Comparative Study of Back-End Vs Front-End System by Performance Analysis during Preliminary Design stages

S. Jagannatha, T.V Suresh Kumar, R. RajaniKanth

Department of Computer Applications,
M.S. Ramaiah Institute of Technology, Bangalore (INDIA)
jagannatha@msrit.edu, sureshkumar@msrit.edu, rajankanth@msrit.edu

Abstract

Performance is an important non-functional attribute of the software system to be considered for producing quality software. Effort estimation helps in the calculation of size of the software and database separately, which is one of the required data in assessing the performance. We use Entity Relationship(ER) and Use case mode for compute the data base effort and software effort respectively. In this chapter we proposed a methodology for performance analysis of back-end and front-end applications by estimation of effort and compute response time. We propose the comparative study of response time and validate the results.

1. Introduction

The effort estimation is the process of estimating the total effort required to complete a software project.Cost estimation is one of the keyaspects of project management based on which performance goals can be achieved. The effort estimation depends on the complexity of software, ER diagram, data requirements Performance problems may be so strict that they require extensive changes to the system architecture in Preliminary design stages. If these changes are made late in the development process, they can increase development costs, delay deployment, or adversely affect other desirable qualities of a design. It is vital to maintain for early assessment of the performance characteristics of distributed database system since its functionality is decentralized.

The data are required for the database effort also depends on the complexity of entities, attributes, relationship type and structural constraints. Predicting these factors reflect the level of complexity and predict in terms of number of lines of codes required for implement the business constraints. The data required for estimation for front end application based on functionality which represented in usecase diagram. The functionality 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. 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 multiply by their weights and calculated the weighted sums 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 * TCF * EF . Finally the effort was estimated by multiplying UCP with person-hours needed to implement each use case point [1].

Performance is an important issue for a software system. Even several development methodologies such as performance considerations in hardware industries, system software and networking environment and application programming. Performance evaluation system must address performance user point i.e. response time of the software solutions.

Software Performance Engineering (SPE) is a method to predict the performance of software systems early (analysis phase) in the life cycle. SPE continues through the detailed design, coding and testing stages to predict and manage the performance of the evolving software and to monitor, report actual performance against specifications and predictions. Modelling software systems to predict the performance using Unified Modelling Language (UML)

The rest of the paper is organized as follows: Review of the related literature is given in Section 2. Section 3 discusses the basic concepts involved in this paper. The proposed methodology is described with an algorithm in Section 4. The illustration of proposed technique with a case study is presented and the results are validated in section 5. The paper concludes with observations and the future work in section 6

2. Literature Survey.

Many researchers are made significant contribution in the field of software effort estimation using in the field of software development.In [1][4][5] estimation technique based on the function-oriented software development frame work. ER model for the estimation of backend effort cost has been proposed [9][11][20][38] Research results on estimating efforts based on complexity of the development of database part of the software has narrowly been reported in the literature. In [39][40] use case point approach is widely used industry for estimation of software cost. Use case approach mainly based on an actor, compute unadjusted actor weights Calculate unadjusted use case weights, based on use case, Calculate unadjusted use case points consider the assumed rates of technical and environmental factors. Calculate adjusted use case point, compute line of code using gearing factor LOC ← UCP *gearing factor, Convert LOC into number of executable statements, Calculate equivalent size of software in KB, Adjust the size for software resource requests, Solve the use case performance model to obtain response time. The effort estimation using this approach is illustrated and its effectiveness is proved with case studies in [43], [44], [45], [46], [47].

Some authors in [41] used number of primary key and foreign key for effort estimation of ER model using the formula. Many techniques for estimating the cost of the relational database development based on ER model have been reported in the literature. However, these are actually execution details and difficult to estimate during the early stage of software development. The entity relationship (ER) model is well used in conceptual modeling for data base part of software systems. The path complexity metric is a complexity metric and used for effort estimation. This path complexity is computed from a graph derived from the ER diagram. Compute the information about total entities, total relationships, total constraint complexity and based on these information, the total complexity and then the effort can be estimated [1].

Software Performance Engineering has evolved over the years and has been demonstrated to be effective during the development of many large systems [3], [29][30].

3. Propose Methodology

Effort estimation helps in the calculation of size of the software which is one of the required data in assessing the performance of software systems using Entity Relationship(ER) model. [1].

In this chapter we proposed a methodology for performance analysis of DDS is as follows

i. Procedure to estimation of effort is based on the complexity of ER model and compute response time. We consider the Technical and Environment factor for computing adjusted ER Point (ERP) for effort estimation. Computing Lines of Code (LOC) from the estimated effort and compute response time and the resource requirement of each of the hardware device for the scenario by considering the size is adjusted as a typical workload for the software resource.

ii. Estimate effort based on the use case diagram by computing the UCP. Computing Lines of Code (LOC) from the estimated effort in step i, and compute response time and the resource requirement of each of the hardware device for the scenario by considering the size is adjusted as a typical workload for the software resource using the smith approach.

4. Algorithm

The algorithm to calculate the response time is as follows.

1. Consider the ER model for the DDs application under assessment.Calculate Compute adjusted ER Point (ERP) using the widely used formula ERP=TC×TCF×EF using ER model approach [1].
2. Estimate The size by using equation Size in KB = (ERP*NLOC*NW*Wlength)/(1024*28) ,Where ERP = is the adjusted ER point obtained in step 1, NLOC=Number of Lines of Code per ER point NW= Number of words required per instruction, Wlength=Word length in bits.
3. Calculate the response time using the resource overhead matrix given in table 1.
4. Consider the Use case model of DDA applications. compute UCP=UUCP * TCF * EF .Where UUCP is unadjusted use case point, TCF is the technical factor and EF is the environment factor.
5. Compute the KB = The size by using equation Size in KB = (UCP *NLOC*NW*Wlength)/(1024*28) ,Where ERP = is the adjusted ER point obtained in step 1, NLOC=Number of Lines of Code per ER point NW= Number of words required per instruction, Wlength=Word length in bits.

6. Calculate the response time using the resource overhead matrix given in table 1.
7. Compare the response time for back end and front applications.

5. Illustrate the Problem with the Case Study.

5.1 ER Model Approach

In this chapter we consider a case study of a Company Database System. The company organized into departments and have several locations and implements several projects. Employees in each department supervise the projects. Company keeps track of all dependents for insurance purpose. We use Entity Relationship(ER) diagram for effort estimation which is presented in the Figure1 the use case diagram represented in the figure 2. We consider eight different projects having different number of constraints. We estimate the performance analysis by computing the response time and the results are validated with the proven results of Connie U. Smith [3].

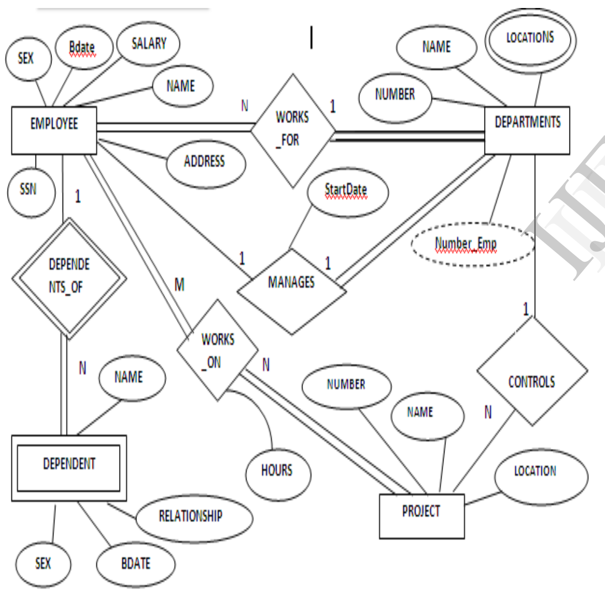

Figure 1. ER model for the case study.

	Devices				
	CPU	Disk	INet	Delay	LAN
Quantity	4	3	1	1	1
Service units	Sec.	I/Os	Kbtyes	Sec	Msgs

5. 2Over Head matrix

Processing Time of each parameter in each resource

Table 1 Processing Resource Overhead Matrix

Input(Per KB)	0.002		1		
DB Acces(Per Acess)	0.0005			0.25	1
LocalDB (Per KB)	0.01	2			
Page size(Per KB)	0.0005		1		
Data size(Per KB)	0.0005		1		1
Service time in Sec	1	0.003	0.14222	1	0.000164

Theoverhead matrix in Table 1 shows the resources that are considered for the processing with their processing time. Table 1 shows that are 4 CPUs, 3 disks, one network, one delay and there is one delay the service units of the devices are mentioned in the service units row. The input size is dividing into 5 parameters input data - the number of Kbytes in the input messages. DB Access – the number of access to the database, Local DB – the number of access to the local database, Page size – the number of Kbytes in the page and Data Size - the number of Kbytes in data retrieved.

5.3 Performance Analysis

The ER model of Company Database System is shown in Figure 1 The ER point of the Company Database System is shown in table 2 in the second row. We have considered eight different projects having different number of constraints and ER point (ERP) and estimated effort with actual effort is shown in table 2.

Table2 Computation of ERP versus Actual Effort

PR NO	TEC	TRC	TSCC	TC	TFACTOR	TCF	Efactor	EF	ERP	Estimated Effort	Actual effort
1	28	5	7.5	40.5	47	1.07	28	0.56	24.26	24.26	28
2	23.5	4	7.5	35	28	0.88	28	0.56	17.248	17.248	25
3	26	5	7.5	38.5	42	1.025	28	0.56	22.099	22.099	27
4	10	7	6	23	25	0.85	28	0.56	10.948	10.948	17
5	16.5	3	7.5	27	37.5	0.97	28	0.56	14.664	14.664	21
6	14	4	12	30	39	0.99	28	0.56	16.632	16.632	26
7	16.5	4	12	32.5	42	1.025	28	0.56	18.655	18.655	28
8	22	3	13.5	38.5	40	1	28	0.56	21.56	21.26	32

Table3 Computation of ERP versus Actual Effort

Project Number	Actual Effort/ERP	LOC	Size(KB)	Response Time
1	28	1484	71.232	8.0890
2	25	1325	63.6	7.7311
3	27	1431	68.688	9.2956
4	17	901	43.248	5.2011
5	21	1113	53.424	7.5773
6	26	1378	66.144	9.4794
7	28	1484	71.232	9.6142
8	32	1696	81.408	11.6631
Average Response Time:				8.5813

Table 4. Estimation of Response Time with the Actual Effort

Project Number	ERP	LOC	Size(KB)	Response Time
1	24.26	1285.78	61.717	6.7294
2	17.248	914.144	43.879	4.9132
3	22.099	1171.247	56.220	7.4883
4	10.948	580.244	27.852	3.0012
5	14.664	777.192	37.305	5.2742
6	16.632	881.496	42.312	6.0742
7	18.655	988.715	47.458	6.2173
8	21.56	1142.68	54.849	7.8681
Average Response Time :				5.9457

Where lines of code is estimated is LOC = ERP *53 and the size of code is computed using formula: Size = (ERP * NLOC * NW * Wlength) / (1024 * 28). Where NLOC: number of lines of code per ERP (53),NW: Number of words per instruction (6), Wlength: word length (64 bits). There for the size is computed as follows:Size = (ERP *53*6*64)/(1000*8).We estimated the software size using the ER point approach and found the response time using the algorithm shown above. We considered the response time as performance metric and calculated the response Time of company database system. The ER point, size in KB and response time of all eight projects of estimated and actual effort shown in table 3 and table 4 respectively.

5.3 Use Case Approach: Consider the company database system. The use case diagram shown in the figure 2

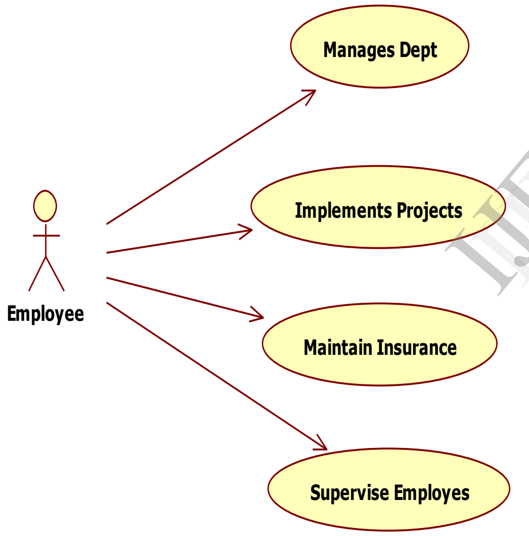

Figure 2 Use case model for the Company Data base System.

Estimation of the size of the use cases: we use the following algorithm for the estimation of the size of the use cases. The use case model given in the figure2 consists of manage department, project implementation, maintain insurance and supervise employees.The algorithm for the proposed methodology is given below. The algorithm uses the procedure for calculating UCP.

```
Identify key performance scenarios
For all scenarios S do
Identify use cases u and actors a
Develop use case model U
{Calculate unadjusted actor weights}
for all actor a ϵ U do
        if a is simple then
                asimple ← asimple + 1
        else if a is average then
                aaverage ← aaverage + 1
        else
                acomplex ← acomplex + 1
        end if
UAW ← asimple*1+aaverage*2+acomplex*3
end for
{Calculate unadjusted use case weights}
for all use case u ϵ U do
if u is simple then
usimple ← usimple + 1
else if u is average then
uaverage ← uaverage + 1
else
ncomplex ← ucomplex + 1
end if
UUCW ← usimple*5+uaverage*10+ucomplex*15
end for
{Calculate unadjusted use case points}
UUCP ← UAW + UUCW
for all technical factors tf do
assign rate rtf
 end for
        $TFactor \leftarrow \sum_{tf=1}^{13} r_{tf} * w_{tf}$
        TCF ← 0.6 + (0.01 * TFactor)
        {calculate environmental factor}
        for all environmental factor ef do
                assign rate ref
        end for
        $EFactor \leftarrow \sum_{ef=1}^{8} r_{ef} * w_{ef}$
        EF ← 1.4 + (-0.03 * EFactor)
                {Calculate adjusted use case point}
                UCP ← UUCP*TCF*EF
                {calculate LOC}
                if using gearing factor then
LOC ← UCP*gearing factor
Convert LOC into number of executable statements
Calculate equivalent size of software in KB using the formula
Size = (ERP * NLOC * NW * Wlength) / (1024 * 28).
Where NLOC: number of lines of code per ERP (53),
NW: Number of words per instruction (6),
Wlength: word length (64 bits). There for the size is computed as follows:
Size = (ERP *53*6*64)/(1000*8).
Adjust the size for software resource requests
Solve the use case performance model to obtain response time
```

Table 5. Response time for the Company Data base System. (Comparative Study)

Use case	Size in KB	ER Model Approach
Manages Dept	30.234	7.7311
Implements Projects	54.23	
Maintain Dependents for Insurance	36.871	
Supervise Employees	30.955	
Total Response time	152.29	

The results shown in the table 5 is the response time of the individual use case and total response time is 152.29 seconds of complete applications for company database system using use case approach. Whereas the response time of using ER model of the same applications is 7.7311 seconds. Hence the ER model gives only used for effort estimation can be considered

6. References

[1] Samaresh Mishra, Prasant Ku. Pattnaik, Rajib Mall Early Estimation of Back-End Software Development Effort International Journal of Computer Applications (0975 – 8887) Volume 33– No.2, November 2011

[2] ParastooMohagheghi, BenteAnda, ReidarConradi, Effort Estimation of Use Cases for Incremental Large-Scale Software Development, ICSE'05, May 15–21, 2005, ACM 1-58113-963-2/05/000.

[3] EvangelinGeetha, D., Suresh Kumar, T. V. RajaniKanth, K..: 'Framework for Integrated Performance Prediction Process (IP3) Model', Technical Report, Department of MCA, MSRIT, TRMCA 03, March 2010.Jhk;

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

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

[6] RamezElmasri, S. B. Navathe. D VLN Somayajulu, S.K Gupta, Fundamentals of Database Systems, Pearson Education 2006.

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

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

[9] 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.

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

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

[12] 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.

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

[14] 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

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

[16] 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.

[17] 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.

[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, JavedFerzund, AsmaBatool, ShaistaGhafoor, 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, KabitaHazra, 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.

[21] Connie, U. Smith: 'Performance Engineering of Software Systems' (Addison-Wesley, 1990).

[22] Connie, U. Smith, and Lloyd G. Williams: 'Performance Solutions' (Addison-Wesley, 2000).

[23] Connie, U. Smith: 'Software Performance Engineering: A Case Study including Performance Comparison with Design Alternatives', IEEE Transactions on Software Engineering, 19,(7), July 1993, pp. 720-740.

[24] Vittorio Cortellessa, RaffaelaMirandola, 'PRIMA-UML: A Performance Validation Incremental Methodology on Early UML Diagrams', Science of Computer Programming, Elsevier, 44, (1), 2002, pp. 101-129.

[25] Dorin, B. Petriu, Daniel Amyot, Murray Woodside, and Bo Jiang: 'Traceability and Evaluation in Scenario Analysis by Use Case Maps', LNCS,

3466, 2005, pp. 134-151.

[26] Andrea D'Ambrogio, Paolo Bocciarelli: 'A model-driven approach to describe and predict the performance of composite services', Proc. Workshop on Software and Performance 6th International Workshop on Software and Performance' , WOSP'07, ACM ,Buenos Aires, Argentina, February 5–8, 2007 , pp: 78 – 89.

[27] Simonetta Balsamo Moreno Marzolla: 'Performance Evaluation of UML Software Architectures with Multiclass Queueing Network Models', Proc. 5th International Workshop on Software and Performance, Spain, July, 2005, pp. 37 – 42.

[28] EvangelinGeetha, D., Suresh Kumar, T. V., and RajaniKanth, K.: 'Early Performance Modeling for Web Based Applications", Proc CIT '04, Hyderabad, LNCS, 3356/2005, Springer Verlag, December 2004, pp. 400-409.

[29] EvangelinGeetha, D., Suresh Kumar, T. V., and RajaniKanth, K.: 'Early Performance Modeling for Multi Agent Systems using UML 2.0', IJCSNS International Journal of Computer Science and Network Security, March 2006, 6, (3A), pp.247-254.

[30] EvangelinGeetha, D., Ram Mohan Reddy, Suresh BenteAnda, HegeDreiem, Dag, I. K., SjobergandMagne Jorgensen: 'Estimating Software development Effort based on Use Cases – Experiences from Industry', www.idi.ntnu.no/emner/tdt4290/docs/faglig/uml2001-anda.pdf. Accessed January 2007.

[31] Edward, R., Carroll: 'Estimating Software Based on Use Case Points', OOPSALA '05, San Diego, California, October 2005, pp.257-26.

[32] Kusumoto, S., Matukawa, F., Inoue, K.: 'Estimating Effort by Use Case Points: Method, Tool and Case Study', Proc.10th International Symposium on Software Metrics (METRICS'04), September 2004, pp. 292-299.

[33] Suresh Nageswaran: 'Test Effort Estimation using Use Case Points', Quality Week, San Francisco, June 2001, pp.1-6.

[34] Roy, K., Clemmons: 'Project Estimation with Use Case Points', Cross Talk-The Journal of Defence Software Engineering, February 2006, pp:18-22.

[35] James Rumbaing, Ivar Jacobson, Grady Booch: 'The Unified Modeling Language Reference Manual', (2nd Edition, Pearson Education, 2006).

[36] Robert, T., Futrell, Donald, F., Shafer, and Linda, I. Shafer: 'Quality Software Project Management', (Pearson Education, 2006).

[37] Linda, M., Laird, Carol Brennan: 'Software Measurement and Estimation – A Practical Approach', (John Wiley & Sons, 2006).

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

[39] Karner, G.: 'Metrics for Objectory', Diploma thesis, University of Linkoping, Sweden, No. LiTH-IDA-Ex-9344:21, December 1993.

[40] BenteAnda, HegeDreiem, Dag, I. K., SjobergandMagne Jorgensen: 'Estimating Software development Effort based on Use Cases – Experiences from Industry', www.idi.ntnu.no/emner/tdt4290/docs/faglig/uml2001-anda.pdf. Accessed January 2007.

[41] Bushra Jamil, JavedFerzund, AsmaBatool, ShaistaGhafoor, Empirical Validation of Relational Database Metrics for Effort Estimation, 6th International Conference on Network Computing, IEEE, 11-13 May 2010, pp-1-5

[42] BenteAnda, HegeDreiem, Dag, I. K., SjobergandMagne Jorgensen: 'Estimating Software development Effort based on Use Cases – Experiences from Industry', www.idi.ntnu.no/emner/tdt4290/docs/faglig/uml2001-anda.pdf. Accessed January 2007.

[43] Edward, R., Carroll: 'Estimating Software Based on Use Case Points', OOPSALA '05, San Diego, California, October 2005, pp.257-26.

[44] Kusumoto, S., Matukawa, F., Inoue, K.: 'Estimating Effort by Use Case Points: Method, Tool and Case Study', Proc.10th International Symposium on Software Metrics (METRICS'04), September 2004, pp. 292-299.

[45] Suresh Nageswaran: 'Test Effort Estimation using Use Case Points', Quality Week, San Fransisco, June 2001, pp.1-6.

[46] Roy, K., Clemmons: 'Project Estimation with Use Case Points', Cross Talk-The Journal of Defence Software Engineering, February 2006, pp:18-22.