KEYWORDS Software Project Management Constraints Risk Management Resource-Constrained Project Scheduling Problem (RCPSP) Performance RISKIT

SOFTWARE PROJECT MANAGEMENT: THEORY OF CONSTRAINTS, Risk Management, AND PERFORMANCE EVALUATION

Mariam Salim
Master's degree student in the Computer Science Department at Qatar University

Abdelaziz Bouras, Ph.D.
ictQATAR Chair Professor at Qatar University
abdelaziz.bouras@qu.edu.qa

Antoine Asseman

Nada Ashqar

Eman Rezk

Nasser Alkhuzaei

Heba Dawoud

Rana Rihan

Souad Mecheter

Computer Science Department, College of Engineering, Qatar University, Doha, Qatar
P.O Box 2713

ABSTRACT

Constraints and risks are two critical factors that affect software project performance; more attention needs to be paid to constraints and risks in order to improve performance. In this paper, investigation will take place to determine the relation between those three factors. An enhanced model has been proposed to describe how these factors affect each other. As an application, the performance is examined for both open and closed source software projects in terms of some constraints and risk factors. Moreover, solutions for controlling both constraints and risks are provided. For constraints, project activities scheduling is enhanced using a genetic algorithm. For risks, RISKIT is briefly explained as a risk management methodology.

INTRODUCTION

Software development is a changing process which entails a great deal of reasonable thinking. Nowadays the process of software development is using more systematic and tool driven models. Thus, the process of software development will suffer from inflexibility, which may cause losing some of the required constraint. On the other hand, it will also suffer from higher risk probabilities, which negatively affects performance. Project managers should initially pay more attention to those constraints and risk factors and provide plans that continuously take them into account.

When project managers have enough experience in managing constraints and risks, they achieve the expected goals and outcomes. In this paper, Constraint Management, Risk Management and Performance will be investigated. Also some strategies will be provided to manage either constraints or risks in order to enhance the performance.

The first aspect of this study regards Constraints Management, the word constraint means any imposed limitation or restriction, software wise. Constraints are defined at the beginning of the project and can be related to time, budget or resources. To be able to efficiently manage constraints, a management paradigm should be followed, such as Theory of Constraints *(TOC)*.

Over the last 30 years, TOC has been successfully used in business and is now taught in many business schools and colleges. Apart from that, the TOC tools have been used by experts all over the world in the field of Health, Education, Science, Psychology and Personal Development. The benefit of using TOC thinking processes is that it has the ability to identify paradigm shifts when the rules and the assumptions don not change over time. Monitoring every assumption to make sure that it will evolve in reality is not an easy task, therefore being aware of the TOC logic tools can be quite advantageous **[4]**.

Moving to the second aspect of this study which is Risk Management, unlike constraints, risks cannot be identified at the beginning of the project, however they occur suddenly and may cause project failure if not correctly controlled.

One of the recent risk studies conducted considered about 13,000 projects, it was concluded that approximately 25% of the projects were either delayed or failed **[14]**. It has been noticed that a great number of problems occurring in the software development process are encountered because of the poor software risk management techniques or due to the absence of any such techniques at all. Risk management can be defined as the practice of determining the possible risks, studying and examining them, then taking the required steps to reduce and minimize the risk **[17]**.

Project managers always work on such limitations and risks in order to achieve the targeted performance. One of the main tasks a project manager should do regards providing details about the performance required in the project. This kind of information is crucial during project development to verify it will be able to achieve its objectives within the defined constraints.

To study how each of those factors affect each other, a new model has been proposed, called Risk, Performance and Constraints model *(RPC)*. In this model, the relationship between different categories of risks, Performance factors and constraints is clearly identified.

As an application of the explained model, the way constraints and risk factors affect performance in either open source or closed source software projects will be highlighted. The effects of these factors were studied through a survey conducted with a group of Software Development Managers in the state of Qatar.

After studying performance in terms of open source and closed source in light of risk management and constraint management factors, one risk management methodology is then briefly explained, and an innovative constraint management algorithm is also presented.

The Risk Management methodology chosen is RISKIT, standing out as a simplistic two-way tool that helps in early and continuous identification of risks, studying the factor, the controlling event and its effect on the outcome *(Performance)*. RISKIT also defines probability and impact in a non-numerical approach, ranking each in a two-way matrix that helps decision makers identify the highest ranking risk in terms of both probability and impact at the same time.

As for the Constraint Management, it has been noticed that time, budget and resources are the most common constraints in many software projects. This is the reason why this study is proposed as an enhanced model to soften the impact of those restrictions on the performance. One basic approach for achieving this is project activities scheduling, its main function is to clarify the project from different perspectives. Not only will it help to meet the constraints of the project proposed but it will also give insights on expected risks. However setting such a schedule may raise another issue, which is Resource-Constrained Project Scheduling *(RCPSP)*.

RCPSP aims to minimize the total duration of the project by applying a non-pre-emptive scheduling on the different project activities. In this regard, a genetic algorithm with two point crossover to handle this issue has been proposed **[11]**.

The paper is organized as follows; Open section A highlights the introduction. Section 2 examines related works. Section 2 motivates the TOC concept and clarifies this concept through two examples. Section 3 describes the risk management concept and defines some of the factors that affect the project performance. Section 4 presents an enhanced model that addresses performance in a chain of risk and constraint factors. Section 5 applies a performance concept on Closed and Open Source Software Development in a comparative way. Section 6 presents one of the methodologies on how to handle risk management called RISKIT and Section 7 presents an example about Resource-Constrained Project Scheduling Problem *(RCPSP)* and the proposed solution used to overcome this problem. Finally, Section 8 represents the final discussion and Section 9 concludes our work.

1. Related Work

Studies of Constraint Management and Risk Management are rich in the literature.

Starting with Constraints; Goldrattand et.al **[1]** suggested that the following five focus steps are the basis for the business improvement process proposed in TOC:

- Identify the constraint(s).
- Decide how to exploit the constraint(s).
- Subordinate all else to the decision in step 2.
- Elevate the constraints(s).
- If in any previous step a constraint is broken or eliminated go to step 1.

Zadry and Yusof **[2]** defined TOC as a blend of philosophy, principles and tools conceived to ensure optimal performance of any organization by enabling the members of that organization to identify control and eliminate any problem that prevents that organization from operating at peak performance.

Mablin and Balderstone's **[3]** synopsis of TOC conveyed the important message that TOC is a powerful management theory that encourages organizational leaders to approach problems they face from a system perspective using a systems thinking process in an environment designed to support the focusing/iterative process of ongoing improvement. Such an approach allows identifying breakthrough and sustainable solutions to both simple and complex problems.

The results of a meta-analysis of over 80 successful TOC applications, reported by Mabin and Balderstone in their literature-based research, showed that a sustained application of TOC resulted in measurable improvement in the operational and financial performance of an organization **[3]**. Operational performance measures included lead time, cycle time and due date performance. The financial performance measures included inventory throughput and profitability.

As for Risk Methodologies, in **[16]** each method has its own characteristics. The selected framework for risk management was selected based on the fact that there are several use cases conducted in distended time frames, with the possibility of using its application not only on software engineering, but also globally applied in various fields.

The RISKIT method was published by Jyrki Kontio from the University of Maryland **[21]**. RISKIT is a framework that covers all aspects of Risk Management; Identification, Assessment, Control and Monitoring. RISKIT was designed for Software Engineering Project Management, its main key values comprise the following: it initially includes stakeholders in the Risk Management Process, secondly - it is a pure graphical representation that was found to be easy to apply, and finally it tolerates imprecise data of both loss and probability. RISKIT can be applied to projects that are not purely from the computer science field.

2. Theory of Constraints

Over the past twenty years, Theory of Constraints *(TOC)* has been developed by Goldratt. TOC applies a methodology that is derived from hard sciences *(cause-and-effect)* to understand and enhance the human-based systems such as the lives of individuals and organizations. Also, it helps people to think creatively, solve problems and implements the best solution found **[4]**.

Conventionally, any project required to be achieved and carried out within certain constraints. These constraints can be classified as time, cost, and scope, identified as the "Project Management Triple Constraints" where each side of the triangle is considered as a constraint.

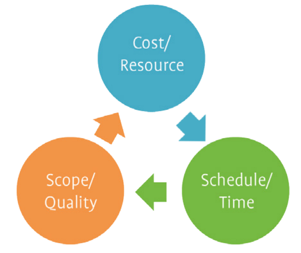

FIGURE 1. *The Triple Constraints [5]*

The triangle in **Figure 1**: The Triple Constraints **[5]** demonstrates the relationship between the main three components of the project. Time or Schedule is when the project has a deadline to deliver a product with certain results. For example, a project must be ended by January 30th. In this case, it is not clear that the project will be ended at this due date, but someone is expected to finalize it working on the project and produce the final product on this deadline. Costs are the available amount of money or it could be the available resources to complete the project, which include people, equipment, information, and so on. For instance, a budget of $150,000 assigned to a certain project or having three employees working for two months. Finally, scope includes the sort of tasks needed to achieve the goals. Also, it is the way to ensure that the project contains only the total work needed to be done in order to perform the project successfully **[5]**.

Successful project managers will meet those triple constraints by balancing each one of them. The relationship between these factors is that one of these factors cannot be changed without affecting the others; at least one of them will probably be affected. For instance if the time has been reduced, the required funds will be increased to add more resources in order to finish the same amount of work in less time. A further enhancement of the constraints triangle model is that the performance or the quality has been separated from the scope and the performance has been turned into a fourth constraint **[5]**. According to Goldratt, the organization performance is dictated by the constraints which prevent the organization from maximizing performance and achieving its goals **[1]**. To improve performance, project managers should continuously execute the planning and reviewing of the project since no project works exactly the same as planned in the initial project plan.

By thinking ahead and focusing on the constraints, project management can contribute to project success when it is applied in any organization that supports a team and continuous improvement environment. When applying the philosophy, concepts, principles, and tools embedded in the TOC framework, organizational leaders are able to provide their members with the necessary tools required to identify, manage, and break the most restrictive limiting factor that prevents them from contributing to the success of projects. In the project planning phase, project managers take into consideration the constraints associated with the resources, time, and cost of the project, that will definitely be able to increase the project outcomes with the lowest possible cost.

For example, according to Noreen, Smith, and Mackey **[6]**, a good practice of using TOC would be when the top management of Baxter medical products plant is acting wisely enough to adapt the constraints associated with the production process in the plant, and the designed system is flexible enough to accommodate these adaptations. For example, when materials are a constraint, they look for other vendors, when machines are a constraint, they increase the load on machines as appropriate and if the machines have reached their maximum load, new machines are ordered. In this factory, the constraint was the plastic extruding machines, a new extruding machine was ordered because the manager wanted to increase the plant capacity. However, before the new machine arrived, the manager realized that the constraint would move to the blenders once the new extruding capacity was added. As a result a new blender was planned and ordered. From this example it can be concluded that by thinking ahead and focusing on the constraints, the plate manager is able to increase the plant's real capacity at the lowest possible cost.

On the other hand, a bad practice of early identification of constraint would be: when the top management of Denver airport **[7]** did not act wisely enough to accommodate the delays of finishing the software that controls the automatic baggage system. Instead of waiting until the whole system was completed, they could partially open the system to perform unit testing on individual parts of the system before testing the overall system. For example, they could partially operate the system for testing purposes to be able to accurately synchronize the timing between the conveyor belts and the moving tele-carts. As a result the misleading, misrouting, and jam problems that happened after operating the system could have been prevented. Thus, they could have avoided the delay which cost Denver $1.1 million a day, including interest and operating costs **[7]**.

3. Risk Management

3.1. What is Risk Management?

Risk management techniques should be considered while developing software to avoid project delay or failure. Generally, any risk management methodology involves the identification, assessment, prioritization, control and monitor of all expected risks **[16]** as shown in **Figure 2**: Risk management process.

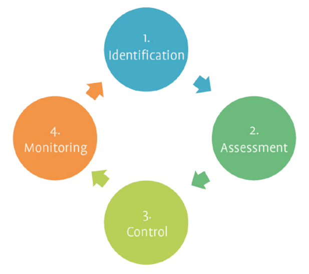

FIGURE 2. **Risk management process**

3.2. Risk identification

Risks must be clearly identified in order to manage them. This phase mainly depends on the organization itself and how much it encourages people to highlight issues that may cause threats. A list of these most frequently occurring risk factors is: user *(U)*, requirement *(R)*, project complexity *(Comp)*, planning and control *(P&C)*, team *(T)*, and organizational environment *(Org)* and Budget *(B)* **[16]**, **[17]**, **[18]**.

User dimension (U)

This dimension focuses on any user involvement which affects the development process, such as:

- Ability to clearly communicate with the developers
- Raising conflicts in users group
- Users resistance for change
- Non positive attitudes headed for the project
- Users Lack of cooperation
- More stress on the software from users than expected (especially in web services)

Requirement dimension (R)

This dimension of risk arises from factors related to different kinds of requirements, such as:

- Requirements are not properly stated
- No agreement on goals
- Frequently changing requirements
- System requirements are not sufficiently identified
- Vague system requirements
- Incorrect system requirements
- No risks or contingency plans are identified
- No proper control is done
- Poor handling of project changes

Project Complexity (c)

Project complexity risk dimension can be characterized by the following factors related to technology:

- Project involves the use of new technology
- High level of technical complexity
- Immature technology
- Technology does not meet expectations
- Project involves the use of technology that has not been used in prior projects
- Less reuse than expected
- Backup not taken and actual document/data loss
- Too many development errors

Planning and Control (P & C)

Planning and control is a risk dimension that affects many aspects and results in many risk factors, such as:

- Low estimation of time and cost Delivery
- Deadline tightened or management change circumstances
- Lack of effective project management skills and methodologies
- Project progress not monitored closely enough
- Inadequate estimation of required resources leading to shortfalls in the required tasks
- Inadequate estimation of required

materials, lack of control on suppliers

- Changing scope / objectives
- Project milestone not clearly defined
- Ineffective communications

Team (T)

Some factors that can cause the risk from the team dimension:

- Team members lack of experience
- Team member's development does not undergo adequate training
- Unspecialized team members
- Lack of training on tool or staff inexperience

Organizational Environment (Org)

This dimension shows how organization environment affects the project and how it can become risks, for example:

- Management change during software development and high turnover
- Having a system that is not created to support the development, policies that effect the process negatively
- Environment that is not rigid
- Restructuring the organization while software development is taking place

Budget Management (B)

Here is a list of some related budget risks:

- Inadequate allocation for resources lead to budget waste
- Unrealistic time and cost estimates
- Funding may be lost because of poor management and communication
- Funding may be lost due to the funding agency's unawareness of the progress

3.3. Risk Assessment & Control

In this phase, previously identified risks are reviewed, prioritized and analyzed in terms of cost, time, and its effect on product quality. This kind of analysis will provide information to decision makers to assist them in their decision making **[15]**.

Also, in this phase a risk management plan is provided, which includes actions to be taken, priority and future consequences for each action.

3.4. Risk Monitoring

Both risks and actions taken to deal with risk are tracked. Gradually actions are changed as a response to risk reduction. Risk management plans are changed to handle new mitigated actions and different triggered events. Finally the risk management process is improved and the plans produced can be used as a reference for future projects **[15]**.

4. Project performance

Providing details about the performance of a project is a common task assigned to project managers. Such information is crucial during the development to verify that a project will be able to achieve its objectives within the defined constraints, or for a company considering investing in a project. There are numerous studies on predictors for evaluating project performance, based on two criteria: effectiveness and efficiency **[8]**.

4.1. Relation between Risk, Performance and Constraint

The proposed model is mainly based on the exploratory model of Wallace et al. **[21]** which illustrates the relation between the risk factors, project performance and project constraints in the scope of software development management.

In this model, performance is indicated by product and process performance, another component of this model is the constraint factors which are, resources, scope, cost and time as shown in **Figure 3**: Diagram of the Proposed Model. For the social subsystem and the technical subsystem risks, they have direct impact on project management risk. According to the proven hypotheses given by **[21]** the impact of the social system on both the technical subsystem and the project management is significantly positive. Moving to project management, it has a negative effect on product performance and process performance. Moreover, process performance has a negative impact on product performance.

Whenever new software is developed, its performance effectiveness is measured by how it meets its objective and works around its limitations. As discussed previously in this paper, the objective is determined by the scope constraints, while the limitations are presented in the available recourses, allocated budget and pre-defined time period. The proposed enhancement of the model addressed the measurement of software projects performance based on meeting constraints.

The product performance is directly proportional to meeting the scope, while the process performance always takes cost, time and resources into consideration. Each constraint factor has to address a certain risk factor to control performance in the best way to ensure great performance.

Referring to **Figure 3**: Diagram of the Proposed Model, it can be seen that resources, cost and time constraints introduce risks in the project management risks category. Meeting the scope increases the user's risk factors, such as the sub-category of social system as well as the technical subsystem risks category.

Since project management monitoring is a continuous process, whenever new constraints or new risk factors arise, the performance will be affected.

4.2. Efficiency and Effectiveness

The efficiency is a ratio of the global output created by the project, on the global input used from the beginning of the project. Therefore, it is an economic indicator whose evolution can be followed during the development.

The effectiveness is the capability of achieving goals. This criteria is focused on the results, it is the ability of the project to match the objectives respecting the constraints **[8]**. It is evaluated by comparing the time used, the costs, and the quality obtained, with the previsions and the expected results.

5. Performance in light of Open Source and Closed Source Software Development Management

Closed source software is developed under copyright licenses. The users have no way to access the sources, and run compiled binaries.

Open Source Software *(OSS)* are released under special licenses that allow the public to examine the source code in addition of using it. The open-source projects challenge the closed-projects by frequent releases, collaboration, co-development and finally by following the open standards. On the other hand, the closed source follows the traditional paradigm, where they treat software development as specialized process, managed by local and highly qualified developers.

The previous methods to measure efficiency and effectiveness cannot be applied in the case of Open Source Software projects for the following reasons:

- The project is not made to generate revenue by selling licenses, nor financed like closed-source software: the efficiency ratio cannot be calculated in the same manner.
- The project does not match predefined constraints nor objectives of a particular customer. The effectiveness cannot be evaluated like with closed-source software, since the context is different.

It is best to investigate a new model that helps ensuring efficiency and effectiveness in the case of open source development, but before that more about the open source development itself.

5.1. Open-source development model

There are numbers of highly successful open-source software *(OSS)* in the public context, such as the Mozilla web browser, as well as among professional circles, such as the Apache web server. Making a software open-source allows the users to contribute or to provide feedback on the project during its development. It allows a large number of potential users to build, test and debug early releases on multiple platforms, instead of limiting those processes to internal and subsequent phases.

There are hundreds of open-source licenses; globally they are classified in two categories **[9]**:

- Copy left licenses: the sources can be accessed, used, modified and shared. However, the used code has to comply with the same license. It is, for example, not legal to include those sources in closed-source software.
- Non-restrictive licenses: the sources can be used and integrated in proprietary software.

At the beginning of an OSS project, the standardized requirements are limited and incomplete. The boundaries are not really defined, as the project will evolve according to the new users' needs **[10]**. The evolution of the development will depend on the user's interests and volunteer developers. So, how does one then measure OSS performance?

5.2. Performance evaluation for OSS projects

There are many metrics that can be used to measure the project performance. James W has studied these metrics and consolidated them in the following **[8]**:

- Overall growth of project functions over time
- Functions added over time
- Overall projects' complexity
- Average complexity of all the functions
- Average complexity of the added functions
- Functions modified overtime
- Functions modified as a percentage
- Correlation between functions added and modified

All these metrics are used to measure the projects' performance, but this part of the paper will focus on defect-removal and functionality enhancements as they are core of OSS development.

Several studies about the DCT *(Dynamic Capabilities Theory)* define the Dynamic as the capability of a company to change and adapt to the market. Capabilities also depend on market changes, where the company can integrate, build, and reconfigure its resources in response to market changes. Depending on the DCT, a hypothesis can be conducted to study the project capabilities with regards of defect-removal and functionality-enhancements in OSS. These two processes should not be confused with maintenance, as the project keeps changing to meet new requirements.

The development of OSS projects is incremental and has dynamic characteristics. This paradigm follows the evolutionary software development model. This model aims at removing defects and functional-enhancements.

With the nature of OSS where there are blurred boundaries, incomplete requirements, and unpredictable changes, it can be considered a high velocity environment. In such projects the developers are volunteers, and the projects can be highly influenced by the volunteers' interests. Attracting these volunteers has major impact on the project effectiveness. Some hypotheses can be formulated as follows **[8]**:

- H1. The stronger OSS projects pro-activeness in defect-removal, the more performance it gets.
- H2. The stronger OSS projects efficiency in defect-removal, the more performance it gets.
- H3. The stronger OSS projects pro-activeness in functionality-enhancement, the more

Factors	Relevance	
	Open-source	Closed-source
Constraints		
Costs	Low	High
Meeting scope	Low	High
Time	Low	High
Utilized resource	Medium	High
Risks		
Budget	Low	High
Organizational environment	High	Medium
Planning and control	Low	High
User	High	Low
Social	High	Low
Technical	Medium	High
Team	Low	High
Project complexity	Medium	High

TABLE 1 . *Constraints and risk factors in both open and closed source projects*

performance it gets.

- H4. The stronger OSS projects efficiency in functionality-enhancement, the more performance it gets.

These hypotheses are presented in **Figure 4**: Relation between Open-Source Parameters and Performance Indicators to track the relations.

5.3. Analysis and Results of the Dynamic Capabilities Theory

The results found from the research conducted **[8]** confirmed the proposed hypothesis. Most of them have matched the expectations and they are as follows:

- Impact of defect-removal on OSS project performance: the results showed a positive effect and supported H1a, H1b, and H2b; however, the analysis did not support H2a.
- Impact of functionality-enhancement on OSS project performance: the analysis supported H3a, H3b, and H4b; however, the results did not support H4a. The results are consistent with the expectations.
- Model fit, control variables: the results show that neither license restrictiveness, nor development status have significant influence.

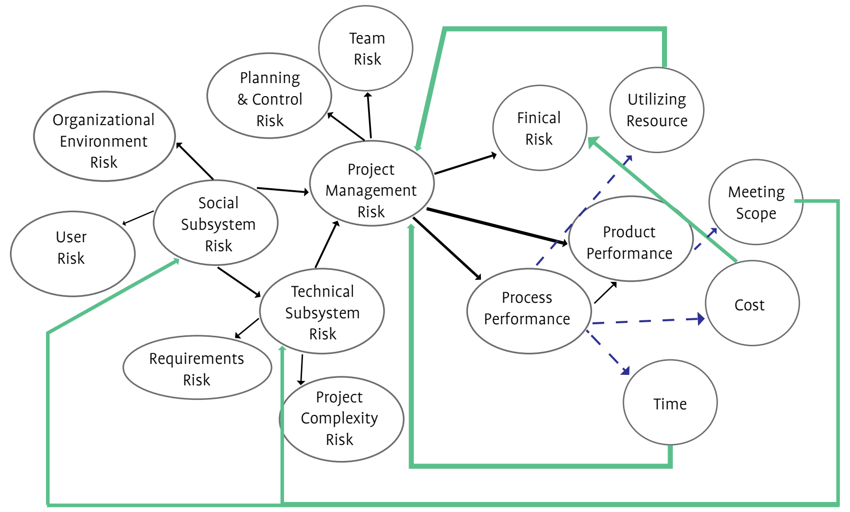

FIGURE 3. *Diagram of the Proposed Model*

5.4. Open-Source and Closed-Source Software Relevant Factors

Risks and constraints do impact the performances. However, due to the inherent differences between open and closed-source project, not all factors are relevant to both strategies. The differences between open and closed source are shown in **Table 1**: Constraints and risk factors in both open and closed source projects.

A survey was made to construct this table, and included nine participants from IT sectors from different companies, namely: Qatar Petroleum, Ooredoo Telecom, Navlink, Wipro Technologies, and Perpetuuiti Technosoft. The first thing to note is that the impact factors differ between Open-source and Closed-source performance. To understand the cause of these differences, a hypothesis is made for each factor in risks and constraints.

Constraints

Costs: The reason for the high impact on the closed source is very clear. Every resource used in the closed-source has a cost, and if the project does not have sufficient budget to cover the required resources, then it will defiantly impact the performance of the project. While in the open-source, since it depends on reusability and volunteers, the cost is considered very low, so the impact would be low.

Meeting Scope: In structured companies, once the scope of the project is defined, it becomes very rigid, and difficult to alter. Failing to meet the scope means failure of the project. On the other hand, the members of the Open-source community define the scope themselves, and there are no penalties to alter these scopes and affect the performance.

Time: The Closed-Source projects have definite timelines that have to be met. Depending on the time setup, the project manager will allocate resources and roles to the project. If the time is short, the project might not have the sufficient resources to meet the time, and this affects the performance. The OSS have flexible timelines, so the project can run at the desired pace.

Utilized Resources: In closed-source projects, every resource is designated and utilized for specific tasks depending on the resource profile. A resource might have the capacity to be used in different tasks, and if it is not enlisted in its profile, it will be a waste. The size of the input in the project will be higher which will impact the performance, since the same output can be achieved with fewer resources. However, the open-source projects have different scenarios, where any developer can work in any task, but the lack of communication might lead to duplicate efforts.

Risks

Budget: As discussed in the costs factor, the OSS budget is very low, so there are no cost restrictions that can affect the performance, while in the closed source, it is the opposite.

Planning and Control: Planning and control are fundamental in closed-source and they have more emphasis in closed-source than the open-source. As discussed before the open-source performance depends on feature enhancements, and fixing bugs frequencies. These actions are not planned and controlled. They can be done by anyone at any time. While in the closed-source, they have to be planned and controlled, which add more overhead to the process and leads to higher impact on the performance.

User and Social: It is generally addressed by using adapted development platforms in open source software projects, which integrates user interaction as well as promotion to the public over the internet. On the contrary, closed-source software projects do not involve users and require less social interaction.

Technical: In open source, technical issues can be solved by new more experienced contributors. This reduces the risk compared to closed source, where the team is fixed: expertise is shared with everyone, technicalities are more likely to be overcome, and problems are solved with more responsibility.

Team: There are no designated teams in open-source projects, but rather collaborative contributions across the globe. As a result, team efficiency does not have the same impact on the performance of open-source, as it does on closed-source.

Project Complexity: Regardless of the type of project, complexity will always have an impact on project performance, but the difference is how to deal with it. The open-source projects start with low complexity, but with more functionality enhancements added, hence more complex. It can be observed that open-source projects do not have to deal with complexity all at once, but incrementally. As a result, the project can start as soon as possible, without overly considering the complexity. The closed-source has to resolve complexity from the start, since it is part of the project's scope, and the outcome is a great impact on performance, since they need greater efforts and resources to resolve complexity.

An interesting point was raised by a project manager at Qatar Petroleum in one of the surveys. He simplified the relation between risks, constrains, and performance and stated, "A project manager usually avoids risks by creating more constraints, this may affect negatively on the performance. It is the project manager's responsibility to balance out these three aspects."

Another statement of specialist at Perpetuuiti Technosoft was "The different values of effects between these factors might change depending on the seriousness of the open-source project and the environment. Some of the open-source projects are serious, and have clear scope, so the effects will be mostly higher". He also added that the lack of ownership is a major issue in the open-source projects, which reduces the commitment by the volunteers in the project.

Risk, Constraint and Performance have been explored individually, and then their relations and effects on each other. The next section provides one selected Risk Management Methodology *(RISKIT)* and one Constraint Problem *(Resource Scheduling)* Solving approach *(RCPSP)*, from the various options for enhancing performance.

6. RISKIT: Another Approach for Risk Management

One of the proposed methodologies **[21]** for risk management is the RISKIT method; its main key values are: First, include stakeholders in the risk Management Process, Second it is a pure graphical representation that was found to be easy to apply, it tolerates imprecise data of both loss - or impact- and probability, and finally it is very general and can be used not only by developers.

The main steps of the RISKIT framework are the following; Risk identification, Risk analysis, Risk control planning, Risk control and Risk monitoring. They are integrated as shown in **Figure 5** hereafter. Each step leads to the next one consecutively, with the ability to return to any step.

6.1. RISKIT Risk Identification

To clearly identify the risk factors, the following questions are asked:

- What is the risk factor?
- What event is it related to?
- What is the expected outcome if the risk event occurs?
- What kind of reaction is expected when the risk factor takes place?
- What are the series of effects when the project plan deviates?
- What utility might be lost, especially for stakeholders?

Each question will be graphically answered in a box labeled with a certain color based on the content of the triangle, for example, the yellow for the risk factor name. The graph is called a risk scenario, an example of a software risk is shown in **Figure 6** hereafter.

6.2. RISKIT Risk Analysis

One of the important features that makes RISKIT unique is defining all losses, this is called risk analysis, which can happen because of any risk factor during the software development process. Losses are identified in the identification phase through a matrix that covers both the loss in terms of utility and the stakeholders. Utility loss is the loss of a material, resource or money to deal with the effect of the risk. Stakeholder loss is the loss of the client or the funding agent due to specific risks.

The matrix is built by sequentially ranking the losses two times, one for utility loss and the other for probability. Joining the two tables results in

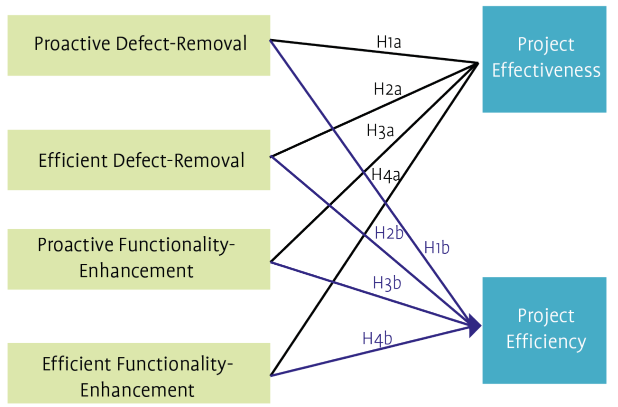

FIGURE 4. *Relation between Open-Source Parameters and Performance Indicators*

visually glancing what each stakeholder has for the development of the software, this Matrix/ Table is called RISKIT Pareto, **Figure 7** shows an example of the RISKIT Pareto.

6.3. RISKIT Control

Risk Control is the way to actually deal with-or avoid-Risks, there are 5 main ways to control a possible risk, as seen in **Figure 8**.

First is Not to Take any Action, when not taking actions requires either waiting and see what will actually happen, or hiring experts to gain more information before making a decision. This is the most used risk scenario **[21]** that recommends GQM system which provides trees and leaves methodology instead of traditional matrices.

The second one is to have a Contingency Plan, which is similar to plan B, Plan B would be ready for recovery only in the case the risk did take place, then new actions will be taken based on that plan. Contingency plans are seen as recovery insurance, they also provide possible reactions and effects to a risk scenario.

The third approach to control stress is to Reduce Loss, loss reduction takes place in 5 methods; The first method is about recovery, which includes four main sub options; the first is to reserve extra resource if the risk occurs, human resources, IT resources or financial resources. The second sub option of recovery is over staffing, which requires more budget but assures that each member of the new staff has enough experience to take the project forward. The third sub option is preparatory work, it is almost like the contingency plan, but it involves taking actions instead of planning, and if the risk does not take place, then the work applied would be wasted. The last methodology of the recovery options is over engineering, which is about providing extra design and coding alternatives if the risk did take place with the product itself.

After recovery, the Second method of Loss Reduction is to Create Compensating Benefits, in other words, it is to provide a benefit to the customer instead of a utility loss, like providing free training, or an IT representative.

Third method is Risk Transfer, which also has sub options but only three of them, the first one regards risk sharing, it happens by taking approvals that are based on the conditions from Project Owners, Stakeholders, Contractor and Customers. Sharing risks regards proper communication with everyone involved, moving forward while addressing probable rising issues *(risks)* and also their impact. The second sub option, which is Management approval, is similar to the previous one *(Sharing Risk)*, it is having risk ownership entirely by management, this may happen when a new product is being developed, for example a new version of an operating system, and as long as it has the approval of management the development process can go on without having to justify the need for a new version, or specify tight deadlines. The last sub-option is insurance via an insurance company, though one cannot really insure a software, so this option would be a rare one. Risk Transfer can be used as a way to avoid responsibility, it has to be done in special circumstances and pre-agreed conditions, in this case, handling risk is not shared, but given to someone responsible, this option is usually used alongside other options.

The fourth and fifth methods in Loss Reduction are to find other Alternatives and to Change Goals, and are also shared with Risk Avoidance.

The fourth way to control a Risk is to actually avoid it! Choosing Alternatives include choosing new approaches, resources, tools, methods and technologies, and once those are chosen the risk itself will then be changed as other alternatives are being used. As for Changing Goals, in the context of this paper, one can also say changing constraints, since what will be lost depending on what the goal is, if the goal changes then the losses will be different, reduced or even erased, especially if those were unrealistic goals. This controlling procedure is the most effective, nevertheless it needs much negotiation with stakeholders, and managers are not always supportive of it.

The last way to control risk is to Reduce Event Probability, this takes place by Influencing Risk Factors or by Observing the Risk Monitoring Metrics, if one knows that a developer is not very experienced, reducing the probability and even influencing it can be achieved by providing training to the developer.

Those controlling options explained by **[21]** were presented as a guideline of options and not as a rigid solution; they also mention that it strongly depends on the judgment of those in charge of the project, hence the better the judgment, the best performance.

RISKIT management was applied and studied in many other contexts such as in NASA, Nokia, DaimlerChrysler and Tenovis. In those contexts the framework application took between 5% - 20% of the overall project management workload **[23]**, an introduction session was required, and was given as a hands-on workshop to the project's team members chosen to lead the project's risk aspects. Though easily understood, ownership of risk Management has proven to be crucial in keeping track of risks and assertively questioning control decisions.

RISKIT can be applied to all types of projects, open source or closed source software. The only difference is that the risk types can vary depending on the software development itself. One may not have control of the contributing team, but as long as a person with ownership is managing the development of the software, with their experience they can adequately assess any possible internal or external risk.

7. Resource-Constrained Project Scheduling Problem (RCPSP)

In the field of operations research, resource-constrained project scheduling is considered an important and popular problem, also attracting many researchers. Its main objective is to minimize the total duration of the project by applying a non-pre-emptive scheduling on the different activities depending on precedence constraints to conclude and to begin, and also on renewable resource constraints. However, it is found that this scheduling problem only works properly for closed source projects with small number of activities because it is extremely NP-hard. To solve this problem and to make this scheduling applicable for both closed and open source projects we proposed a genetic algorithm with two-point crossover **[11]**.

7.1. RCPSP Description

The main idea of Resource-constrained project-scheduling problem *(RCPSP)* is executing a number of activities by finding its starting and finishing times, and therefore reducing the project makespan. On the other hand, constraints sometimes limit activities. For example, precedence relations force the activities to wait until other ones are completed. Also, a set of previously defined resources used for processing activities is needed, where each resource has a fixed capacity for a specified time.

In general what we need for RCPSP:

- A set E = {(i, j) ∈ A^2 / i precedes j} which represents the precedence relations between activities.
- A set R = {1,..., m} of resources.

Therefore, each activity depends of its precedence relationship and its use of resources **[11]**.

7.2. The Genetic Algorithm (GA)

The genetic algorithm selected in this paper to solve the RCPSP is considered as metaheuristic. When there is no exact method to solve a problem, metaheuristic are used to solve it, giving an approximate solution in a reasonable amount of time. They are based on two different approaches:

- Single solution approach
- Approach depending on population: in this type more efficient individuals are produced or improved by the interactions of a population of individuals.

The genetic algorithm proposed in this paper is considered as a metaheuristic-based population **[11]**.

The best way to understand the way this genetic algorithm works is by following **Figure 9**: General structure of the GA **[11]**, which shows the GA general structure. First, a generation of population is created by a defined number of individuals to find the initial solution; the generated solution may or may not be the best solution for the problem. To produce the next generation, a number of techniques such as evaluation, selection, crossing, mutation, replacement are applied to the individuals in the population. For each iteration, and once the condition is accomplished the best solution is found and the whole work is done **[11]**.

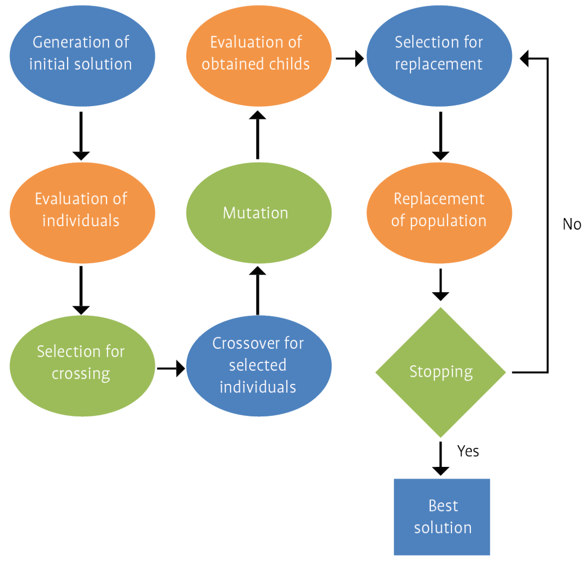
Generation of initial solution
Evaluation of obtained childs
Selection for replacement
Evaluation of individuals
Mutation
Replacement of population
No
Selection for crossing
Crossover for selected individuals
Stopping
Yes
Best solution

FIGURE 9. *General structure of the GA [11]*

7.3. Reproduction of The GA with Two-Point Crossover (AG^{2P})

Genetic Algorithm is a technique used to teach systems applied in Genetic algorithm and can support a self-learning machine. It is a search algorithm based on the natural selection mechanism and on genetics. It states an analogy between a set of problems to be solved and the set of individuals in a natural population.

The mechanism it uses consists in coding the solution information *(population)* into chromosomes, which are string of numbers *(arrays)*. Then apply a crossover to these chromosomes *(array elements)* in order to create new chromosomes *(new arrays)*, these new chromosomes are called children. Secondly, evaluate these children by using an evaluation function **[11]**.

7.4. Description of The Algorithm

Before explaining the way it is used to solve the RCPSP problem, we first introduce how GA generally works. It generally consists of six main steps *(taken from* **[12], [13]***)*:

- 1. Start step: a random population of n chromosomes is generated. These generated chromosomes are called parents, which present suitable solutions for the problem.
- 2. Fitness step: evaluate fitness value for each chromosome x in the population, which means that a fitness function is generated for each parent in population.
- 3. New population step: a new population is created by operating and repeating the following steps until the new population is complete:
  - a) Selection: select two parent chromosomes randomly from a population according to their fitness, where the better the fitness, the greater the chance to be selected.
  - b) Crossover: with a crossover probability, cross over the parents to form new children. If no crossover was performed, then the children are an exact copy of their parents.
  - c) Mutation: with a mutation probability mutate new children at each position in chromosome, this step is done by randomly selecting any gene from the new created chromosome and change its value from 0 to 1 and vice versa.
- 4. Replace step: place new children in a new population in order to use the new generated population for a further algorithm run.
- 5. Test step: evaluate fitness value for the new created solutions (chromosomes), if they are satisfied, then stop, and return the best solution in the current population.
- 6. Loop step: if they are not satisfied, back to fitness step to evaluate the fitness for each new generated population and repeat steps 3, 4, and 5.

Now moving to our RCPSP problem, the following modification is done *(note that we represent the project activity or task by a chromosome)* **[11]**:

In the starting step, we select the parents by using permutation probability techniques, as we added two additional genes to the end of each selected parent. This intends to determine the method that is used to generate a scheduling starting from a given sequence of activities. The first gene is called s/p, which determines if the activity *(chromosome)* is necessary to use the series or parallel algorithm. The second gene is called f/b, it determines the position of the project task.

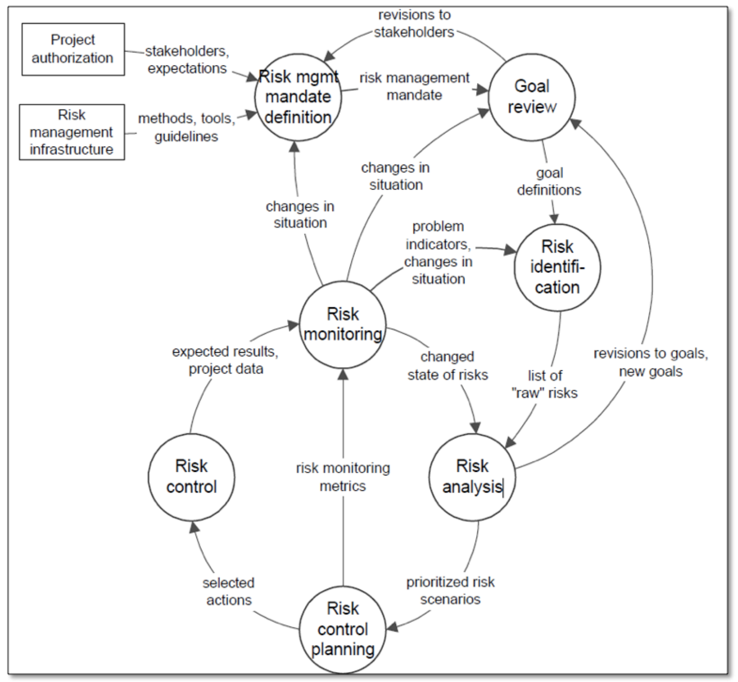

FIGURE 5. *Overview of RISKIT Process* **[21]**

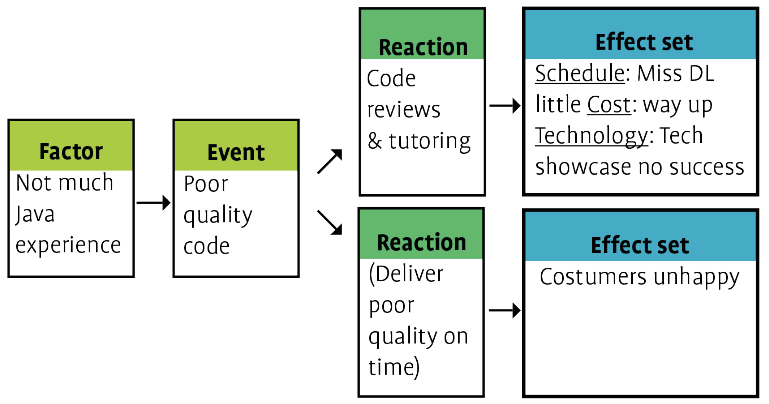

FIGURE 6. *Example of the RISKIT Analysis Graph*

Stakeholder A		Risk scenario probability				
		Rank 1	Rank 2	Rank 3	Rank 4	Rank 5
Risk scenario utility loss	Rank 1	Risk A				Risk D
	Rank 2			Risk C		
	Rank 3	Risk B	Risk E			Risk F
	Rank 4					
	Rank 5					

FIGURE 7. *RISKIT Pareto Table*

To generate the initial population, we used a probabilistic progressive construction. Since in our case we need to respect the constraints of precedence, we select the next gene depending on them proportionally with the Latest Finish Time LFT priority rule: min{t_i^{LF}}, where tiLF is the latest finish time of activity i from the eligible set. For both s/p and f/p, they have been randomly initialized.

Now in order to generate N chromosomes, we select two parents to do the crossover; apply the representation by permutation to produce children that inherit genes s/p and f/b of the first parent; and repeat this process N/2 times, where the crossover is done according to a probability P^{Crois}. The parents that are not crossed go directly into the child population.

For the mutating step, each activity of the sequence chosen according to a probability P^{Mut} given randomly; selects a new position and the inserts it. In order to maintain compliance with precedence constraints, the new position should be between the last and the first predecessor of the successor activity. The last two genes, s/p and f/b, are reversed according to the same probability P^{Mut}.

The pseudo-code of this algorithm, GA2P, is as follow **[11]**:

- Step 1: Initialize t = 0, Generate Initial Population (POP). Select the solution representation technique.
- Step 2: If t > 1 updates population, otherwise go to step 3.
- Step 3: Crossover.
  - Step 3.1: Parent Selection for Crossover.
  - Step 3.2: Generate Child population (POP) such that Number of parent schedules = number of child schedules and Number of child schedules consists of equal number of sons and daughters.
- Step 4: Mutation.
  - Step 4.1: Set Mutation probability.
  - Step 4.2: Select Chromosome (Schedule).
  - Step 4.3: Perform swap mutation such that precedence is not violated.
  - Step 4.4: Generate mutated schedules.
- Step 5: Selection of next Generation Population.
- Step 6: Increment t.
- Step 7: Check if termination criteria is achieved.

If yes end loop otherwise go to step 2.

A final remark about this proposed algorithm used to solve the Resource-constrained project-scheduling problem *(RCPSP)* and reduce the total duration the project may take. It is concluded after a number of experiments and comparisons with other metaheuristic methods using benchmarks, that this algorithm produces better results as the number of the activities increases and in general this method performs well **[11]**.

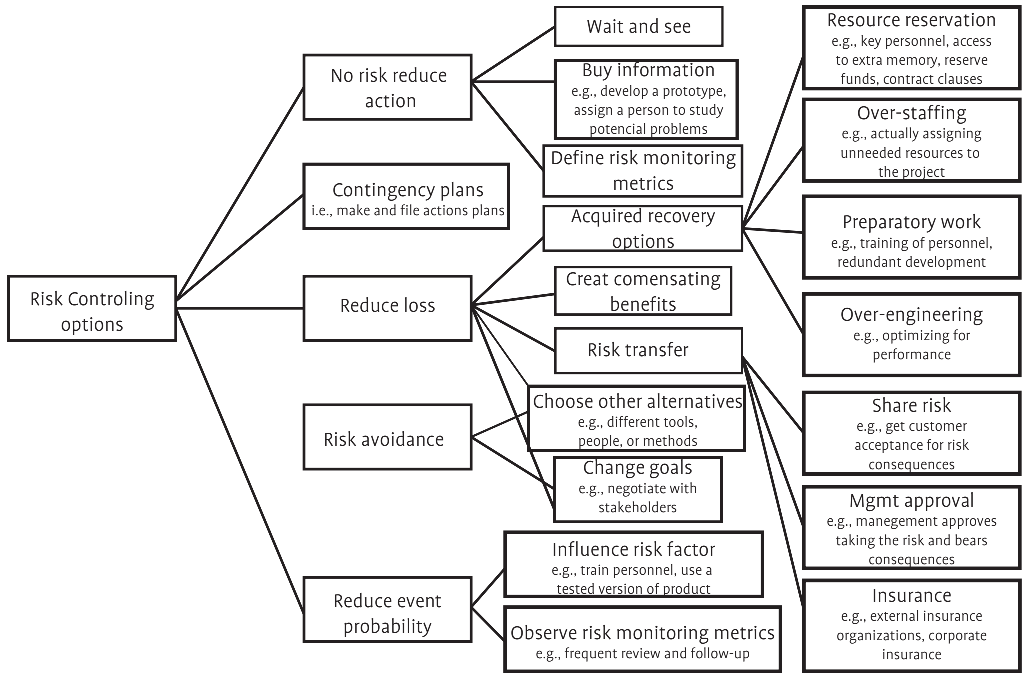
Risk Controling options
No risk reduce action
Contingency plans i.e., make and file actions plans
Reduce loss
Risk avoidance
Reduce event probability
Wait and see
Buy information e.g., develop a prototype, assign a person to study potencial problems
Define risk monitoring metrics
Acquired recovery options
Creat comensating benefits
Risk transfer
Choose other alternatives e.g., different tools, people, or methods
Change goals e.g., negotiate with stakeholders
Influence risk factor e.g., train personnel, use a tested version of product
Observe risk monitoring metrics e.g., frequent review and follow-up
Resource reservation e.g., key personnel, access to extra memory, reserve funds, contract clauses
Over-staffing e.g., actually assigning unneeded resources to the project
Preparatory work e.g., training of personnel, redundant development
Over-engineering e.g., optimizing for performance
Share risk e.g., get customer acceptance for risk consequences
Mgmt approval e.g., manegement approves taking the risk and bears consequences
Insurance e.g., external insurance organizations, corporate insurance

FIGURE 8. *Options for risk management decision making*

8. Discussion

The paper focuses on the fact that Theory of Constraints is a tool that can enhance project management. It tackles one constraint at a time, takes it into consideration and how it will affect the project process. The constraint factors include Time, Scope, Budget and Resources; play a massive role in the management decision. First, each factor affects the other. On the other hand the effect of Constrains is either constructive or de-constructive. If they are limiting decision makers from moving forward then they are de-constructive. If a manager tackles those constraints and works around them focusing on high quality delivery, constraints could then be effective tools.

Since constraints introduce risks, and risks lead to cause harm or loss and affect the ability to achieve objectives. Risk Management is the next field Managers will ensure that proper performance is taking place. Main risk factors: User, Team, Requirements, Complexity, Planning and control, Organizational Environment and Budget are the main aspects in which decision makers identify, Assess, Monitor and Control, to reduce the probability of risk occurrence, or provide a backup plan, action or even avoid the risk.

Performance is usually measured by the way it meets the scope and other predefined constrains, since meeting constrains is the goal of a successful software project. By defining each constrain a new risk has been introduced. A model has explained how those three factors affect each other.

Since the focus of this paper is project management in the field of software development, Open source and Closed Source software are compared as to hoe they are licensed and the fact that efficiency and effectiveness do not apply in the same way for both cases. Efficiency and effectiveness are the two main factor in which performance is measured for meeting constraints.

A survey was then conducted to study each constraint and risk factor's relevancy on performance. In both open source and closed source software development, the effect of each factor did vary, as the two types have different environments for development.

To take a step forward, two approaches were summarized as tools in Risk Management and in Constraint Management. These are good examples of what is being discussed and proposed in the literature.

9. Conclusion

In this paper three main factors of Project Management were explored, which are Constraints, Risk Management, and Performance. The relation between these three aspects is highlighted leading to a new model that has been studied in two extents: Open and Closed Source Software Projects development.

This paper is an attempt to provide information and perspectives that could help Software Development project managers to achieve the best possible result considering both Risks and Constrains, constantly monitoring them and eventually delivering effectively and with efficiency.

authors

Mariam Salim joined the international AIESEC organization in 2006, there she worked in the field of Talent Management; conducting recruitment, allocation and development for local members, in addition to managing events, workshops and conferences focusing on youth leadership and empowerment themes. In 2011 she organized the Youth Forum to launch the Third Annual Report conducted by the General Secretary of Development Planning, in partnership with Silatech and AIESEC Qatar. In 2012 she managed ictQATAR's national internship program for ICT Career Awareness. Mariam is currently pursuing her master's degree in the field of computing at Qatar University, while working on projects related to innovation and development.

Abdelaziz Bouras is ictQATAR Chair Professor at Qatar University, QU. He is currently the Chair of the IFIP WG5.1 on « Global Product development for the whole life-cycle". His current research interests focus on distributed systems for lifecycle engineering, including ontologies and lifecycle modeling for intelligent products. He teaches Software Project Management and Simulation in the Department of Computer Science and Engineering of QU. Also was professor at the University of Lyon (France) where he leads a research team of the LIESP laboratory. He has been conferred the HONORIS-CAUSA honorary Doctoral Degree in Science of the Chiang Mai University (Thailand) from Her Royal Highness Princess Maha Chakri Sirindhorn of Thailand. Prof. Bouras is Editor-In-Chief and founder of the International Journal of Product Lifecycle Management (IJPLM), Associate-Editor and co-founder of the International Journal of Product Development (IJPD), and Editorial Board Member of several International Journals related to knowledge management and supply chain management. He is also co-founder of the new International Federation of Information Processing IFIP WG5.1 Group on Product Lifecycle Management (PLM) for which he acts as Vice-Chair in charge of Europe and Africa; and is member of the IFIP WG 5.7 Group on Integration in Production Management. He is also the co- founder of the international WG-PLM and its annual international doctoral workshop on PLM.

Antoine Asseman, Nada Ashqar, Eman Rezk, Nasser Alkhuzaei, Heba Dawoud, Rana Rihan, Souad Mecheter, students of Computer Science Department, College of Engineering, Qatar University, Doha, Qatar. P.O Box 2713

references

[1] Goldratt, E. M., & Cox, J. *(1984)*. The goal: An ongoing improvement process *(2nd Ed.)*. Great Barrington, MA: North River Press.

[2] Zadry, H. R., & Yusof, S. M. *(2006)*. "Total quality management and theory of constraints implementation in Malaysian automotive suppliers: A survey result". Total Quality Management, 17, 999-1020.

[3] Mabin, V. J., & Balderstone, S. J. *(2003)*. "The performance of the theory of constraints methodology". International Journal of Operations & Production Management, 23, 568-595.

[4] Frenklah G. & Mann R. *(2011)*. "Marriage with TOC delivers improved product". TRIZ, Dublin, Ireland.

[5] Tomtsongas. *(2011)* "Scope, Time and Cost – Managing the Triple Constraint". In Programsuccess, May 2, 2011.

[6] Noreen, E., Smith, D., & Mackey.J. *(1995)*. "The Theory of Constraints and its Implications for Management Accounting". Montvale, NJ: The IMA Foundation for Applied Research.

[7] L. Bernstein. *(1996)*. "Software in the Large," AT&T Technical Journal, vol. 1, pp. 5–14.

[8] A. H. Ghapanchi, A. Aurum. *(2011)*. "The Impact of Project Capabilities on Project Performance: Case of Open Source Software Projects", International Journal of Project Management.

[9] S. A Baird. *(2008)*. "The Heterogeneous World of Proprietary and Open-Source Software", University of Hong Kong, unpublished.

[10] G. Ferrie. *(2010)*. "Software Development Processes: Research Comparing and Contrasting Open-Source Versus Closed-Source Software Projects", Athabasca University, unpublished, 2010.

[11] H. Ouerfelli, A. Dammak. *(2013)*. "The Genetic Algorithm with two point crossover to solve the Resource-Constrained Project Scheduling Problems". 5th International Conference Modeling, Simulation and Applied Optimization *(ICMSAO)*.

[12] Y. H Liao, and C. T Sun, An Educational Genetic Algorithms Learning Tool, *(2001)*,from http://www.ewh.ieee.org/soc/es/May2001/14/Begin.htm **[accessed on Nov 2013]**.

[13] Genetic Algorithm tutorial, from http://www.obitko.com/tutorials/genetic algorithms/ga-basic-description.php **[accessed on Nov 2013]**.

[14] H. I Mathkour, B. Shahzad. *(2011)*. "Software Risk Management and Avoidance Strategy", International Conference on Machine Learning and Computing, IPCSIT vol.3. Singapore.

[15] A. Chowdhury, Sh. Arefeen. *(2011)*. "Software Risk Management: Importance and Practices", IJCIT, ISSN 2078-5828, Vol.2, issue 1.

[16] T. Arnuphaptrairong. *(2011)*. "Top Ten Lists of Software Project Risks: Evidence from the Literature Survey", proceeding of the international multi conference of engineers and computer scientists. Vol. I, Hong Kong.

[17] D. Tesch, T. Kloppengburg. *(2007)*. "In project risk factors, the project management professional's perspective", Journal of computer information systems.

[18] Al-Mudimigh, ZahidUllah. *(2010)*. "Risk Identification and Preemptive Scheduling In Software Development Life Cycle", Global Journal of Computer Science and Technology Vol. 10 Issue 2.

[19] Wallace, L., Keil, M., Arun, R., *(2004)*. How software project risk affects project performance: an investigation of the dimensions of risk and an exploratory model. Decision Sciences 35 *(2)*, 289–321

[20] Kontio, Jyrki. *(1997)*. "The riskit method for software risk management, version 1.00."Computer Science Technical Reports. University of Maryland, College Park, MD, USA.

[21] Kontio, Jyrki, G. Getto, D. Landes. *(1998)*. "Experiences in improving risk management processes using the concepts of the Riskit method." ACM SIGSOFT Software Engineering Notes 23.6: 163-1.