An Experiment to Improve Cost Estimation and Project Tracking for Software and Systems Integration Projects

Brian Chatters, ICL, Wenlock Way, West Gorton, Manchester, M12 5DR, UK
(brian.chatters@icl.com); fax: (+44) 161 230 5770
Peter Henderson, University of Southampton, UK (peter@ecs.southampton.ac.uk)
Chris Rostron, ICL, Manchester, UK (chris.rostron@icl.com)

Keywords: Software development process, process improvement, metrics, empirical studies, systems integration

Abstract

It is becoming increasingly difficult to predict the resources, costs, and time scales for the integration of software and systems built from a number of components supplied by third parties. Many cost models use the concept of product size as the prime driver for cost estimation but our experience has shown that the supplied quality of components and the required quality of the integrated systems are becoming the dominant factors affecting the costs and time scales of many projects today. ICL has undertaken an experiment using an alternative life cycle model, known as the Cellular Manufacturing Process Model (CMPM), which describes how a product is integrated from its constituent components. The experiment has helped to develop a method and a set of metrics to improve cost estimation and project tracking.

1. Introduction

In order to remain competitive, ICL (as well as many other companies) needs to continually improve its predictability of costs and schedules for integration projects, to reduce time to market and to reduce costs without detriment to the quality of the products. The developments of our complex software and systems increasingly rely on using commodity components and collaborations as a way to meet these business objectives. The ability to accurately predict effort and time scales and the ability to keep within budget is becoming increasingly difficult in such projects.

This paper documents the results and conclusions of SIMMER (Software and Systems Integration Modelling Metrics and Risks); an ESSI funded Process Improvement Experiment. The specific purposes of the experiment were to develop a method to apply the "Cellular Manufacturing Process Model" (CMPM) technology to a business critical, live software and systems integration project and to develop and tailor the model and associated metrics to improve the project management processes.

The application of CMPM defines an extended set of cost drivers, which include measures of the quality of supplied components and measures of the progress towards achievement of the target delivered quality.

2. Starting scenario

Over recent years, software and systems development has become more complex and the trend has been towards the use of bought-in components. There is an expectation that, by buying in components, the time and cost to bring a system to market can be significantly reduced. However, the use of third party components introduces a number of unknowns into the development activities, increases risks and jeopardises delivered quality. This fact is particularly true when the component is software that may not have previously been exposed to the specific operational environment, often leading to performance problems.

Traditionally, ICL has used the V-diagram waterfall life cycle model to plan and control the development of platform software and systems. Estimates of effort and time scales are based on an understanding of the architecture of the solution, and expert opinion of the degree of difficulty and potential problems likely to be encountered during the integration activities. The availability of resources and the team size is also taken into account when making the estimates.

A number of alternative life cycle models and development methods, [3] and [13] are examples, are now in existence but none of them adequately address the specific issue of managing complex integration of third-party supplied components. The traditional method of establishing a work breakdown structure using life cycle models that describe process flow fails to aid understanding of the specific issues associated with supplied components.

The CMPM [5], developed jointly by ICL and Southampton University, is a more appropriate model for the changing business. It provides a better means to capture and metricate the interfaces between the contributing suppliers and integration activities. The model enables earlier and more comprehensive understanding of the integration requirements and issues. This understanding ensures better verification and validation of the integrated software and systems and helps to mitigate against potential risks.

3. The cellular manufacturing process model

3.1. Rationale

COCOMO [4] is a model for estimating effort (W) from product size (S) for a number of different types of development projects:

$$W = f(S)$$

The model has been adapted [1] to address the impact of COTS (commercial-off-the-shelf) components. CMPM also addresses this problem space but extends it to cater for systems (hardware and software) and for components designed and built in collaboration with third parties. The uncertainty of the quality of supplied components and the impact on schedules of clearing problems late in the development life cycle [7] has led to the establishment of additional cost drivers to help predict and manage projects.

Our conjecture is that the quality of supplied components (Q) and the target quality of the integrated product (P) will have a significant impact on project costs and need to be included explicitly in any predictive model:

$$W = f(Q,P,S)$$

The model will demonstrate the behaviour that, if the target P increases, either W will increase or Q needs to be increased. Similarly, if S increases, then W or Q will need to be increased in order to achieve the same level of output quality.

3.2. Description of the CMPM

The CMPM is defined to be a set of “Manufacturing Cells” with the relationships between the cells described as a set of metrics. The model is based on network models of software development [9], and on the “value chain” model [12].

The model represents a view of products integrated from a mixture of bought-in and self-built components (Fig.1). In this context, system integration is defined to be those activities which identify (and specify) components and develop “glue” to bind them. For such components, the choice of one influences or restricts the choice of the others. The nature and quantity of glue required is a significant property of the system design. Each integration activity is defined as a cell within the CMPM. The model is clearly hierarchical. Each cell can be a component in a higher-level integration activity. Each component used by a cell can, itself, have been integrated from lower level components, either by in-house development or supplied by a third party.

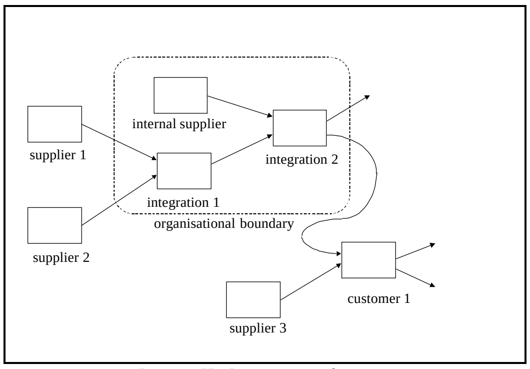

Figure.1: The Cellular Manufacturing Process

Products with hierarchical structures lend themselves to being developed and built in a network of manufacturing cells. Each cell is responsible for one level of integration. The cell receives components from suppliers, makes some components locally, glues the components into an integrated product (which is tested to output standards) and ships the integrated product to a customer or to the cell performing the next level of integration. The traditional software development process can be defined as an integration cell within the definition of the CMPM. It makes all components (lines of code) in-house and glues (compiles and builds) them into a software module or software product.

The behaviour within a cell can be as formalised or as ad hoc as the business or product demands. The measurement regime of the CMPM (the metrics) is not dependent upon detailed knowledge of how each cell performs its integration tasks, only how it meets its external obligations.

3.3. Metrics within CMPM

Six metrics are defined for each cell within the CMPM network (fig.2). Q, P, and S are cost drivers and are used to determine the effort required to complete a project. T is determined from W by taking available resources, N, into account.

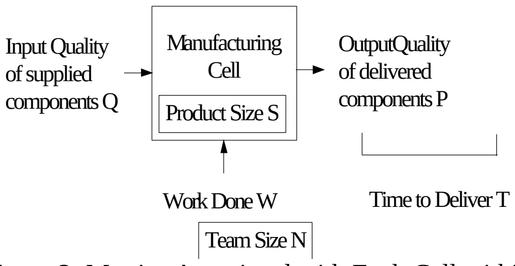
Input Quality of supplied components Q
Manufacturing Cell
Product Size S
OutputQuality of delivered components P
Work Done W
Team Size N
Time to Deliver T

Figure.2: Metrics Associated with Each Cell within CMPM

W, T, and N are defined within COCOMO [4]. The definitions of the cost drivers are:

- Q: The proportion of the requirements on the input components that are met without cost being incurred in the integration cell.
- P: The proportion of the customer requirements on the integrated product that are met by the delivered product.
- S: The size of the delivered and consumed components in “Standard Integration Units” (SIU’s) - see 4.4.

Q and P are derivatives of the idea of “probability of perfection”, advocated by Voas and Miller [14] and further developed by Bertolino and Strigini [2].

Clearly, the quality metrics are directly related to project risks. The poorer the input quality, the greater the risk of encountering unforeseen problems during integration. Similarly, the poorer the delivered quality, the greater the risk of customers encountering unforeseen problems.

4. The experiment

The aim of our experiment is to develop practical metrics that support project management and process improvement, based on the ideas of Grady [8]. The metrics are designed to demonstrate the following properties [6]:

- They are meaningful to the baseline project
- They are supported by data that is readily available
- They allow the behaviour postulated by the theoretical model to be tested against real world situations
- They enable convergence between the real world behaviour and that predicted by the model.

Practical metrics are defined that approximate to the theoretical cost drivers Q, P, and S. They measure actual behaviour within the baseline project. The experiment evolves the definitions of the real world metrics and uses them to verify or refine the behaviour postulated by the theoretical model.

4.1. Description of the baseline project

The experiment was undertaken by working closely with the project staff of the baseline project. The baseline project is part of a broader programme aimed at providing platforms, which exploit emerging technologies and meet the future needs of ICL’s customer base. A number of systems management products are included in the system and a minimal amount of non-invasive glue is developed to make them easier to use and to improve their ARM (availability, reliability, maintainability) characteristics.

The project is split into a number of integration sub-projects with an average size of eight people. The software is a combination of COTS products, in-house development, and collaborative development with partners. Software products, for example for backup, performance monitoring, printing and event management, are included. The suppliers of the relevant hardware modules provide platform specific software products (e.g. peripheral drivers). Regular, incremental, deliveries of the system are made to the customer base. Each sub-project is responsible for the delivery of an incremental release of the evolving system.

4.2. Baseline data

The baseline data (table.1) was collected from completed projects prior to the experiment. The only data available were the total effort spent and the duration of the development activities.

	Actual			*Forecast*				
Release	**W_a**	**T_a**	**N_a**	**W_e**	**T_e**	**N_e**	**$(W_a/W_e)\%$**	**$(T_a/T_e)\%$**
release a	2284	149	28	1320	60	28	173	248
release b	440	84	7.9	576	40	18	76	210
release c	2758	213	15	1512	140	14	183	152
release d	793.4	213	4.2	640	140	4.5	124	152

Table.1: Baseline Data

These measurements can be compared to the original estimates for each release. The data show that actual effort varies from 76% (24% under-spend) to 183% (83% overspend) of predictions whereas duration varies from 152% (52% overrun) to 248% (148% overrun). Even when the project under-spent against original estimates of effort, its elapsed time was more than twice the estimate. Part of the reason for this discrepancy is that, due to slips in “release a”, the resources originally assigned to “release b” were unavailable. Compare N_e against N_a. However, major problems caused by poor quality of supplied components also had a significant impact. Anecdotal evidence from other projects within the organisation confirms the generalisation that most projects overrun by 50% - 100%. Note that these dates are used for internal planning and do not reflect commitments made to customers.

4.3. Overview of the cost estimation and project tracking method

Fig.3 illustrates the method. The first step is to define the project as a set of inter-related manufacturing cells using the CMPM, each cell representing the integration of a number of components that produce an aggregated (part) system that feeds into another integration cell or delivers the full system to a customer. Then, for each cell, estimate values for the cost drivers Q, P, and S, using the checklists and historical data in the experience database. The cost drivers are used to predict effort (W) and time scales (T), taking availability of resources into account.

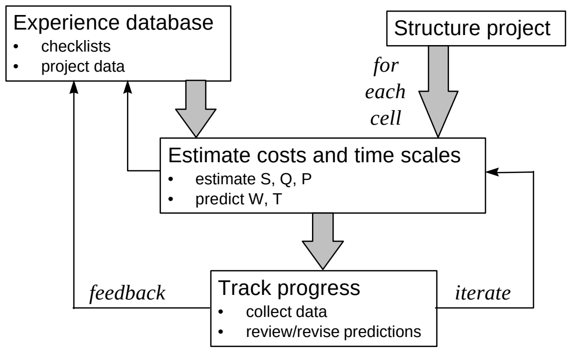

Figure.3: Overview of the cost estimation and project tracking process

Data is collected at regular intervals throughout the life of the project on resources consumed, the number of units of size completed, the quality of the supplied components and the progress towards the target delivered quality. The estimated values for the cost drivers are reviewed and adjusted, as appropriate. Similarly, the algorithm used to predict effort from the cost drivers is adjusted, if necessary, based on comparing actual effort spent to date against original predictions. Outstanding effort and time scales are re-predicted and management action is taken, as appropriate, if there are deviations against the plan.

4.4. Definitions of practical metrics

4.4.1 Size (S): the number of work products created or used by an integration cell. A standard unit of size, the "Standard Integration Unit" (SIU) is used to calibrate each work product. The experiment introduces a conversion factor to calculate the number of SIU's for each work product - see 4.6. The work products to be produced by an integration cell are classified into five categories (table.2). The numbers in the first column are examples of values from a typical project and the number in the second column are these values converted to SIU's.

			SIU's
Baseline documents			
Produce SD,RD,CD,ED,Plan, Config Documents etc			
1. No of documents required		3	30
	Subtotal	3	30
Project infrastructure			
Define, acquire and build			
1. Test equipment (rigs, test beds, etc)		0	0
2. Standard and non-standard IT platforms		2	40
(for development, management, system builds, etc.)			
3. Networks		0	0
4. Configuration management system		0	0
5. Document control system		0	0
6. Problem management system		0	0
	Subtotal	2	40
Components			
Make, reuse (including COTS products), or obtain supply of			
1. Boards and adapters		0	0
2. Subsystems (including firmware)		0	0
3. Software products		6	48
4. Cabinetry		0	0
5. Customer documentation		3	24
6. Release packaging		3	24
	Subtotal	12	96
System tests			
Develop and run			
1. Tests (can be enumerated from test schedules)		80	160
	Subtotal	80	160
Delivered system requirements			
Complete			
1. Items in the release assessment checklist		162	162
	Subtotal	162	162
	Total		488

Table.2: Checklist of generic work products used to estimate size

4.4.2. Input quality (Q): the number of supplied components introduced without faults divided by the total number of supplied components.
4.4.3. Delivered quality (P): the number of completed work products that are not faulty as a percentage of the total number of work products created or consumed by a cell.
4.4.4. Effort (W): the number of person days consumed by the project (as recorded in the time recording system). W is broken down into:

- Effort needed to integrate the work products (both delivered and consumed by the cell)
- Effort needed to analyse and resolve problems raised by activities from within the cell
- Effort needed to repeat tasks due to faulty work products.

4.5. Application of the method to the baseline project

4.5.1. Structure project. An initial attempt was made to structure the whole of the baseline project as a set of inter-related cells, each cell representing the major components that are integrated into the target system [5]. However, it was not possible to relate the data associated with the behaviour of the cells to individual releases. Consequently, it was not possible to define common behaviour in terms of checklist items or in terms of the

CMPM metrics. To overcome these problems, cells were defined to cover all the activities required to deliver an incremental release to the customer base. This alternative approach enabled the use of CMPM to be simplified. A single cell became the unit of planning and the interactions between other cells within the baseline project did not need to be considered. Consequently, consistent behaviour was observed between different cells enabling comparison between cells and enabling the development of generic checklists.

Learning points. Factors affecting the choice of cells are:

- Each cell needs a formally controlled deliverable - e.g. a customer release which may be an increment towards the final solution. This factor is the most critical when deciding on the cellular structure.
- Projects need to run for at least four months to get benefit from the method. Team sizes need to be between about four and twenty people. If less, the individual variations in performance and availability can adversely affect predictions of available effort (e.g. when holidays, training, etc. are taken into account). If more, it becomes increasingly more difficult to collect data and assess the impact on the progress of the project. For large projects, the activities need to be broken down into a number of cells of about eight people.

4.5.2. Estimate costs and time scales. Initial checklists of work products to facilitate the estimation of the cost drivers were constructed from the base practices defined by the SPICE reference model [11]. The checklists were verified and refined by analysing the experience from two pilot sub-projects. The number of checklist items relevant to a specific cell was used to derive S. Relative sizes of the work products between the different categories are defined in the metrics database, allowing total size to be converted into SIU's. Q was determined by counting the number of problems caused by poor input quality and estimating the percentage of additional effort needed to address them. In this definition, the broader scope of problems is considered - e.g. it includes issues caused by supply not being delivered on time, or exceeding costs, or failing to meet its specification. The costs include any repeat work that needs to be undertaken, such as re-installation of corrected components or re-testing of the system. Actual effort on previous projects is held in the database as examples. At this stage of development of the method, output quality P is defined to be 100%. The only acceptable deviation is to allow reduced functionality to maintain delivery within the window of opportunity (this deviation may not be acceptable for projects that are working to a customer-defined specification of requirements). A variation in the delivered quality would manifest itself as a change to the estimate of size. The predicted effort needed to complete the project is that calculated by multiplying the number of SIU's estimated for S by the cost per SIU and then adding the additional cost for poor input quality. The elapsed time of the project can then be estimated based on available resources. Projects can over-rule any values if they believe that the resultant cost estimation is unrealistic.

Learning points.

- The main benefit of the method at this stage of its development is not that it will give a value for effort but, rather, that it provides a structured approach to defining the work products and associated costs needed to complete a deliverable. In some cases (e.g. "system test" and "delivered system requirements" categories), there is sufficient historical information to have more confidence in the estimation process.
- The key to the success of the revised process is that the cost drivers of size and input quality are used to control the progress of the project. Thus, attention is given to managing the causes rather than the effect (overspend and overrun). Improved planning helps to mitigate against potential risks to achieving the project's goals and the cost drivers help to focus on those things that need special attention.
- Measurements of the effort needed to manage poor input quality enable the performance of the suppliers to be monitored and provides a catalyst for process improvements to improve the quality of the incoming components.

4.5.3. Track progress. Data is collected as part of the existing management review process (table.3). "PONC" is the abbreviation used to denote the additional effort due to poor input quality. If there is a mismatch between actual performance and estimates, it will be due to one of three reasons (these reasons are not mutually exclusive)

- The effort needed to complete the tasks or to achieve the required output quality is different to that predicted by the model. In this case, the project re-estimates outstanding effort and records new values to be fed back into the database.
- The estimated size is incorrect - either requirements have changed or additional, unpredicted work products are required. If the latter, the work products are added to the checklist of items within the generic categories.
- The input quality is different to that predicted. Actual counts of problems and effort expended are recorded.

In all cases, the outstanding effort is re-estimated and time scales are re-computed.

PROJECT		Forecast				Actual						
Start Date	End Date	W	S	ponc	Probs in	W	T	N	S	ponc	Probs in	Probs o/s
Plan	**T = 82**	**205**	**256**	**10.3**	**6**							
1-Jun-98	30-Jun-98	**205**	**256**	**10.3**	**6**	80.5	17	4.0	24	3.8	2	2
1-Jul-98	31-Jul-98	**205**	**256**	**10.3**	**6**	56.5	23	3.0	28	17	4	5
1-Aug-98	31-Aug-98	**205**	**256**	**10.3**	**6**	32	20	1.5	18	8	2	6
1-Sep-98	30-Sep-98	**245**	**256**	**10.3**	**6**	65	22	3.5	28	12	3	8
1-Oct-98	31-Oct-98	**285**	**256**	**10.3**	**6**	38	22	1.5	18	6.4	0	8
1-Nov-98	30-Nov-98	**305**	**256**	**10.3**	**6**	52	21	2.5	126	0	0	5
1-Dec-98	31-Dec-98	**345**	**256**	**10.3**	**6**	38	18	1.5	14	6	0	2
						362	**143**	**2.5**	**256**	**53.2**	**11**	

Table.3: Sample data from pilot sub-project

Learning points.

- The two pilot projects made their original estimates without reference to historical data and the results clearly indicate big discrepancies between predicted and actual performance. In addition, there is little consistency between the two cells. It is vital that the experience database is populated with historical data and that the data is used to refine the checklists and to improve the predictability for future projects.
- Although major problems may be encountered by an integration activity, such problems have traditionally been solved "in-house" by the project engineers. However, if such a problem is encountered with a supplied component, then resolution becomes the responsibility of the supplier and, if the problem is critical (that is, it must be resolved before the system can be delivered), then significant delays can occur in the project's schedule. Although the number of such problems may be small, their impact on a six-month schedule (say) can cause a delay of three months, with disastrous affect.

Estimating the number of major problems that they expect to encounter helps project managers to consider the impact on the effort needed to manage poor input quality. The actual number of problems is then tracked. It is believed that there is a relationship between input quality and the number of such critical problems but more work is required to understand this relationship.

4.6. Results of the experiment

- The overall objective of demonstrating that the CMPM can be applied to business critical software and systems integration project is proven.
- The approach has enabled an improved cost estimation and project tracking process to be established and is capable of being deployed for all projects within the organisation, not just for the baseline project. Some quotes from project managers who are using the method:

  "It feels right and is worth using"

  "It is relevant and lightweight"
- The results have been presented to a number of project managers. There is positive support for full deployment of the method.
- The experience database continues to be populated and results to date have enabled SIU values to be assigned for each of the categories of work products (table.4). Current experience gives a value of 0.7 net person days per SIU. Further, the cost of poor quality of supplied components is typically 15% - 30%.
-

Category	*No. of SIU's / work product*
Baseline documents	10
Project infrastructure	20
Components	8
System tests	2
Delivered system requirements	1

Table.4: Conversion of work product sizes to SIU's

- It is too early to quantify improvements to predicted costs and time scales although it is anticipated that they will be substantial. Such quantification requires the improved process to be in operation for about 12 months.
- A simple formula has been derived to estimate total effort required, based on the cost drivers. Work continues to refine a mathematical model that simulates the observed behaviour of incremental spend on resources throughout the project life.
- Work continues to quantify the impact of critical problems on costs and schedules.

5. Application of the method to SME's

The method is clearly applicable to small projects and, hence, could be used within any organisation. Indeed, we have found that the best application was for teams of between 4 to 20 people and projects of minimum duration of about four months. The value of the method is that it creates a detailed understanding of what constitutes "size" within an integration project and an understanding of the impact of poor input quality. The checklists of work products, the values for converting size to SIU's, and the cost of poor input quality can be used as an initial starting values for cost estimation using a proxy-based estimation approach [10]. The method also introduces a mechanism to support continuous learning to enable these initial values to be refined to reflect observed behaviour and, hence, improve the accuracy of predictions. Of course, with a large organisation, it will be quicker to populate the experience database because the organisation will be involved in more projects.

6. Lessons learned

- The experiment only works with the full collaboration from the project.

- It is best to build the experiment around the issues that projects currently care about and are therefore likely to have useful data. You then need to get disciplines in place to capture and analyse such data.
- The metrics need to be simple and easy to collect. Data collection needs to be established as part of normal business; carried out by project staff.
- Metrics provide objectivity into the project management decisions. The act of measuring alone can bring about improvement.
- There is clear benefit in using the extended metrics set (Q, P, and S) in managing integration projects. The issues to be managed in an integration project are much broader than product problems (bugs) and all problems that impact costs significantly need to be considered.

7. Future plans

- Further experimentation will be undertaken to refine the mathematical formula to simulate the behaviour of spend over time throughout the development life cycle. When such a formula is derived, it will be possible to produce a curve of predicted behaviour against which actual performance can be monitored.
- The checklist items and the relative costs will be refined as more projects are completed. It is recognised that a range of relative costs may need to be used for different types of projects. Thus, a classification scheme similar to that devised by COCOMO may need to be developed.
- The method will be made available across the organisation. Deployment will be supported by a guidebook, PC tools, and facilitated workshops.

8. Conclusions

The experiment has successfully demonstrated that the CMPM provides a more effective means of planning and tracking software and systems integration projects that are dependent upon components supplied by third parties.

Involvement of staff from the baseline project has ensured that the method is acceptable and can build on existing practices.

The method is sufficiently flexible to be deployed across any organisation that integrates systems from components supplied by third parties, be they COTS or collaborative developments. It is particularly applicable to small to medium size projects.

References

1. Abts Christopher M., COTS Software Integration Cost Modelling Study, University of Southern California, June 1997

2. Bertolino Antonia, Strigini Lorenzo, On the Use of Testability Measures for Dependability Assessment, IEEE transaction on Software Engineering, Vol. 22 No. 2, pp 97 - 108, Feb 1996

3. Boehm B.W., A Spiral Model of Software Development and Enhancement, in: IEEE Computing 21(5), pp. 61-72, 1988

4. Boehm, B. W., et al, Cost model for future life-cycle processes: COCOMO 2.0, Annals of Software Engineering, 1995

5. Chatters B.W., Henderson P., Rostron C.J., The Cellular Manufacturing Process Model: Planning a Complex Software And Systems Integration Project, in: Proceedings of the European Software Measurement Conference, pp. 559-564, Technologisch Instituut vzw, 1998

6. Fenton Norman, Pfleeger Shari Lawrence, Glass Robert L., Science and Substance: A Challenge to Software Engineers, IEEE Software, pp 86 - 95, July 1994

7. Grady Robert B., Practical Software Metrics for Project Management and Process Improvement, Prentice Hall, 1992

8. Grady Robert B., Successfully Applying Software Metrics, IEEE Computer Vol. 27 No. 9, pp 18 - 25, 1994

9. Humphrey W.S., Managing the Software Process, Addison-Wesley, 1990

10. Humphrey Watts S., The Personal Software Process and Personal Project Estimating, American Programmer, vol. 9, no. 6, June 1996, pp. 2-15.

11. ISO/IEC TR 15504-1998, Information Technology - Software Process Assessment, Part 2: A Reference Model for Processes and Process Capability, 1998

12. Porter M.E., Competitive Advantage: Creating and Sustaining Superior Performance, The Free Press, New York, 1985

13. Stapleton J., DSDM: Dynamic Systems Development Method, Addison-Wesley, UK, 1997

14. Voas Jeffery M., Miller Keith W.,Software Testability: The New Verification, IEEE Software, pp 17 -28, May 1995