Early Estimation of Back-End Software Development Effort

Samaresh Mishra
School of Computer Engineering
KIIT University,
Bhubaneswar, India

Prasant Ku. Pattnaik
School of Computer Engineering,
KIIT University,
Bhubaneswar, India

Rajib Mall
Dept. of Computer Science & Engineering,
Indian Institute of Technology,
Kharagpur, India

ABSTRACT
An appropriate cost metrics for estimating development effort of the database part of an application at conceptual design stage using ER model is an important consideration. We propose four cost metrics based on the level of complexity of an ER diagram. Our effort estimation technique is based on these metrics using an empirical mathematical expression which is capable to predict total database complexity efficiently. Again, its outcome shows that the developmental effort is directly proportional to the total entity complexity and its total database complexity.

General Terms
Software Cost Estimation, ER Model

Keywords
Entity Complexity, Relationship complexity, Constraints, Depth of Inheritance, Effort estimation

1. INTRODUCTION
Software cost estimation is one of the key activities based on which all other project management activities are planned and carried out. Starting from the initial feasibility analysis stage, as more information becomes available, it should be possible to estimate the remaining development effort more accurately. The development process of the database part of a software application is usually carried out in the following phases: (1) conceptual design phase, (2) logical design phase, and (3) implementation phase. The success of the database design of data centric software depends on the understanding of the exact data requirements, their relationships as well as the constraints on the database part of the software. The estimation of effort would also depend on factors such as complexity of the software, the size of the software, the level of expertise of the development team and the tools used for development. Several published works [1][4][5] considered the number of entities, their attributes and the number of relationships among entities for determining the size and complexity, but the identification of different category of constraints at entity level, at attribute level, at relationship level and the business constraints may be considered for complete estimation. The prediction of these factors not only reflects the level of complexity but also predict the size in terms of number of tables required at the end as well as the number of lines of codes required for implementation of the business constraints [15].

This paper has been organized as follows: In section-2, we review related work. We discuss our proposed cost metric, in section-3. We present our model for estimation of effort, in section-4. In section-5, we present the experimental results. In section-6, we compare our work with related works. Finally, in section-7, we conclude this paper.

2. REVIEW OF RELATED WORK
Many estimation techniques [1][2][4][5][10][11][12][20] have been proposed and are being used extensively by industry on projects based on the function-oriented software development framework. Relatively, few works has been reported for object-oriented software development. However, research results on estimating efforts based on complexity of the development of database part of the software has scarcely been reported in the literature.

Constructive Cost Model (COCOMO-II) is at present being widely used [6][20]. It estimates the software effort in terms of size, which is based on lines of codes [8][17]. Later Mk-II Function Point Analysis was used for effort estimation [10][16]. The Use case Points estimation methods introduced in 1993 by Karner estimates effort in person hours based on use cases. This use case point method classified actors and use cases into three categories as simple, average and complex and assigned different weight factors to actors based on their interaction with the system, like using defined application user interface or through protocols like TCP/IP or through GUI or a WebPages. Similarly the weight factors assigned to use cases based on the number of transactions. Then they calculate the unadjusted use case point (UUCP) by adding unadjusted actor weights (UAW) and unadjusted use case weights (UUW). After this they assign values to the technical and environmental factors [0..5], multiply by their weights [-1..2], and calculated the weighted sums (TFactor and EFactor) for generating technical complexity factor (TCF) and environmental factors (EF) for the final estimation of adjusted use case points (UCP) by using formula $UCP = UUCP \times TCF \times EF$. Finally the effort was estimated by multiplying UCP with person-hours needed to implement each use case point (PHperUCP) [13]. These approaches may not specifically intend to estimate the database development cost. The DC (Database Complexity) method [10] measures database complexity based on the logical structure of physical database used in information systems. The DC method is silent about measuring database complexity at conceptual design phase. While determining the complexity of the database part of data centric software, it has been observed that the number of entities, number of relationships, number of attributes [1][4][9] identified at ER modeling stage are sufficient for estimating database size and complexity. The path complexity [1] was also been used for effort estimation. In addition to the entity sets,

relationship sets, and attribute sets, the business constraints were included for database cost estimation, without considering the features of object orientations of extended ER diagram [9]. The worked on estimating development effort of database by considering relational model artifacts also been undertaken and experimented with specifically on logical design phase [15] rather in conceptual design phase.

3. THE BEC (BACK-END COMPLEXITY) METRIC

The ER model includes an ER Diagram (ERD) and the semantic integrity constraints reflecting the business rules about data [2][3][7]. The ER Diagram was initially used in top-down approach for database domain modeling, but now it is also being used in Object-Oriented Analysis [1] approach. Although UML (Unified Modeling Language) has gained its popularity as standard software modeling methodology, but ER model is still widely accepted and used to model the data conceptually in the requirement capture and analysis phase by the practitioners [1]. The information about the number of subsystems, use cases and classes are required for estimating size of an object oriented development project. Similarly for component-based projects, the number of components, interfaces and classes are required for estimating software size. In the same way for the web-based projects, the number of web pages uses cases and scripts are considered for computing software size. Knowledge of these elements is vital for estimation of cost. In this manner, the artifacts of ER (EER) model may be applicable for measuring database cost based on its complexity and size.

3.1 Cost Metrics

The effort of development of the database part of data centric software primarily depends on the complexity of data modeling. The ER (and its extension, EER) diagrams are used for modeling relational as well as object relational database systems. Therefore, the estimation of effort of development of database on account of its complexity at early phase of development may correspond to estimating effort of database based on the complexity of ER model (and its extension). Here we have considered the Peter Chen's notation on ER modeling [12]. In ER (and its extension EER) modeling, not only one can identify the number of entity sets, relationship sets, attributes of entity sets and relationship sets, but also can identify the mapping cardinality as well as the participation constraints exists in the model. In addition to these, the depth of inheritance from generalization and specialization concepts as well as the aggregation can be considered in the process of estimating complexity. The higher is the number of the above specified factors may result with a higher in overall complexity of modeling. Keeping this in view, the following factors may be considered for computing the complexity of an ER model:

- Number of entities in an ER Diagram. [1]
- Number of relationships in an ER Diagram.[1]
- Number of attributes in an entity set in an ER Diagram.
- Number of descriptive attributes in a relationship set.
- Number of multivalued attributes in an entity or relationship set.
- Number of derived attributes in an entity or relationship set.
- EC: Entity Complexity.
- AC: Attribute complexity.
- RC: Relationship Complexity.
- DIT: Depth of Inheritance Tree.
- The mapping cardinality of relationships.

3.2 Entity Complexity

We classify the complexity of entities into simple and complex based on their associations with other entities of the same or other entity sets as well as their degree of dependency with other entities. The simple type is assigned to entity sets which add a foreign key in their own state after converting to relation in order to reflect their association with other entities. This increases the attribute size of entity sets. So, a weight measure of 1is assigned to this category of entity sets. The complex type is assigned to weak entity sets as they depend on strong entity set for their own existence. In this process, the prime attribute and the foreign key of the relation of weak entity set are derived from strong entity set. So a higher weight measure of 2 is assigned to this category of entity sets as they not only reflect their associations through foreign key definition but also through their dependency on strong entity set.

Table 1. The Weight Measures of Entity Sets

Entity Sets	Entity Type	Weight Measure of Entities (WE)
Entity set participated with 1:1 relationship OR participated with M:1 relationship OR participated with unary 1:M relationship	Simple	1
Weak entity set	Complex	2

The different type of attributes of entity sets also contributes to their overall complexity. Therefore, we classify the complexity of attributes as simple, average and complex based on their contribution to the state of the relation to which they belongs to. The multivalued attributes are categorized as complex type as their presence in the base relation resulted creation of smaller relations through decompose. This is done in order to maintain the relation state in 1NF. So, a higher weight measure of 2 is assigned to this category of attributes. The average type is assigned to all the derived attributes of an entity set as they derive data value from some calculation. This requires procedural implementation through lines of codes. Hence, a lower weight measure of 1.5 is assigned to this category of attributes. The simple type is assigned to all other category of attributes as they neither require procedural extension nor do they create separate relations. Therefore, the lowest weight measure of 1 is assigned to this category of attributes.

Table 2. The Weight Measure of Attributes

Attribute Category	Attribute Type	Weight Measure of Attributes (WA)
Multivalued attribute	Complex	2
Derived attributes	Average	1.5
Other Attributes (including descriptive attributes of relationship sets)	Simple	1

Another aspect of complexity of individual entity set is based on its depth of inheritance (DIT) in a generalization relationship. It is the maximum of the DIT (Depth of Inheritance Tree) values obtained for each entity set in the ER model. The DIT value for an entity set within a generalization hierarchy is the longest path from the entity set to the root of the hierarchy. The more in the DIT value may results in inheriting attributes from higher level entity set(s) to lower level of entity set.

The structural complexity of entity sets can be calculated by counting the complexity of individual entity sets based on the different category of attributes they have and also counting the DIT in a generalization. The total entity complexity is calculated by counting the number of entity sets of each category (based on their association with other entity sets), multiplying each by its weighting factor, and adding up the products.

An entity can have more than one association with entities of other entity sets. In some association, it may fall in simple category and in some it may be in complex category. Therefore the entity complexity (EC) may be computed as follow:

$$EC = \sum_{i=1}^{i=NOA}(WA)_i + DIT + \sum_{j=1}^{j=NoAss}(WE)_j \qquad (1)$$

Where, NoAss represents the number of associations the entity set has with other entity sets. NOA represents the number of attributes the entity set.

Once the EC is calculated, then the total entity complexity (TEC) of ER diagram can be calculated as follow:

$$TEC = \sum_{j=1}^{j=NOE}(EC)_j \ . \qquad (2)$$

Here, NOE represents number of entity sets.

3.3 Relationship Complexity

Similarly to entity complexity, we categorized the relationship complexity as simple, average and complex type based on the number of referential integrity constraints established by the relationship and based on the association of relationship with aggregation. The simple type is assigned to relationships with mapping cardinality 1:1 or 1:M or M:1 as they do not results any new relations. This category results only one foreign key (or a composite foreign key) in the relation of many side entity set. As this category do not generate any new relation, so no weight measure is assigned to this category of relationship sets. The average type is assigned relationships having many to many (M:N) mapping cardinality constraints and associative entity sets as they results a new relation for the relationship set and generates at least two foreign keys in the resultant relation and the foreign keys contribute to formation of a composite primary key. The associative entity sets are similar to binary M:N relationship sets and having a peculiar extra attribute which can act as a primary key. So, a weight measure of 2 (or more than 2) is assigned to this category of relationship sets as they add two sets of foreign keys (or more than two sets of foreign keys) in the resultant relation. This weight measure depends on the degree of relationship, that is, 2 for binary, 3 for ternary and n for n-ary relationship. The complex type is assigned to relationships that exist among aggregation and another entity set. As aggregation relationships are treated as higher level entity sets [14], the relationship created among aggregation and other entity sets results substantially more foreign keys in the relation which reflects this relationship. So, a higher weight measure of 3 is assigned to this category.

Table 3. The Weight Measure of Relationship Sets

Relationships Set	Relation Type	Weight Measure of relationship (WR)
M:N Relationship or Associative Entity Sets	Average	≥2
Relationship with Aggregation	Complex	3

The individual relationship complexity (RC) can be calculated by counting the number of attributes the relationship has in terms of its descriptive attributes and the primary key attributes inherited from participated entity sets for forming the primary key for the relation of this relationship set and then adding the weight measure. The relationship complexity (RC) may be expressed as follow:

$$RC = \sum_{l=1}^{l=NOA}(WA)_i + (WR) \ . \qquad (3)$$

Here NOA represents number of attributes the relationship has (refer Table 2) which include the descriptive attributes and the attributes used for primary key.

The total relationship complexity (TRC) of ER diagram can be calculated as follows:

$$TRC = \sum_{j=1}^{j=NOR}(RC)_j \ . \qquad (4)$$

3.4 The Business Constraints

The types of constraints that can be specified in ER diagram are mapping cardinality constraints and the participation constraints, which are taken care by our proposed cost metrics. But there may exist many other business constraints which can be categorized as schema-based integrity constraints and semantic-integrity constraints. These constraints are needed to be represented during logical design. But if some (or all) of these constraints are known during requirements gathering and analysis phase of software development as well as during the conceptual design phase of database development of the software, then this may contribute substantially to the overall complexity estimation. Here we have taken only semantics integrity constraints which are identified only during the early phase of database design. Our complexity metric named as Total Semantic Constraint Complexity (TSCC) can be expressed as:

$$\text{TSCC} = \sum_{k=1}^{NOC} C_k \quad . \qquad (5)$$

Here NOC represents the number of constraints and C_k represents the semantic integrity constraints captured during requirements gathering and it has been assigned a weight measure of 1.5.

3.5 Total Complexity of ER Model

The total complexity (TC) of ER model can be calculated as the sum of all above estimated complexities with the following expression:

$$TC = TEC + TRC + TSCC \ . \qquad (6)$$

4. Effort Estimation Based on BEC

After finding the Total Complexity of ER model (TC), it is required to find the Technical Complexity Factor (TCF) and Environment Factor (EF) using the formula: $\text{TCF} = 0.6 + (0.01 \times \text{TFactor})$ and $\text{EF} = 1.4 + (-0.03 \times \text{EFactor})$ [13]. We calculated the adjusted ER Point (ERP) using the widely used formula $\text{ERP} = \text{TC} \times \text{TCF} \times \text{EF}$. The Estimated effort in person-hours may be calculated as $\text{Effort} = \text{ERP} \times \text{PHperERP}$. The PHperERP is stands for person-hour per ERP. Here we have not considered the environmental factor rather consider the technical complexity factors for effort estimation. The PHperERP considered here is 1.00 and it can be increased to a higher value if the effort of modeling, analysis, design, coding and testing of ER model is considered.

As shown in Figure 1, data related to data modeling needs to be gathered under software requirements gathering phase. Then the ER (and its extension EER) diagram is undertaken as part of data analysis followed by estimating the complexity using BEC cost metric and then effort calculation is based on the back-end complexity using ERP.

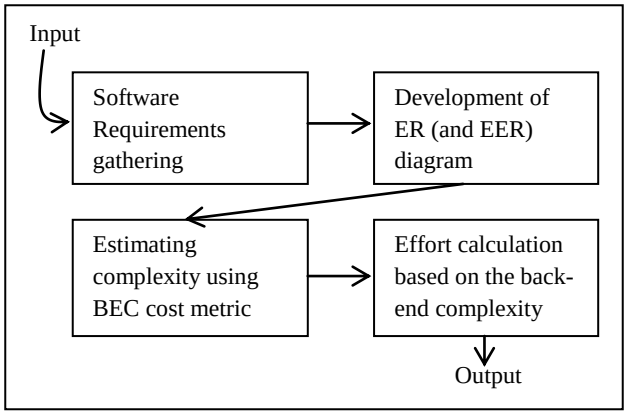

Fig. 1. Effort Estimation Using BEC Metric

5. EXPERIMENTAL STUDY

For empirical evaluation of proposed approach of effort estimation, we considered ER diagrams (and its extensions, EER) of ten different student projects developed as part of course assignments of KIIT University, having varying with different number of business constraints. The details of all ten projects and their data have been depicted in Table 4. It has been observed from Table 4 that ER diagrams having more entities, relationships and constraints (both ER constraints and business constraints) are resulted with higher total complexity as well as effort of development. The higher in the overall complexity results with increasing in overall size of the system. Also, it has been observed from the Table 4 and Figure 2 that the mean MRE is with value 0.26394 of the estimated effort with actual effort

Our proposed model has been validated by analyzing their accuracy in terms of error range and PRED [18]. The MRE (Magnitude of Relative Error) of all projects and the MMRE (Mean MRE) of all eight projects are presented in Table 4. The MRE and MMRE are computed using the following formula:

$MRE = \left|\frac{Effort_{Estimated} - Effort_{Actual}}{Effort_{Actual}}\right|$ and $MMRE = \frac{1}{n}\sum_{i=1}^{i=n}(MRE)_i$

The PRED(25) can be defined as the proportion of frequency that predicted effort fall within 25% of actual effort and this can be achieved by the equation $\text{PRED}(25) = \frac{k}{n}$, here k denotes the number of projects with MRE less than equal to 25%. It has also been observed from the data set that 62% of projects estimated effort with PRED(25) = 0.625, which is encouraging.

The BEC metric approach for estimating effort of development of database part of software is primarily based on the level of complexity rather than only on the size of the database. Our result based on the dataset from Table 4 shows that the estimated effort is very close to the actual effort of development. This is presented graphically in Figure 2.

Table 4. Cost Versus ER Diagrams Complexity

Project No.	TEC	TRC	TSCC	TC	TCF	ERP	Estimated Effort in PH	Actual Effort in PH	MRE	Mean MRE
1	9	7	6	22	0.64	14.08	14.08	18	0.218	0.26
2	31.5	5	10.5	47	0.64	30.08	30.08	42	0.284	
3	19	12	12	43	0.64	27.52	27.52	36	0.236	
4	15.5	4	7.5	27	0.64	17.28	17.28	24	0.28	
5	16.5	3	7.5	27	0.64	17.28	17.28	24	0.28	
6	23	2	9	34	0.64	21.76	21.76	32	0.32	
7	26	5	7.5	38.5	0.64	24.64	24.64	33	0.253	
8	38	0	6	44	0.64	28.16	28.16	37	0.239	

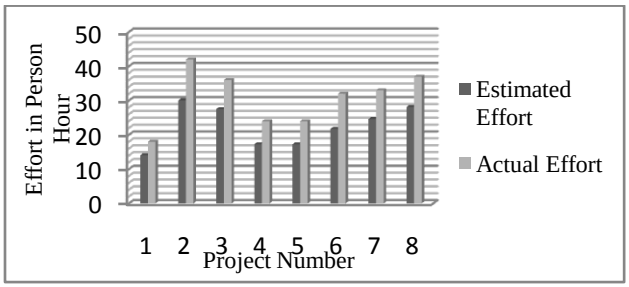

Fig. 2. Estimated Versus Actual Effort

It has been observed from Figure 3, that the overall effort of database development primarily depends on total entity complexity taken from ER diagram and moderately on total semantic complexity. The effort takes less account on the presence of total relationship complexity which is based on M:N relationship and aggregation.

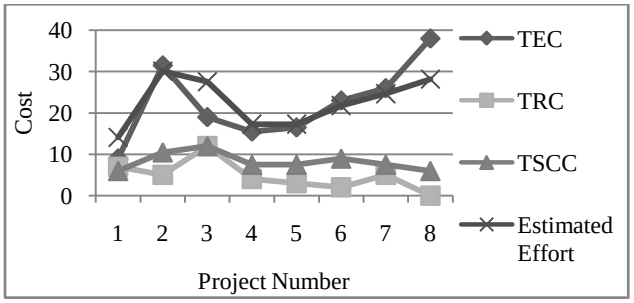

Fig. 3. Effort versus TEC, TRC, TSCC

Again, we considered the total complexity (i.e. the sum of entity complexity, relationship complexity and semantic constraint complexity) with the estimated effort. It has been observed from the Figure 4, that database development effort proportionately increases with the total complexity.

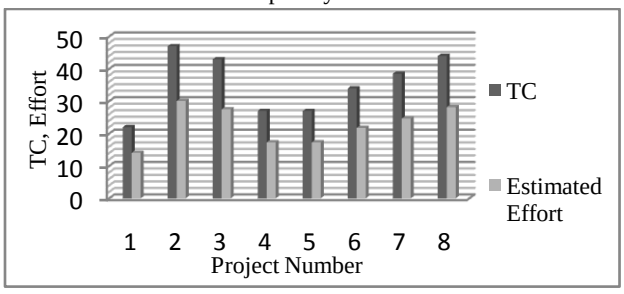

Fig. 4. Total Complexity (TC) versus Estimated Effort

The multiple regression analysis approach has been adopted here in order to study the effectiveness of our model. The resultant equation of effort estimation based on multiple regressions with respect to actual effort is given below:

$$effort = -1.074 + 0.759 * TEC + 0.303 * TRC + 1.629 * TSCC$$

The multiple coefficients (R^2) is 0.9959 which is close to 1 and the adjacent R^2 is 0.9929, which is close to R^2 indicating better strength of multiple regression relationship.

6. COMPARISON WITH RELATED WORK

Many techniques for estimating the cost of the relational database development based on ER model have been reported in the literature. ER model is known for its rich capability of specifying various business constraints. Several related works have used the number of entities, relationships and attributes and some have considered the path complexity [1]. The path complexity metric is a complexity metric and used for effort estimation and it is used for getting the information about number of paths one entity could influence other entities and the length of each path. This path complexity is computed from a graph derived from the ER diagram. So the process of creating a graph from an ER diagram and then calculating complexity from the graph is itself an additional effort in the process of estimation. Compared to the above, we used a catalog table (Table 4) containing the information about total entities, total relationships, total semantic constraint complexity and based on these information, the total complexity and then the effort can be estimated. The work [19] takes into account only the number of fields, primary keys, and foreign keys for effort estimation with the given formula:

$$Effort1 = 2.94 - 0.032 * Number\ of\ fields + 2.90 * Number\ of\ Primary\ Keys - 2.62 * Number\ of\ Foreign\ Keys$$

This paper also estimated effort by considering the number of 1:1 and 1:M relationships using the formula:

$$Effort2 = 4.24 + 3.23 * Number\ of\ 1{:}1\ relationships + 0.007 * \text{Number of 1: M relationships}$$

We compare our proposed effort estimation approach with the above work [19] and the result as shown in Figure 5.

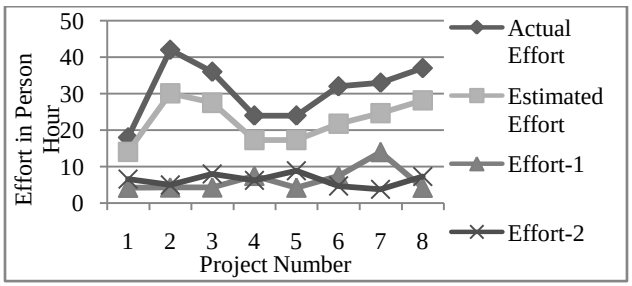

Fig. 5. Comparison of Actual Effort with Estimated and Existing Approach

However, the BEC metric model includes the features of ER and EER diagram like mapping cardinality, degree of relationships, different types of attributes, the concepts of aggregation and the depth of inheritance from generalization/specialization. So, for a better estimation of database development effort, it is desired to identify all factors contributing to total size and complexity of the database at the conceptual design stage.

7. CONCLUSION

We have proposed an effort estimation technique namely BEC metric model. The experimental outcomes conducted by us predict the development effort within an improved accuracy of 3 to 4%. The model may deploy to outsource the development of the database part of an application. As a future scope of work, this can be experimented with more industry standard projects.

8. REFERENCES

[1] Yuan Zhao, and Hee Beng Kuan Tan, Wei Zhang, Software Cost Estimation through Conceptual Requirement, Third International Conference On Quality Software (QSIC'03), 2003 IEEE, pp.141

[2] B. Londeix, Cost Estimation for Software Development, STC Telecommunications, UK, Addison-Wesley Publishing Company, 1987.

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

[4] Capers Jones, Estimating Software Costs, Bringing Realism to estimation, 2nd Edition, TMH 2007.

[5] Geoffrey J. Kennedy, Elementary structures in entity-relationship diagrams: a new metric for effort estimation, 1996 IEEE, pp. 86-92

[6] B. W. Boehm, C. Abts, A. W. Brown, S. Chulani, B. K. Clark, E. Horowitz, R. Madachy, D. J. Reifer, and B. Steece, Software Cost Estimation with COCOMO II, Prentice Hall PTR, 2000.

[7] Narayan S. Umanath, Richard W. Scamell, Data Modeling and Database Design, THOMSON INDIA Edition 2007.

[8] Emilia Mendes, Nile Mosley, Steve Counsell, Web Effort Estimation, Web Engineering, Springer International Edition 2006.

[9] Samaresh 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.

[10] Mile Pavlic, Marin Kaluza, Neven Vrcek: Database complexity measuring method, Proceedings of the ISRM 2002 Conference, Las. Vegas, NV, USA, 2002.

[11] Marcela Genero, Luis Jiménez, and Mario Piattini Measuring the Quality of Entity Relationship Diagrams, A.H.F. Laender, S.W. Liddle, V.C. Storey (Eds.): ER2000 Conference, LNCS 1920, pp. 513-526, 2000. Springer-Verlag Berlin Heidelberg 2000

[12] P. P. Chen, The Entity-Relationship model – towards a Unified View of Data, ACM Trans, Database Syst,1(1), Mar. 1976, pp. 9-36

[13] Parastoo Mohagheghi, Bente Anda, Reidar Conradi, Effort Estimation of Use Cases for Incremental Large-Scale Software Development, ICSE'05, May 15–21, 2005, ACM 1-58113-963-2/05/000.

[14] A. Silberschatz, H. F. Korth, S. Sudarshan, Database System Concepts, 5th Edition, McGRAW HILL Publication. pp-234.

[15] Samaresh Mishra, Krushna Chandra Tripathy, Manoj Kumar Mishra, Effort Estimation Based on Complexity and Size of Relational Database System, International Journal of Computer Science and Communication, Vol. 1, No. 2, July-December 2010, pp. 419-422.

[16] Koh, T.W., M.H. Selamat and A.A.A. Ghani, Exponential effort estimation model using unadjusted function points, Information Technology Journal, 2008, Volume: 7, Issue: 6, pp 830-839.

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

[18] Jianfeng Wen, Shixian Li, Linyan Tang, Improve Analogy-Based Software Effort Estimation using Principal Components Analysis and Correlation Weighting, 16th Asia-Pacific Software Engineering Conference,2009, pp 179-186

[19] Bushra Jamil, Javed Ferzund, Asma Batool, Shaista Ghafoor, Empirical Validation of Relational Database Metrics for Effort Estimation, 6th International Conference on Network Computing, IEEE, 11-13 May 2010, pp-1-5

[20] Samaresh Mishra, Kabita Hazra, Rajib Mall, A Survey of Metrics for Software Development Effort Estimation, International Journal of Research and Reviews in Computer Science, Vol. 2, No. 5, October 2011, pp. 1199-1204.