Software Project Profitability Analysis Using Temporal Probabilistic Reasoning; an Empirical Study with the CASSE Framework

Joseph K. Balikuddembe[1], Isaac O. Osunmakinde[2], Antoine Bagula[3]

Department of Computer Science, University of Cape Town
Private Bag Rondebosch 7701, South Africa
{jbalikud, segun, bagula}@cs.uct.ac.za

Abstract. Undertaking adequate risk management by understanding project requirements and ensuring that viable estimates are made on software projects require extensive application and sophisticated techniques of analysis and interpretation. Informative techniques and feedback mechanisms that help to assess how well and efficiently a specific development methodology is performing are still scanty. Analyzing project tasks would enhance how well individual tasks are estimated, how well they are defined, and whether items are completed on-time and on-budget. In this paper, we propose a temporal probabilistic model that addresses feedback control mechanisms in project planning using the Complex Adaptive Systems Software Engineering framework (CASSE). We have tested our approach in industry with a software development company in South Africa on two commercial project evaluations. Our preliminary results show that the temporal probabilistic model of the framework demonstrably enhances practitioners' understanding in managing software projects profitably - hence increasing business sustainability and management.

Keywords: Requirement Engineering, Project Management, Project Profitability, Probabilistic Modeling, CASSE Framework

1. Introduction

Many projects fail in their expected scope, benefits, cost and time targets due to inadequate technical skills or management quality [1]. Conventionally, a project should deliver agreed-upon functionality on time and within budget, governed by the overall objective of maximizing the net present value of all cash flows of that project. However, in the practice of software engineering today, it is still a central issue that total budget and human resources are frequently not managed optimally to bring about successful project completion and optimal production [2].

Recent studies show that most developments are more expensive than projected, with processes rendered more difficult through series of problems including poor project management, cost and schedule overruns, poor quality software and under-motivated developers [3]. Failure to anticipate these uncertainties and to link the planning capabilities of different sections of project management, particularly with budget op-

timization while analyzing profitability, leads to disconnected silos of information which may limit the potential benefits of the whole project.

Our approach aims at establishing a link between management capabilities. It examines the strategic, process and organizational issues that enterprises need to address in order to implement world-class development strategies that generate value. We have perceived a compelling need for organizations to link their projects to their corporate strategies and to train advanced project management skills, which will better enable projects to survive shifting organizational priorities. Accordingly, and to meet this need, our framework is based on the notion that economic concepts, models and tools can help to advance understanding and improvement of the development of software and the processes that produce it.

2. Rationale

The planning decision is essentially a strategic process which requires planning for requirements of varied resources and types relating to every time period of the planning horizon [4]. Inevitably, failure to interpret different complexities within the planning process results in huge project bottlenecks. Misunderstanding and management of software project risks lead to a variety of problems, including cost and schedule overruns, unmet user requirements, and the production of systems that are unused or do not deliver business value [5]. In some instances, projects have been characterized by project estimates being made to appear lower than they would be in reality, with a hope of getting final project approval. Such a predicament affects the overall development portfolio.

How can we therefore integrate value-based methods into project planning and control for strategic business valuation so as to ensure that software projects are successful and profitable? Can such feedback arising out of project evaluation assist in re-aligning our development process so as to produce profitability in our engineering processes? In this work, we aim at answering these questions so as to facilitate the realignment and optimization of software development processes.

3. Coordination and Management

3.1. Project coordination in the agile environment

In this paradigm shift of agile software development, software engineering project work is increasingly becoming a highly cooperative activity, and promises to continue in this way. Coordinating projects demands managerial control over time and resources. The more complex the project is, the more expensive and time consuming it becomes [6]. Project planning should consider the availability of good resources along with commitments to customers and the organization. A well-defined plan where all facts and practical aspects have been determined is the key to success.

Planning and executing are two different functions. Planning requires that consultations be held with those who are actually going to execute the plans, cross-checking facts and calculating risks using a good dose of practicality. If commitments to customers are broken, whatever the reason may be, it becomes one cause for the client to drift away from the project. Cost factors should be considered and balanced, and previous experience and best practices taken into account so as to refine the development approach.

Tracking tasks, compiling costs and expenses, and managing people involved in business projects are all made more manageable when using a project scheduling technique that views requirements as value-generating tasks with complex interdependencies. The way we schedule our resources on the project, mandated with various tasks, determines their productivity and overall performance and ultimately, the success of the project [7]. The objective of the agile development process is to create software development and maintenance processes that maximize the agility of application development. Project failure directly impacts upon customers by lowering customer satisfaction, while giving competitive advantage to our competitors. A collective and systematic project management approach is required to substantially increase the project's success rate and help to minimize the loss of benefits.

Projects are often driven and defined by customers. Management needs to review all existing projects in relation to strategy and if they are not contributing to strategy, they should be dropped. Benefits management which applies project profitability analysis as well as business strategic evaluation is therefore important. Organizations need to make the business case for the benefits that each project can bring. Benefits analysis and evaluation need to be part of the metrics that senior management use to measure the business. Success of any project must be the objective for those individuals who are making decisions for the business.

3.2 Deriving project value tasks

Projects can be driven off the Use-case list and every step in the life cycle derives benefit from this list. If a project is approached from this perspective, Use-case offers rewards for each group, including analysts, project managers, clients, testers, designers, estimators and programmers. What drives all scheduling are the project requirements. Requirements play a vital role in enhancing value creation in the properties and value attributes of an entire project. The way we allocate time to these requirements must derive value in order to mitigate the unforeseen challenges on the project. [8]. If a project is likely to operate within the break-even space, without necessarily making profit, it is imperative that experienced programmers are scheduled while the inexperienced ones undertake the testing. If the project is likely to make a profit, it is essential that a combination of experienced programmers with a juniors peer program so as to disseminate knowledge and build human capital development on the project.

It is easier to quantify work done per functionality – which may occasionally be synonymous with requirements – than entire components at a time. Analyzing and achieving functionality in a bottom-up, micro value to macro system value approach, derives value for all stakeholders on the project and usually accelerates functional acceptability and project signoff [9]. By using this approach you can gauge how pro-

ductive the resources on the project are and how the project is progressing. Equally, milestones can be evaluated and attained easily and issues that usually cause IT projects to fail, like inability to deliver products on time, would be mitigated.

Most researchers, in particular Nagappan [10] and Kan [11], agree that poor requirements are the biggest single source of defects in software projects. Poor requirements lead to weak estimates and over-ambitious project plans. There is enough evidence to suggest that during the early stages of the development process, most software projects are already in trouble, that project managers are overly optimistic in their perceptions, and that executives receive status reports very different from reality, depending on the risk level of the project and the amount of bias applied by the project manager [12]. Key findings suggest that executives should be skeptical of favorable status reports. Rather, they should concentrate on decreasing bias if they are to improve the accuracy of project management and reporting [13].

The software business is surely about satisfying and sustaining our clients. We can only achieve this if we firstly understand project requirements and secondly improve our reporting techniques and mechanisms on each project. However, many questions still arise as we turn our endeavours to realizing this. If we understand the client requirements well, how can we selectively implement requirements that generate value to us as a business and the clients we serve on the project? How can we analytically verify those requirements that increase our clientele commitment to the project while accelerating milestone acceptance? How can our scheduling and resource allocation mechanism help us realize value on the project throughout the project lifecycle? Our approach aims at addressing these issues using temporal probabilistic modeling and reasoning as a guiding factor in the project design space.

4. Temporal Probabilistic Reasoning

4.1 The Fundamental Theory

A Bayesian belief network is formally defined as a directed acyclic graph (DAG) represented as G = {X(G), A(G)}, where X(G) = {X_1,…,X_n}, vertices (variables) of the graph G and $A(G) \subseteq X(G) \times X(G)$, set of arcs of G. The network requires discrete random values such that if there exists random variables X_1, . . ., X_n with each having a set of some values x_1, . . ., x_n then, their joint probability density distribution can be defined as shown in equation 1 but not over time. Suppose the variable X is represented as variable V then, $\pi(V_i)$ represents a set of probabilistic parent(s) of child V_i [14]. A parent variable otherwise refers to as *cause* has a dependency with a child variable known as *effect*. Every variable V with a combination of parent(s) values on the graph G captures probabilistic knowledge as conditional probability table (CPT). A variable without a parent encodes a marginal probability. Having a Bayesian network model in place, a probabilistic inference is required for reasoning about any software project situations using the Bayes' theorem [14].

We base our modelling on the Dynamic Bayesian Networks theory defined in various works [15]. We specifically utilise this model to evaluate project performance over

time by examining acceptance signoff of the various tasks on the project and how this acceptance pattern affects the overall profitability curve of the development process.

As literature suggests, Dynamic Bayesian Networks (DBN) can be understood to be interconnections of ordinary Bayesian networks over finite time steps as temporal measurements [15]. Temporal probabilistic modelling is an extension of ordinary Bayesian Networks, as defined by Bayesian Network proponents such as Muphy [16] and Choudhury [17]. Modelling is applied on key variables of interest in any domain of assessment.

Variables of a time step, usually referred to as frames, can have various impacts on the variables of the subsequent frames through temporality links across the frames. According to Russel [14], construction of the temporal model requires prior matrix, Pr (V_0); transition matrix, Pr ($V_t \mid V_{t-1}$); and sensor matrix, Pr ($E_t \mid V_t$) of state variables *V* and *E*. The three matrices can be estimated during the intra- or inter-frame learning of the model using parameter learning algorithms such as maximum likelihood estimate (MLE). The intra-frame learning estimates the conditional probability distributions (CPDs) for every time step *t,* while the inter-frame learns the CPDs over time. We describe these required matrices in equations 1 and 2 below.

The joint probability distribution for any frame of random variables V_1 to V_n at time *t* is given as:

$$\Pr(V_1,...,V_n) = \prod_{i=1}^{n} \Pr(V_i \mid \pi(V_i)) \tag{1}$$

The combined joint probability distribution for any temporal model to a finite time *t* is also described as:

$$\Pr(V_0, V_1,...,V_t, E_1,...,E_t) \tag{2}$$

$$= \Pr(V_0) \prod_{i=1}^{t} \Pr(V_i \mid V_{i-1}) \Pr(E_i \mid V_i)$$

These two foundational steps result in Emergent Situation Awareness (ESA) techniques used in this study.

4.2. The Emergent Situation Awareness Technology

The temporal modelling technique adopted in this work is the ESA initially proposed and developed by Osunmakinde [18]. ESA is an innovative technology, which realistically evolves temporal models and reveals what is currently happening over time in any domain of interest. One of its powerful features is its evolvement from Multivariate Time Series (MTS) data in the absence of domain experts.

ESA advances the algorithms of ordinary Bayesian Networks to evolve dynamically as it changes its network and the probabilistic distributions with time. It has 3 notable components: the learning algorithms, the probabilistic reasoner and the trend analyzer.

The learning component uses genetic algorithms to produce temporal Bayesian Networks, called frames, over the time steps from the MTS environments. The prob-

abilistic reasoner is the Bayesian inference engine, which executes the necessary forward and backward propagations through the links of the frames and generates probable results. The trend analyzer is an interface that generates *n*-dimensional transition matrices of knowledge, where *n* corresponds to the pieces of knowledge to be revealed, e.g. a transition matrix of target probabilities, a transition matrix of target parameter values, etc.

We utilized this approach in our study to answer the following questions on the software projects: [i] What is happening on the project(s)? [ii] Why is it happening? [iii] What will happen next? [iv] What can one do about it? The experimental results discussed in the next section detail the characteristics of the projects analyzed using this technique, while fostering managerial answers to the above questions.

5. Experimental analysis and interpretation

5.1. The ESA as Applied to Software Projects

The case study was carried out with one of the software development houses in South Africa. In this analysis, two commercial projects were selected. Project 1 started in November 2007 and was scheduled to end in March 2008, while Project 2 was scheduled to start in February 2008 and end in April 2008. We specifically wanted to understand project progress complexities and how they impact on the overall company revenue. In our study, profitability analysis implied evaluation of the amount of time taken to complete a given task. This implementation time determines the overall cost for that project since resource inputs are valued and paid for on an hourly basis. For example, if a task was scheduled to take 40 hours, and it overlaps to 60 hours, this implies that the company is incurring more time on completing this task than earlier on anticipated, thus eroding project profits or even setting the project into loss levels. If this task implementation time has overran and worse still it has not reached acceptance standards for the client, more time would be added to satisfy client acceptance standards. More time added would imply more cost to company and less on profit. If this pattern applies to various tasks on the project and there are mechanisms of detecting such performances (such as our model) behavior, we can already detect such complexities on even ongoing projects.

Given this background therefore, we looked at one project which was complete and another ongoing one. For the completed project, we were interested in understanding how tasks were quickly completed and how best they met client requirements to be acceptable for signoff. For the ongoing project, we wanted to analyze the possible risks that were slowing the project either in terms of task completion or acceptance bottlenecks. This was for the purpose of evaluating whether there was a need to realign the development process or optimize on resource allocation. Our model would help capture these evolving properties of the project.

Both projects selected were utilizing requirements of the same product-line, that is, utilizing core components of the development house. They only differed in visualization requirements and data-handling types. Both projects had inelastic budgets. Project schedule requirements differed depending on a number of externalities affecting

the projects. Both projects were developed in agility using eXtrem Programming and had to be executed in parallel from the point of project period intersection. All projects went through the requirements analysis phase where key project requirements and milestones were agreed upon. There was a 60% allocation rate of the resources assigned to execute tasks on both projects.

We looked at a 3-dimensional variable analysis of the results. For both projects, the key variables of interest examined entailed: requirement or task acceptance patterns, task completion patterns throughout the project duration and profitability analysis.

5.2. Task acceptance patterns

5.2.1 Project 1 acceptance analysis

Using our model, we analyzed how each project milestone was accepted by the client throughout the project duration. In order to increase client participation and commitment to the project, an incremental functional delivery and signoff development approach was utilized. In measuring the task acceptance pattern, we examined how each project performed on a monthly basis. Figure 1 below shows the results obtained on Project 1 in this dimension of acceptance criteria evaluation and validation pattern analysis. The X-Axis measure observation period while the Y-Axis measure the project acceptance percentages.

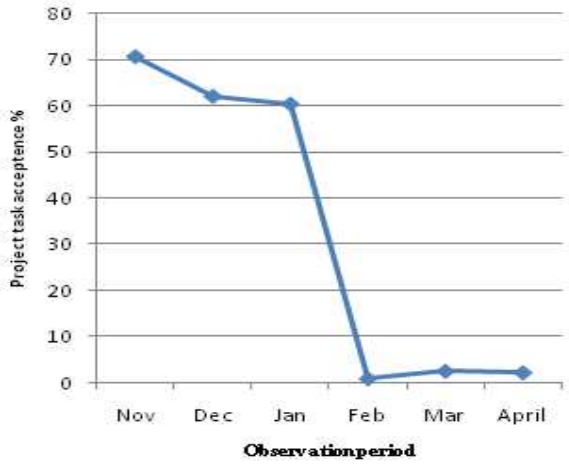

Fig. 1. Project 1 acceptance pattern

The graph above shows that for Project 1 in November, there was a 70.72% likelihood that task acceptance would not suffice in the first frame of the project. The 29.28% difference implied that earlier milestones in the project were accepted, such as project inception deliverables which were derived in the earlier part of the month. Given that requirement analysis, documentation and validation had to precede implementation, functional acceptance would therefore have a higher completion failure in

that time frame. As the project progressed in December, the validation failure decreased by 8.59% of the overall tasks in that time frame. This implies that as development started, project delivery maximization was being delayed at a rate of 8.59%. An optimal validation rate would be targeted to being higher than 37.87% since functionality required was mostly from the existing core components. In January, the delivery rate was not very significant either; it only increased by 1.67% from the previous time frame. However, as the February time frame set in, the validation rate increased to 98.99%. This shows that a great deal of effort was expended to ensure that the March deadline would be reached while all tasks were completed and signed off. The remaining 1.01% would only be for bug fixes while more attention was directed to Project 2.

5.2.2 Project 2 acceptance analysis

Figure 2 below shows the acceptance pattern evaluation of Project 2 in the study.

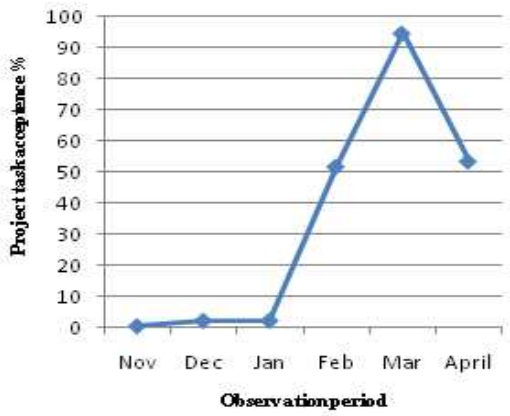

Fig. 2. Project 2 acceptance

At the start of Project 2 in February, the acceptance targets were improved to 48.25%. Perhaps lessons learned on Project 1 led to this improvement, or components development completed in Project 1 which were required by Project 2 led to this accelerated increase in performance. The March time frame reveals that the acceptance rate was improved by 42.94%, leading to a 94.69% likelihood of overall project acceptance in that time frame. However, in the April time frame, this performance seems to have dropped by 41.21%, leading to a 53.48% likelihood of project acceptance in this time frame.

These findings show that there is a strong need to manage the acceptance rate over and above 60% in each project time frame if a given project is to be signed off gracefully at completion, if all requirements will be accepted at signoff and if resources need to take up other roles on other projects that may begin before the present project is completed. Maintaining the project acceptance rates at the rate of 40% in a given time frame would increase the likelihood of a project being delayed due to unaccepted

tasks. Therefore, project managers need to evaluate project time frames with keen interest so as to mitigate project delays that may arise due to relative acceptance rate targets.

5.3 Task completion patterns

5.3.1 Project 1 completion analysis

In this measurement, we analyzed the task completion rate of each project in a given time frame. As shown in Figure 3 below, for Project 1 there was 40.88% likelihood that tasks would be completed over and above 70% at project inception in November. In December the completion rate likelihood seems to have decreased by 2.02%, with tasks only being completed between 31 – 45% on average.

Fig. 3: Completion rate analysis for project 1

In January the completion rate dropped by 3.61%, leading to a 46.51% likelihood of tasks being completed with a progress level of 31% on average, perhaps due to the festive and holiday season around the previous time frame. As the project tended towards completion in February, the task completion rate improved significantly at a rate of 1.1% per task, leading to 98.9% increment in task completion with a task progress average of 69%. In March the completion average dropped to 38%. This was the time for bug fixes and resources were executing other tasks on the next project in parallel.

The findings indicate that as the project tends towards completion, the task completion rate reduces considerably. This could be due to other project externalities on the project, such as delayed customer feedback on the deliverables. Although the acceptance rates rise in the last time frames, other project externalities affect the completion rate on final tasks preceding project signoff, hence impacting on our average profitability margins.

5.3.2 Project 2 completion analysis

Figure 4 below shows the results of the completion rate analysis done on Project 2 in this study.

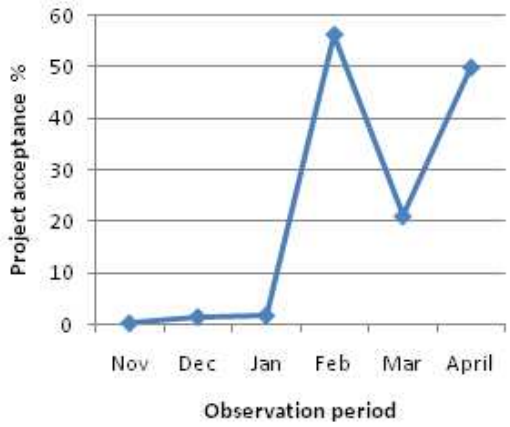

Fig. 4: Completion analysis for project 2

Although Project 2 started in March, our model shows that should the project have started in November, there would have been an average of 1.12% likelihood of task completion through till January. This can be attributed to the fact that since the same resources were working on both projects, they would not achieve higher completion margins on competing projects; hence completing one at a higher acceptance rate before embarking on the next one. At project inception in February, the project completion rate increased by 55.27% leading to a 69% completion progress rate on average. This can be explained perhaps with the fact that both projects required similar components. Given that Project 1's acceptance rate had already reached 97.37% around this time frame and that the functional baseline had already been realized, the acceleration of project completion for Project 2 tasks was invariably enhanced. In March, however, the completion rate dropped by almost 78.98% leading to an average task completion rate of 38%. This is perhaps due to the fact that resources were divided between both projects; as they fixed bugs on the other, they would also try to work on tasks on Project 2. The pattern seems to have changed in April after Project 1 was completed. The completion rate rose to 50%; implying that resources were now redirected to Project 2.

5.4 Profitability analysis patterns

5.4.1 Project 1 analysis

We examined how profitability fluctuated over time in the various time frames. Figure 5 below shows the results obtained on Project 1 under this evaluation. In the

November time frame, there was a 67.04% chance that profits would be realized in that time frame given the number of tasks and resources available. The 32.96% difference implies that profits on the tasks would be realized at such a threshold. In December, however, the profitability likelihood increased by 1.52%, leading to a 34.48% chance that maximum profit of R1600 would be realized on each task.

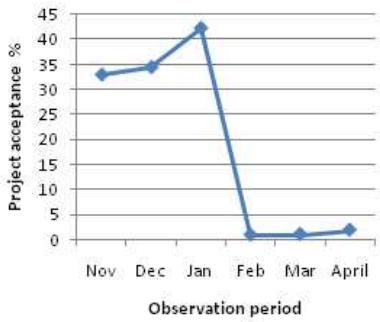

Fig. 5: Profitability on project 1

In January, however, the likelihood of making some profit on project tasks increased by 7.83% from the previous time frame. This led to an overall 42.31% profit realization likelihood. From February until the end of the project in March, the likelihood that profit margins would increase dropped significantly. In February for instance, there was a 99.1% chance that profits would be realized on tasks executed in that time frame but zero profits were attained. In March, the profitability likelihood dropped to 98.97%, implying that there was only a 1.03% chance of making R1 600 on each task in this time frame.

Although the tasks were being accepted as the project was progressing in the last three months, the profitability pattern seems to have been dropping considerably. This implies that more time was invested in completing the tasks to acceptance standards while compromising profit on this project.

5.4.2 Project 2 analysis

For Project 2, the profitability pattern seems to have been different, as shown in Figure 6 below.

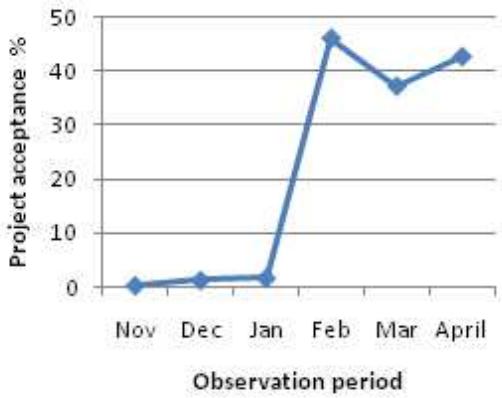

Fig 6: Project 2 profitability analysis

As the project started in February, there was a 46% chance that some profits would be realized in that time frame. However, the threshold was too low to realize any profit. In March, the likelihood of realizing R1600 profit per task executed in this time frame decreased by 8.86% to 37.13%. In April the pattern seems to have improved significantly, by 5.47% from the previous time frame, implying that there was a 42.61% chance that a profit margin of R1600 would be realized.

As the task acceptance rates increased in February and March, the profitability margins seem to have fluctuated considerably, with a 3.39% drop rate on the average R1600 task profitability projection.

Therefore, as we increase the acceptance rates in each time frame, we ought to take our profit margins into consideration. The likelihood that we can win customer trust in the first two time frames after project inception, by delivering validated tasks, implies that we are likely to decrease our average profitability rates by 3.39% in the subsequent time frames.

6. Implications and future work

The area of post project review as a key project management competence is still in its infancy requiring rigorous techniques of analysis. We have demonstrated that we can look at various aspects of the project, including profitability analysis using our technique. Our approach however, is an alternative solution that has been tested in the commercial environment on real life projects specifically embracing key business project values such as project profitability. Our technique is developed in such a way that it is extensive and pluggable, thus it can be generalized to other areas of Software Engineering and other disciplines.

Project variants including requirement or task acceptance, task completion throughout the project duration and profitability analysis patterns impact considerably, on development processes in the first place, and on resource allocation and man-

agement capabilities in the second. The order in which selected tasks are prioritized, executed, completed and validated is very important. Our results show that if we use such modeling techniques for ongoing project review processes, we can enhance project portfolio management in the following ways: by determining proper project estimation, by delivering projects on-time and on-budget, and by properly identifying key project requirements and risks.

The benefit of this approach is that measurement of overall project success is improved, thus enhancing a company's ability to handle project portfolio significantly, as it compels the company to consider the magnitude or complexity of software projects taken on. Consequently, if we extend our representation model to address changing tasks and requirements on a project, we can overcome some of the problems that arise under multi-project environments such as resource allocation, coordination or communications. The final goal of our work therefore is not only the improvement of software engineering as a discipline, but also the improvement in management of projects in order to derive value in development of IT projects.

Managers will be able to prioritize activities for the effective management of project completion, and shorten the planned critical path of projects by pruning critical path activities, by performing more activities in parallel, and/or by shortening the durations of critical path activities through adding resources, hence preparing for scheduling and resource planning effectively. The identification of the critical chain of events would make it possible to mitigate their negative effects since risk lists of projects can be generated as a result of sensitivity analysis.

Acknowledgment
Great thanks to Complex Adaptive Systems (Pty) Ltd for their support both technically and academically.

7. References

1. Zafra-Cabeza, A., Ridao M. A.,. Camacho, E. F.: Using a Risk-based Approach to Project Scheduling: A Case Illustration from Semiconductor Manufacturing. EJOR. 190, 708—723(2008)
2. Alba, E., Chicano, J.F.: Software Project Management with GAs. Info. Scie. 177, 2380—2401(2007)
3. Verner, J.M.: Quality Software Development: What do we Need to Improve in the Software Development Process?. In: WoSQ'08, (2008).
4. Gonçalves , J.F.,. Mendes, J.J.M., Resende, M.G.C.: A Genetic Algorithm for the Resource Constrained Multi-project Scheduling Problem. EJOR. 189, 1171—1190(2008)
5. Wallacea, L., Keilb, M., Raic, A.: Understanding Software Project Risk: a Cluster Analysis. Information & Management. 42, 115—125(2004)
6. Wateridge**,** J.: The Role of Configuration Management in the Development and Management of Information Systems/Technology (IS/IT) Projects. Int J. Proj Manag. 17 (4), 237—241(1999)
7. Zhang, H., Li, H., Tam, C.: Particle Swarm Optimization for Resource-constrained Project Scheduling. Int J Proj Manag. 24 (1), 83—92 (2006)

8. Fidel, R., Scholl, H. J., Liu, S., Unsworth,K.: Mobile Government Fieldwork: A Preliminary Study of Technological, Organizational, and Social Challenges. In: *8th* AICPS: dg.o '07, pp. 131—139 (2007)
9. Morales, A., Barra, L.: System Development Techniques for Small and Medium Size Installations. In: 15th SIGCPR, pp. 241--247 (1977)
10. Nagappan, N., Ball, T.:Use of Relative Code Churn Measures to Predict System Defect Density. In: 27th ICSE`05, pp. 284 –292(2005).
11. Kan**,** S. H**.:** Metrics and Models in Software Quality Engineering - 2nd Ed., Longman, Boston**,** (2002)
12. Snow, A.P., Keil, M.: The Challenge of Accurate Software Project Status Reporting: a Two-stage Model Incorporating Status Errors and Reporting Bias. IEEE Trans on Eng. Manag. 49(4), 491--504(2002)
13. Snow A.P., Keil, M.: The Challenges of Accurate Project Status Reporting. In: HICSS '01, 8, 8043 (2001).
14. Russell, S., Norvig, P.: Artificial Intelligence, A Modern Approach - 2nd Ed. Prentice Hall Series , New Jersey (2003).
15. An, X., Jutla, D., Cercone, N.: Privacy Intrusion Detection Using Dynamic Bayesian Networks. In: ICEC'06, pp. 208—215. ACM, New York (2006)
16. Murphy, K.: Dynamic Bayesian Networks Representation, Inference and Learning. PhD thesis, UC Berkeley, (2002)
17. Choudhury, T., . Rehg, J. M. . Pavlovic, V.: A. Pentland, Boosting and Structure Learning in Dynamic Bayesian Networks for Audio-visual Speaker Detection. In: ICPR`02, 3, pp. 789—794 (2002).
18. Osunmakinde, I.O., Potgieter, A.: Astute Decisions In Business Intelligence Using Temporal Probabilistic Reasoning. SAIMS (2008)