IJCSC International Journal of Computer Science & Communication Vol. 1, No. 2, July-December 2010, pp. 419-422

# Effort Estimation Based on Complexity and Size of Relational Database System

Samaresh Mishra[1], Krushna Chandra Tripathy[2] & Manoj Kumar Mishra[3]

[1]School of Computer Engineering, KIIT University, Bhubaneswar, India
[2]Department of Computer Science & Engineering, S'O'A University, Bhubaneswar, India
[3]Department of Information Technology, S'O'A University, Bhubaneswar, India
Email: [1]samaresh2@yahoo.com, [2]kct9_kiit@yahoo.co.in, [3]mkmishra_iter@yahoo.com

ABSTRACT

Software cost estimation is an important activity of software project management. Database plays a very important role in data centric software. In this paper a model for estimating the effort of database development based on complexity and size of the database of relational model has been proposed. The proposed work also includes the new metrics on estimation of complexity and size of the database part of data centric software. The metrics has been validated with student projects.

*Keyword:* Complexity Estimation, Size Estimation, Effort Estimation, Relational Database, ER Diagram

## 1. INTRODUCTION

Software project estimation is an important and complex activity and requires proper planning and examination of different factors affecting in a software developing process. It is vital for the effective control and management of the whole software development process [3] [6] [7]. Software cost estimation basically means estimation of effort, time of development and this effort depends on the size of software and its level of complexity. More is the size of software means more effort is required for development. Similarly, more is the level of complexity means more effort is required for development. In addition to size and complexity, many other factors contribute equally for estimating effort, like development environment, expertise of development team and availability of tools [1]. Again, the cost of estimation is required to be done in the early phase of development. Database part of data centric software plays a very important role compared to any other type of database. Here emphasis is given to database planning, analysis, documentation and their development. The success of data centric software design and development depends on how well one has understood the data requirements, the relationships that exists among them and the various constraints applicable to database. Relational database is more popular in business application because of its rich capability of specifying different constraints. The development process of RDBS (Relational Database System) requires the creation of ER diagram (Entity Relationship diagram) during analysis phase followed by translation to relations and then the normalization activity. The total complexity of the relational database of software can be done at the following stages [2]:

a. At conceptual modeling, i.e. at ER modeling.
b. At logical modeling, when the relations are created out of ER modeling.
c. At physical modeling, i.e. at the implementation level of database.

Since the logical design is derived out of conceptual design, therefore we have considered the estimation process at logical design level. Here the practitioners can identify the relations, constraints on different attributes of those relations, the required triggers, views and data entry forms.

## 2. MOTIVATION

It has been observed from many researches that size of software plays an important role among many other product properties like complexity, cohesion and coupling etc. as cost factors [5]. Here the emphasis can be given on the role of complexity in determining the effort of database development. Again, as software grew in size, it becomes very much necessary to determine the backend complexity of the system [5]. The size of software increases as the complexity increases. This is particularly useful for back-end part of any software. So, if more data appears in the database, then more relationship and more constraints may appears in the database also, and that not only increases the size of database, but increases the complexity of database in terms of relationships and various constraints. This will require the creation of more lines of code. With increased size and complexity, it is difficult to predict the cost of effort of development of the database part. The success of the software development depends on successful design of database

system. The size of database system primarily depends on number of relations in a relational database system as well as the number of tuples of each relation. In addition to this, the presence of the number of other database objects like views, triggers and data entry forms contributes substantially on size estimation as well as complexity estimation. This size helps to estimation the effort of development of database system.

Again, this effort of development as well as the effort of maintenance depends on the level of complexity of the database. The study of backend complexity not only helps to determine accurate estimation of effort, but helps to measure the quality of database system. So, it is important to study the back-end complexity of database part of any data centric software, which is the motivating factor of this paper. This paper has proposed a model for estimation of database development effort and the metric for estimating the back-end complexity.

## 3. REVIEW OF RELATED WORKS

Many estimation models have been proposed by many researchers for estimation of effort of the software and some of them are popular by the practitioners. The COCOMO and COCOCMO-II are very popular for estimation of effort and time of development of software. These primarily use Function Points or Lines of Code for estimating the size of a software system which has given priority on software rather than on database part [3]. The COCOMO II model recognizes different approaches to software development such as prototyping, development by component composition and use of database programming. COCOMO II supports a spiral model of development and embeds several sub-models that produce increasingly detailed estimates. Some works [3] considers the number of entities, relationships, number of attributes of entities and relationships and the path complexity for estimating cost at ER model. Some work [4] was based on object relational system and proposed metrics on Table Size (TS), Complexity of Weighted Methods (CWM), Cohesion between methods (COM), Coupling between Objects (CBO), Referential Degree, Depth in the Relational Tree, Number of Inherited Properties which are not applicable to relational model.

## 4. OUR PROPOSED DCS (DATABASE COMPLEXITY AND SIZE) METRIC

The relational database system is a collection of tables (two dimensional tables) having some properties. The complexity of relational database system depends on the following factors:

[I] Number of tables.

[II] Number of attributes of tables.

[III] Number of different types of constraints on given attributes.

[IV] Number of triggers to be used in the database.

[V] Number of data entry forms required.

[VI] Number of views to be created.

Item number [III] advocates the presence of entity integrity constraints (i.e. primary keys), the referential integrity constraints (i.e. foreign keys), the domain constraints (i.e. check constraints, not null, unique constraints etc.). Item number [IV] is derived from the number of functional dependencies. A trigger is a named PL/SQL block stored in a database and executed implicitly when a triggering event occurs. A triggering event is a DML (INSERT, UPDATE or DELETE) statement executed against a database table, view, schema or a database. It can be fired before, or after a triggering event.

The more is the number of triggers, the more the size of database as the more LOC (Lines of Codes) will be required for their implementation. Again, the triggers we discussed here include both application triggers and database triggers. Item number [V] is primarily depends on the number of relations of the database. Each forms to be generated for data entry purpose requires the creation of LOC. Item number [VI] advocates the use of virtual tables and implementation of each virtual table requires some LOC for their implementation. The more is the number of view to be created means the more the number of LOC to be generated. The presence of the above factors not only speaks about the size of lines of codes to be generated but it contributes a lot on overall complexity of database system. These factors one can identify during requirements gathering and analysis phase of database development. Here we have assigned weight measures to different constraints based on their level of complexity.

**Table 1**
**Weight Measure Assigned to Attributes**

| *Attributes* | *Weight Measure* |
|---|---|
| Primary Key | 0.75 |
| Foreign Key | 1.00 |
| Attributes having Constraint | 0.50 |
| Attributes having no constraint | 0.25 |

The reasons behind assigning the weight measures is that a weight measure of 1.00 out of 1 is assigned to foreign key as it establishes the referential integrity constraint among two (or same) table(s), so having the highest weight measure. The primary key is assigned with 0.75 weight measure as a primary key ensures not null as well as unique key constraints. Attributes having

other domain constraints requires practitioner to add statements in table definition. So a weight measure of 0.50 is assigned to it. Attributes having no constraints are assigned with a weight measure of 0.25 out of 1.

Based on the above assumption of weight measures on different attributes, we proposed the following complexity cost metric for each relation:

$$RC = Size * (\#PK*0.75 + \#FK*1.00 + \#AHC*0.50 + \#AHNC*0.25) \quad (1)$$

Where *RC* stands for relation complexity and *Size* indicates the number of tuples of the relation, #*PK* indicates number of attributes forming the primary key, #*FK* indicates the number of attributes forming foreign keys, #*AHC* indicates the number of attributes having other domain constraints and #*AHNC* indicates the number of attributes having no constraints.

Again, estimate the total number of triggers, number of views to be created for each relation, and the number of data entry forms to be generated and the estimated lines of codes (LOC) for each of them with the following metric:

$$TVFC = \sum_{t=1}^{t=n} LOC + \sum_{v=1}^{v=m} LOC + \sum_{f=1}^{f=k} LOC \quad (2)$$

Where *TVFC* stands for *Trigger_View_Form_Complexity* of a relation and it is estimated by summing up the estimated total lines of codes of all triggers, views and data entry forms of relations. The subscript *t* stands for trigger and it takes values from 1 to n. Similarly, the subscript *v* stands for views and it takes values from 1 to m and the subscript *f* stands for data entry forms and it takes values from 1 to k.

Then the total cost of complexity of database can be estimated with the following metric:

$$\text{TRC} = \sum_{i=1}^{i=k} (RC) \quad (3)$$

Where *TRC* stands for total relation complexity and the subscript *i* indicates the number of relations and varies from 1 to k.

Similarly, the total size of database in terms of lines of codes with respect to the number of triggers and views can be estimated with the following metric

$$TTVFC = \sum_{i=1}^{i=j} TVC \quad (4)$$

Where *TTVFC* stands for total *Trigger_View_Form_Complexity* and the subscript *i* indicates the number of relations and varies from 1 to *j*.

Here each and every relation may or may not have any trigger or view. So, we have taken the subscript *j* instead of *k*.

## 5. THE PROPOSED EFFORT ESTIMATION MODEL BASED ON DCS METRIC

Here we propose a model for estimating effort of database design and development based on our proposed DCS metric. This is depicted in Figure-1.

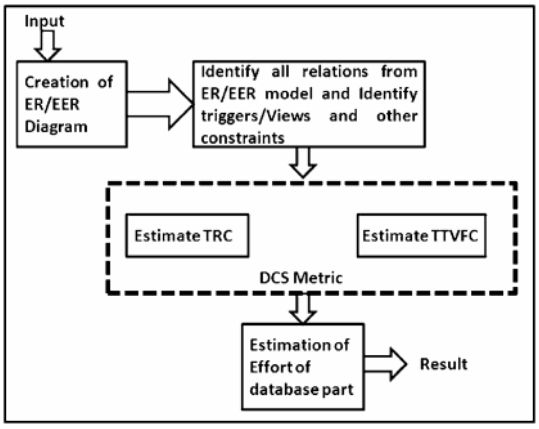


**Figure 1: Effort Estimation Model Based on DCS Metric**

In this model, first the requirements on database part of data centric software must be gathered. Then the ER/EER model is created after the proper analysis of gathered requirements. Next, the relations, and the different constraints on those relations along with the required triggers, views, data entry forms for each relation must be identified. Based on the proposed DCS metric, the complexity of database part as well as the size in terms of lines of codes based on the number of triggers, views, data entry forms are to be estimated. Based on DCS metric, the effort on database development can be estimated.

## 6. EXPERIMENTAL STUDY

For validation purpose of our proposed model for effort estimation based on our DCS metric, we took five students database projects. Our main aim was to study the database part without emphasizing on application part of the software. The database of all five projects were developed using Oracle 10G platform. The estimation result we found was very near to actual effort with an average difference of 5%. In this process, the estimation was done before the complete design of database, i.e. before the actual normalization process. The process was based on the initial requirements gathered and the information that we derived after ER/EER model. Table-2 reflects the experimental data that we found in our experiment.

**Table 2**
**Effort Estimation Table**

| | *Proj-1* | *Proj-2* | *Proj-3* | *Proj-4* | *Proj-5* |
|---|---|---|---|---|---|
| TRC | 1142 | 1844 | 1822 | 1882 | 1944 |
| TTVFC | 1500 | 1643 | 1682 | 1766 | 1902 |
| Estimated Effort | 8.9 | 12.2 | 12.2 | 12.8 | 13.6 |
| Actual Effort | 9.4 | 13 | 11.5 | 13.5 | 14 |
| Difference | 5% | 6% | 6% | 5% | 3% |

The result has been depicted in Fihgure-2 in a graphical form. The result has been found very satisfactory with an average 3% to 5% varying with actual effort and this is limited to small projects. A better result can be expected with larger project.

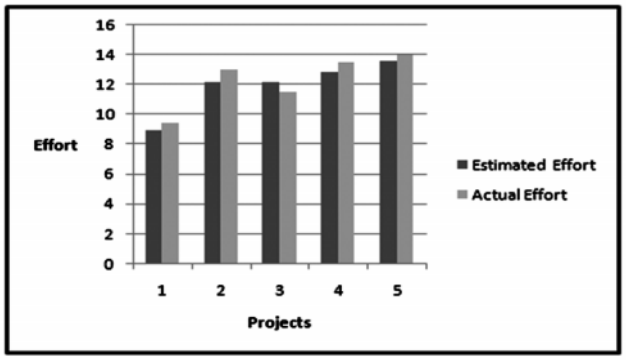


**Fig. 2: Graphical Representation of Estimated Effort with Respect to Actual Effort**

## 7. CONCLUSION

In this paper, we proposed a model for estimating effort using DCS metric of the database part of data centric software after a thorough study on the data requirements of the software. The experimental study was very encouraging. The future work can be done on validation of this work by taking industry related projects.

## REFERENCES

[1] Rajib Mall, "Fundamentals of Software Engineering", PHI, Second Edition, 2007.

[2] Ramez Elmasri, S. B. Navathe. D VLN Somayajulu, S.K Gupta , "Fundamentals of Database Systems" Pearson Education 2006.

[3] Yuan Zhao, and Hee Beng Kuan Tan, Wei Zhang, "Software Cost Estimation through Conceptual Requirement", *Proceedings of the Third International Conference on Quality Software (QSIC'03) 2003 IEEE.*

[4] Justus S, and K Iyakutti, "A Formal Suite of Object Relational Database Metrics", *International Journal of Information Technology,* 4:4 2008.

[5] Vandana Bhattacherjee Prabhat Kumar Mahanti Sanjay Kumar, "complexity estimation" *Journal of Theoretical and Applied Metric for Analogy Based Effort Information Technology*, 2009, **6**, No1. (pp 001 – 008).

[6] M Esperanza Manso, Marcela Genero , Mario Piattini. "No-Redundant Metrics for UML Class Diagram Structural Complexity" *CAiSE 2003, LNCS 2681*, pp. 127–142, 2003.© Springer-Verlag Berlin Heidelberg 2003.

[7] S. Mishra, P.K. Pattnaik, Rajib Mall, "A Novel Effort Estimation Model for Data Centric Software", *National Conference on Embedded System, Current Issues and Applications*, NCESA-2009, Feb14-15, 2009.