. Host, M. and C. Wohlin, 1998. Experimental study of individual subjective effort estimation and combinations of estimates. Proc. Intl. Conf. Software Engg., pp: 332-339.
11. Kellner, Madachy and Raffo, 1999. Software process modeling and simulation: Why, what, how. J. Systems and Software, 46: 91-105.

12. Gray, A. and S. MacDonell, 1997. A comparison of techniques for developing predictive models of software metrics. Information and Software Technol., 39: 425-437.
13. Ruhe, M, R. Jeffery and I. Wieczorek, 2003. Cost estimation for web applications. Intl. Conf. Software Engg., Washington, DC, USA., pp: 285-294.
14. Aamodt, A. and E. Plaza, 1994. Case-based reasoning: Foundational issues, methodological variations and system approaches. AI Communications, 7: 39-59.
15. Regolin, E. and G. de Souza *et al.,* 2003. Exploring machine learning techniques for software size estimation. Proc. XXIII Intl. Conf. Chilean Computer Science Society, IEEE, pp: 130-136.
16. Patterson, D.W., M. Galushka and Niall Rooney, 2005. Characterization of a novel indexing technique for case-based reasoning. Artif.l Intell. Rev., 23: 359-393.
17. Mendes, E., S. Counsell and N. Mosley, 2001. Towards the prediction of development effort for hypermedia applications. Proc. ACM Hypertext'01 Conf., Denmark, pp: 249-258.
18. Auer, M. and S. Biffi, 2004. Increasing the accuracy and reliability of analogy-based cost estimation with extensive project feature dimension weighting. Proc. Intl. Symp. Empirical Software Engg., pp: 147-155.
19. El-Emam, L.C.K., D. Surmann, I. Wieczorek and K.D. Maxwell, 1999. An assessment and comparison of common cost estimation modeling techniques. Proc. ICSE 1999, Los Angeles, USA, pp: 313-322.