World Applied Sciences Journal 28 (10): 1366-1374, 2013
ISSN 1818-4952

DOI: 10.5829/idosi.wasj.2013.28.10.1679

Assessing Issues of Change Impact Analysis Process for a Software Projects

[1]Hassan Osman Ali, [2]Mohd Zaidi Abd Rozan, [3]Adamu Abubakar, [4]Akram M. Zeki,
[4]Abdullahi Mohamud Sharif, [5]Mueen Uddin and [6]Jamshed Memon

[1,2,4]Department of Information System, Faculty of Computing, UniversitiTeknologi Malaysia (UTM)
[5]Department of Computer and Information Sciences,
UniversitiTeknologi PETRONAS (UTP), Tronoh, Perak, Malaysia
[3,6,7]Department of Information System, Kulliyyah of Information and
Communication Technology, International Islamic University Malaysia (IIUM)

Abstract: Software Change Impact Analysis (SCIA) has defined as a process of identifying the consequences of the software changes requests. Almost no major corporations are free from the challenge of developing and implementing successful strategies for managing change. Different studies have been conducted in this subject for the last two decades, several approaches have been proposed. But most of these studies have less support to the current process of a software change management. The complexities of software change development nowadays cause the process of managing software change to be difficulty. There is major organizations that are free from the challenges of initiating, developing and implementing effective software changes management. As a result, software practitioners recognize that strategic change is not temporary issues but it's continued process. Only few project managers got the ability to manage th change efficiently. Therefore, this paper investigates issues of change impact analysis process and identifies and compares current practice issues of a software change impact analysis, by evaluating their strengths and weaknesses. It also reviews existing tools and models of change impact analysis and how it supports to the current software change managements. Hence, it proposes existing processe issues and suggests the need of an effective process of software change management.

Key words: Current process issues · Change requirement management · Project management practitioners and change control

INTRODUCTION

Software Changes is a work that cannot be avoided during the software development life cycle from concept to retirement. Always things happen which sometimes require the system to be changed. The hardware and software platform changes, customer needs evolve, defects are found and so on. Consequently, it is difficult to have complete management over and experience what is the common theme of change to project otherwise, it might cause project deterioration and other different problems. Software change control and change impact analysis are sameness in this context. More specifically, the word impact analysis has been used different goals, sometimes it is important for characterizing diverse impact analysis processes, it also supports to identify the effect of the change to the project requirement. Hence, "the process of assessing the implications of realizing a change is termed as change impact analysis" As software project management practitioners recognize, that software development complicity is increasing nowadays, so the need of change impact analysis approach is highly demended. However, this paper identifies existing processes of change impact analysis and assesses issues of these processes. Besides that, it compares the current practice issues of impact analysis in the literature. Finally, it reviews how these issues are associated with Impact Analysis by assess their weakness and strengths of the change management processes.

Current Practice: Performing software change impact analysis an important step when changing software requirement, specifically in incremental processes [1], it permits developers to decide the required work to

Corresponding Author: Hassan Osman Ali, Department of Information System, Faculty of Computing, Universiti Teknologi Malaysia (UTM).

Table 1: Assessment of a Change requirement management tools

Tool	Strengths	Weakness
Borland CaliberRM	Borland is a tool used to manage requirements [8]. It can be divided into two important points: Caliber DefineIT, which stands as software requirement at the first stage of the project. It also used to control the software changes and support the collaboration and communication during the project Lifecycle [8]. It provides several approaches to maintain the outline of the requirement and combines these features to evaluate the effect of the change request. Meantime, it keeps all historical changes to be case studies for the future change requirement management. Second point is traceability links, it stores by checking which artifact are linked and its direction [8]. • This tool keeps the situation of the link and shows whether the link is suspected or not as a result of a change in one of the two related artifacts • The tool uses traceability approach to determine artifacts in detail and to assess and trace the change using traceability link • It allows initiating or tracing the change in multiple links at the same time [9]. • It's also designed to manage the direction of the links.	As defined, CaliberRM tool can be supported only traceability approach for controlling requirements management and change request. CaliberRM Always uses manual to mange the requirement and control the software change impact analysis [8]. To form different artifact type part, it is complicated to assign attitude to an artifact and address it with several groups that can be identified using group attributes [8]. • The tool be supported only Change authorization note filled. • The change can be traced only via traceability links. • The tool supports only traceability change impact analysis approach. • The tool does not support change prioritization to specify the status of the change (normal or argent) • There are no change implementation and verification process. • The tool does not support Configuration management strategy [10]. • It was not designed for Document impacts, cost and estimate decide changes. • It's time consuming to identify the impact of the changes. • Mostly, caliber links are not designed to support mainly change impact analysis. • The tool is only supported small and medium size projects [10].
Heavyweight (IBM Rational Requisite Pro	RequisitePro is software change and requirement management tool which is under IBM's Rationale Suite [11]. It supports to model software change and store them in a relational database. In the tool, change histories which had made by the tool to the requirement is maintained, therefore, it is clear for everybody of the stakeholders what kind of changes has been done and what time. Normally, it can be linked the Requirements and artifacts to each other using RequisitPro through traceability perspective. Using traceability perspective, it can be simply do an impact analysis to describe the consequence of the change to another artifact if one element of the requirement is changed. [10]. Using RequisitePro, it is possible to integrate all Microsoft word documents, therefore, again requirement and artifacts can be interpreted into work document and will allow to control them for any modification or updates them which can come from the tool. The tool also provides an approach to do impact analysis and identify its consequences. • The tool also helps traceability tree view that addresses the project requirement in a hierarchical way. In this view, supports to address the requirement in a graphical way with clear relationships between them to trace it simply. [12]. • The tool supports to capture change impacts and participant collaboration. •It has two possible links to use different kinds of traceability, every one of them has its own meaning	Unfortunately, heavyweight tools are complex, inflexible and costly. Normally, tools with a lot of features are complex because it needs to train the staff very high cost and takes time to understand and how to perform impact analysis and how to use it as well [8]. This tool is costly in price it needs hundreds or thousands of dollars per license, training and maintenance. More specifically, it is difficult to adjust requirements in a traceable way by creating traceability links. • Usually, Graphics and OLE objects have MS-Word limitation, when there is a specific change in an object, it is difficult to reference it and change can be tracked. [11]. • It supports only traceability approach to control and capture the change. • The tool does not support Document impacts, cost and decisions • There is no process of configuration record and configuration management strategy. • This tool does not have the foundation of handling code artifacts or to use a change control system to detect faults [12]. • It is appropriately for small/medium-sized projects. • There is no process of storing the works and observations during the change specification [12].
RTM (Integrated Chipware)	RTM (Requirements and Traceability Management) from Integrated Chipware is a software change and requirement control tool designed to support a large integrated software project development [12]. The tool supports to use for any project size. It helps the software change control and project Lifecycle as well. It supports to perform change management on consumer and other stakeholders • Using this tool, software requirements are controlled and linked to a different project requirements and producing activities such as, source code, change management and design specification. [10]. • RTM designed for uses Oracle a relational database, to keep the artifacts in a traceable manner. • Integrates impact identification and estimation with decision process. • Manages and structures the requirement in a traceable way [10].	RTM does not help object-oriented properties. More specifically, when the project becomes complex it difficult to trace backward [8]. • This tool limits Oracle as the selected database for storage. • Using the oracle database, the requirements databases have relatively limited records • It does not support a dependent approach to identify the change and which element depends on. • It's difficult to differentiate the links to trace impacted elements of the change • There is no Use of a change control system to capture changes [13]. • It also does not support change implementation process [13].

Table 1: Continued

Tool	Strengths	Weakness
RequireIt (Telelogic AB)	RequireIt is a software change and Requirements Management tool, which is based entirely on MS- Word. This tool designed for novice users. it gives change to the users to utilize its existing, Familiar interface [12]. it supports change impact analysis by establishing links among the documents within the same project. Although the tool is based on MS-word, it supports unidirectional traceability with hyperlink. • It is a user friendly tool and can be simply understood, users are familiar with most of the features [14]. • It is easier to trace the change request by using hyperlinks • It supports to identify the impact of the change through MS-Word techniques • The item details are kept and controlled along with the traceable • It has templates used for software Lifecycle control. • Links created during the requirement record support the analysis of the change.	RequireIt tool limits to get change history and identification of a past change request approaches. Further, it ignores database administration to keep the project requirement in a traceable way [11]. • It can only use unidirectional traceability for developing correlation among the requirements in the database. • This tool does not use configuration management • It does not support to document impact, cost and decisions [10]. • It's difficult to determine the type of change. • This tool does not support multi-user requirements, i.e. Using this tool, it can support one team can collaborate on a similar detail at the same time [12]. • It does not integrate the outcome of change impact analysis with the life cycle of the project • It is difficult to describe the impact of a changes on customers and other external stakeholders • It does not support change implementation process.
DOORS AND DOORSNET (TELELOGIC AB)	DOORS, is a software change requirement management tool that designed to use bi-directional traceability. It also allows to change impact analysis [13]. It comprises a complete change request process that allows baseline of the modules [12]. This tool supports: (1) detecting the artifacts by directly inserting them into DOORS, (2) controlling requirements in a traceable way [8]. • The tool can be classified/categorize requirements during change identification • This tool supports change impact analysis. i.e, user can trace the change consequences to any piece of data from the rest of the system [10]. • This tool support to create a relationship between modules in several project requirements or generating traceability reports that indicate how one project impacts another. • It supports specifying changes in traceable approach.	This tool Provides only a single-database repository [13]. As DOORS tool user views, the tool configuration management does not support with high project requirements churn i.e. If for example, 70% of the project requirements in a database change in a short period of time [12]. The tool needs to train a proper setup of the DB with respect to the attributes and traceability is good for suitable saving of project requirement. • It performs very limit in traceability directions (backward and forward) • This tool can only identify changes with goal attribute called No. of modifications. • If the object has been modified without change request process, this script will not recognize as a change to the object [12]. • The tool can only calculate elements that in the proposed change which have already applied • The tool does not consider the change until it's approved or implement. • The tool does not support multi change requests and does not follow a change impact analysis process to estimation accurately.

implement the change request [2]. Several studies have been conducted in this area for the last 20 years. A lot of different processes and approaches have been published. All these processes and approaches are constituted with a set of roles, artifact and activities that are used to manage the software change process [3, 4]. However, most of the large enterprises today used tools and models for managing their software changes and requirement management [5]. These processes are designed to control all change requests and project development Lifecycle. However, some of these tools are software tools others are light weigh tools (pen & paper) [6]; Mostly, these processes focus more on capturing change request, managing requirements, requirementtraceability and managing change requests. However, the following sections will be classified and assessed the current processes of change impact analysis.

Tools Assessment: In general, tool is a process that designed to achieve a specific purpose, especially if the item is not consumed in the process". There are different tools, frameworks and models available in the market and can be classified into a certain numberof categories in order to assess and identify their weaknesses and strengths.

According to Vanita, he has discussed the most populars of impact analysis and how it supports the current practices. Many project developers used these tools to manage their software change requirements management. However, these tools are consisting of a variety of processes [7]. Some of these tools are commercial off-the-shelf software applications such as RequireIt, RequisitePro, DOORS, RTM SLATE, Ultra-lightweight and Lightweight. Thus, the following sessions reviews different tools of software change

management in practice and assess their strengths and weaknesses to identify current issues of software change impact analysis in the field.

However, software Change Requirements are the basis for which the tool needs to pre-plan. The high rate of software failure shows that there is a lack of proper approach to change requirement management. This may cause fixing a lot of change requests. In order to manage software change requests, it's important to have proper requirement management process from the beginning and pre-plan the strategy of software changes. Further, evaluating the requirement metrics in a project development is behavior of a good project requirement management. The various requirements management processes and change management tools which available in the market causes the work to be complicated or increase the number of projects to be a failure. Consequently, these tools are not essentially providing the metrics for change and requirements management process. On the other hand, the following sections will be assessed and specified the strengths and weaknesses of models and frameworks of the current process.

Assessment of Existing Models: In this context, process model, defines what are we going to be done (activities), who will be accountable (role) and what it should be the input and output [7, 15]. These are known as the items or elements of the model [7, 16]. These elements are the constructs of all models, which we have integrated set a framework [17]. Mostly, different researchers have provided several items and there is no consensus for their items [16, 17] but when we look such as activities, roles and artifacts which are mostly discussed in the literature. Based on that, We are synthesizing from the literature define role as a set of responsibilities to be assigned to an actor. It provides a framework for actors to carry out tasks [7, 16, 18-21]. Activity is defined as action performed during the process and has clearly defined objective, entry and exit conditions e.g. writing code for a module or writing a test case etc. [16, 18-20]. Artefacts/Deliverable is defined as product created, used, or modified during a process [14, 18-20]. However, in these sections we have reviewed some concepts presented in the software changerequirement management models in the literature [15, 22-26]. In the following Table 2 shows activities, artifacts and roles/actors.

In Leffingwell and Widrig model [20] ignored the main part of the process which is change implementation. More specially, there are no verification activities shown in this model, thus, it is difficult to understand the proper work of change implementation. It also difficult to know the completion of the process and all activities were not documented in any form, e.g, it is difficult to refer, if someone wants to use the process of change decision next time, there is no detail information in this situation. Likewise, there is no process of change request and who has requested the change and when.

Table 2: Existing activities in a most software change management model from literature

RCM Process Models / Activities	Dean Leffingwell and Widrig Model	Olsen's Model	V-Like Model	Ince's Model	Spiral Model	NRM	S.A. Bohner Model	CHAM	S.A. Ajila Model	Simon Lock Model
Plan for change	√							√		
Baseline requirements.	√									
Establish channel to control change	√			√						
Use a change control system to capture changes	√									
Change Impact on functionality	√		√						√	√
Change Impact on cost	√							√		√
Change Impact on customers and other external stakeholders	√									
Potential of the change to destabilize the system	√									
Negotiation process	√							√		√
Budget reconciliation process	√									
Decision making	√									
Manage change hierarchically	√									
Update documents	√			√				√		
Change creation		√		√					√	√
Change implementation		√	√	√		√	√		√	
Verification		√				√		√		√
Problem understanding			√		√		√			
Solution analysis			√		√			√		√
Solution specification			√		√		√			
Submit solution			√							
Regression testing			√				√			
Acceptance testing (Validation)			√	√		√			√	√
Submission of modification report			√							
Change authorization note filled				√						
Current configuration record updated				√						
Examined from the non-technical viewpoint.					√					
Documenting the actions and observations					√					
Determine the type of change								√		
Estimate effort								√		
Release								√		
cost benefit analysis										√
Document impact, cost and decisions										√
System release planning					√			√		√
System release and integration										√

In V-like model [17], planning the resource on the change control is missing, as whenever there is a change request, it is necessary to estimate the effect of the change to allocate the required cost for implementation, this kind of change is practically possible. Further, impact analysis activity has not been discussed in this model; this activity is used to identify the impact of the change [27, 28]. A change decision which is the main part of the process is missing [29]. However, very limited artifacts were discussed, while artifacts involved in requirement management for change. More specifically, artifact case, which is the main element of the process, is missing [30, 31].

Ince's Model [17] totally ignored the type of decisions to be taken, who will decide the change, what is the strategy for the change and what should be the process of having decision, what kind of details is needed to have proper decision and how the change will look like?. Furthermore, it has verification and validation component to know whether the implemented change has a problem or not? But Limited artifacts were discussed and still the content of the artifacts mentioned are not enough [19]. However, in this mode, decision making activity is missing and it is difficult to know whether the related requirements are needed to update in future or not and process of what approach should be used for the change implementation. There are no testing activities discussed in this model to verify and validated changes.

NBM [32] has developed the activities to be taken at an abstract level, as there is no activity designed to understand the change like that if there is change request that already ordered and already rejected. Thus, why to lose a time again like this change [33], In this case, it can only be possible when the stakeholders have an activity component in their work frame which main target is to identify the impact of the change, however, resources can be used properly. On the other hand, It supports to implement the change in the code, nevertheless [34]. Nevertheless, there are no artifacts item and actor mentioned in this mode, so, it is not clear what should be the outcome of the activity and who should conduct these activities are also another missing area of the model.

On the other hand, S.A Bohner Model [23] is ignored the impact analysis component, that usually used to identify the consequence of the change to estimate the cost needed for the change implementation. If we estimate the required resource for the change implementation, it's important to decide whether the decided change is appropriate to implement now or in a future release [35]. In addition to that, the decision activities which related directly to change request as every change cannot be accepted; this apart also is missing. In the model, there is no information given which shows whether the details about the change process is being developed is documented or not and how to document it. Therefore, actors are also missing.

Besides that, CHAM [35], is only used to estimate impacts on the resource and assesses whether change detail is enough or it's impact on the effort. Besides that, cost benefit analysis can also help in this context, but no testing activity e.g. Acceptance testing or regression testing is not mentioned. Usually, when implementing change, regression test is important to assess whether the change is properly accommodated or not. Furthermore, documenting activity is also missing. Therefore, this model is not supported current process of software change impact analysis. On the other hand, S.A Ajila model [25] is used to identify the consequence of the change on the functionality, but how can that change be finished without assessing the effect of change on requirement, cost and effort Discussion being another key activity as whenever requested a change and its decision cannot be taken in isolation, collaboration among the concern of the project team will be required. No artifacts and actors discussed in this model which adds value to the completeness of the model [37]. Simon lock model [26] is missing change initial stage; there is no process of understanding the change request. Based on that, the change request will be stopped at the initial stage of the process that helps in saving the software cost [38].

Practitioner's Perception: To get practitioners perception and ideas, we adopted survey approach to collect the primary data of the study. The survey method was suitable because, usually it's used to explore subjects' ideas and perceptions and it can empirically test the generalizability of the research outcome. We developed the survey instrument followed by general instrument development guidelines [9, 39]. Although most of the research items are derived from past studies and modified to support software change management but it has not been applied in CIA process improvement context. However, all of the instruments have been validated using two stages recommended by Moore and Benbasat [40]. After discussion with colloquies about the instrument validation to identify ambiguously or poorly worded questions, we conducted a pre-testing further to evaluate the instrument and measure it. Netemeyer [41] have recommended each and every instrument to have five respondent or more for pre-testing. Based on this

suggestion, we firstly distributed the survey in the department of software engineering lectures for pre-testing purpose to review the questionnaire to assure the suitability of the instrument and decide the appropriate time for the instrument answer with respect to their feedback. The researcher has made small changes to the questionnaire introduction and some wording modification to be more clear and answer easily.

Besides that, Straub [39] suggests to test the reliability of the instrument on data from Pilot study to check the suitability of the instrument for primary data collection. Based on that, the researcher has collected pilot study data using a small-scale pilot test for 40 usable responses from expert on small, medium and large software project management enterprises. According to the related few sample sizes of the pilot study, we tested reliability of the data by calculating Cronbach's alpha and item-total correlation. In addition to that, all the scales show preferable reliability (0.93) According to Nunnaly (1978) and Jöreskog and Sörbom (1989) concept, which says, 0.70 is an acceptable Alpha reliability value. After the survey instrument validation process, we developed the final version of survey instrument which consist of four main components such as: (1) survey introduction and invitation to participate: an overview about the research questions was created and its purpose as well as the confidential statement about the respondents answer. (2) A group of demographic questions was presented (3) items measuring the independent and dependent constructs in the research model; (4) Current Issues of IA was also addressed for prioritization purpose.

Furthermore, survey questionnaire were distributed byentrepreneur experts. A sample of population of software development practitioners has been considered for collecting data. The main target group was project board, project manager and deliverables (developers). The majority of the people surveyed were project team, the second majority was project manager and project board was the smallest number. In addition to that, a total of 453 questionnaires were distributed by 25different project management companies in Malaysia, 209 completed and usable questionnaires were returned. These companies were randomly selected from 37 firms lists Published by Malaysian Small, Medium and large Enterprises Info web page, the survey was distributed face to face, two weeks' time frame were given to finish and complete the survey. Besides that, the items used to operationalize, as mentioned earlier were mainly adapted from previous studies and modified for use in the change impact analysis process context. Subsequently, all elements of the survey were measured using a five-point Likert-type scale (ranging from 1 Unimportant to 5 critical).

In addition to that, the overarching objective of the research was to assess the role of Impact analysis in software change management, in order to be able to develop approaches of change impact analysis performance. Further, quantitative approach was performed during the primary data analysis. The researcher used to analysis the data using quantitative method. During primary data collection in quantitative approach, the purpose was to explore the study problem. when quantitative data continue, it is necessary to discover first with a large sample for variable testing, base on that, explore more in depth with some cases throughout the quantitative processes. According to the literature and preliminary data collected during the initial phase of the study, we determined current process issues of impact analysis. In order to improve the existing approach of a software change impact analysis and organizational hierarchical level we identified all problems related to impact analysis process.

On the other hand, 58.4% of the respondents totally disagreed that they do CIA process for their project. Doing so requires having the proper approach of change impact identification and analyzing them in order to be taken the right decisions. The majority of the respondents agrees that change impact identification is very significant activity in their project, they also agreed on not using any standard process of change impact analysis. However, it's important to know, how do they identify change impacts and why they don't use a standard process of change impact analysis for their projects? Based on that, most of change impact identification for software development companies handled them manually and they don't have any guideline or effective process that can support them to have an accurate decision for their long term project goals. More specifically, practitioners approved that current impact analysis process is not support to the current software change requirement management process.

Integration of Models and Tools' Issues: This classification was created after assessing the existing work which aimed to identify the issues of a current process of software change impact analysis. We have reviewed some existing models and tools of software change impact analysis to compare in order to distinguish their problems to understand what is the main cause of

Table 3: Change impact analysis issues

Current Practice Issues of Impact analysis		
Existing Models & frameworks Issues	**Exist Tools issues**	**Practitioners view Issues**
❖ Relevant structure and change specification to support the analysis are missing	❖ The tools does not support multi change requests and does not follow change impact analysis process to have accurate decision.	❖ Most of current practitioners in Malaysian use their own method of managing changes.
❖ It is challenging to manage conflicting and synergetic change requests	❖ Change Impact on customers and other external stakeholders are complicated.	❖ It is very challenge to get resources for conducting impact analysis
❖ System impact is underestimated or overlooked	❖ The tools does not provide Baseline requirements for the change	❖ There is not enough time to carry out impact analysis.
❖ Requirements and baseline are missing for early change requests.	❖ It does not support to document impact, cost and decisions (Zhang, 2007).	❖ Analysis require much expertise and experience
❖ Responsibility and product/project balance are difficult to handle for analyses that span several systems.	❖ It does not integrate the outcome of change impact analysis with decision process	❖ Analyses and change implementation evoke stress.
❖ Different change request have different levels of complexity, and there is no strategy for appropriate decision.	❖ It difficult to determine the type of change.	❖ There is no standard Impact Analysis process strategy for the complexity of software projects nowadays
❖ Iit's difficult to identify the impact of a change for estimation (Zhang, 2007).	❖ There is no Use of a change control system to capture changes (Gotel, 1993).	❖ Mostly, stakeholders don't like to perform impact analysis
❖ There is no option for determining the type of change.	❖ Using the oracle database, the requirements databases have relatively limited records and decision making	❖ Change request decisions are based on interest
❖ Existing processes of models and frameworks are not appropriate for the current software requirement changes	❖ There is no process of documenting the actions and observations during the change identification (Grinter, 1996).	❖ Analysis are performed by the wrong persons
	❖ Supports only traceability approach to control and capture the change.	❖ Its difficulty to classify the change using change impact analysis
	❖ It is quite difficult to accurately compute the impact of changing dynamic point cuts.	❖ Solutions are specified with too much detail by high-level analysts to perform accurate decision.
	❖ The tools does not support change Specification activities to identify the rate and status of the change	❖ Impact analysis checklist process is not perfectly supported current software projects.
	❖ Mostly, caliber links are not designed to support mainly change control process.	

the problem? On the other hand, during our specification and identification of existing processes issues, we identified the main issues of the process which consist of two important processes, such as, Models and tools issues and practitioner's perspective. These three issues were classified and addressed in separate.

In addition to that, we recognize that, most of the existing processes are not supporting current software change requirement management. For example, most of the tools used manual to specify impact of the requested change while other processes same like software requirement management do systematically. There is no help for indirect correlation in all tools excluding Top Team Analyst and in RequisitePro and Caliber only links which are in a same path are used, so a manual check for effected requirement is still needed. The biggest problem with this tool is for example, the DOORS Change request process, permits a dedicated group to do the analysis and keeping the overview. It does not use to support link during analysis yet, but using the script language and it can be made more useful. Further, in this tool, software change request process is an exceptional feature that adds value for large software projects. Furthermore, most of this process requires high level of knowledge, not having only potential application of the tool, but in the real use of the process base itself [24]. More specifically, These Tools are enabled to support existing software change control process. But, most of these tools are able to specify the change request process in traceability approach only. More specifically, it can help common functionalities only [7, 24]. Hence, the issues of these tools are very important to consider and come out with appropriate solution.

Consequently, the details of the issues will be discussed in the following Table 3.

On the other hand, the second reviewed approach of a current process are models, these processes were designed to help software change management. In this study, we have conducted a review of change impact analysis models; this review reveals that there are a lot of distinctions among their descriptions. We didn't find any similar pattern within these models; Most of the models which we have analyzed above are lack of deep detail of the approaches. There was no clear or defined

relationship between these processes (roles, artifacts and activities). Therefore current change impact analysis processes are not supported to the project stakeholder's. This situation lets researchers by area to verify in detailed and improve like Software Development process models.

CONCLUSION

This paper summarizes our review. It shows that software changes requirements management tools are more famous and used instead of purely traceability tools and limited process. Each tool has some strong and weak points. Others have limited support. But none of the processes are supporting change impact analysis process effective, so, there is a need to do more research in this direction, On the other hand, we have also presented current practice issues of Impact Analysis from two important sources in the literature and practitioners. This review confirms issues of current change impact analysis process and reveals that there are a lot of weaknesses and differences between the descriptions. All processes are lack of detail in their approaches. There is no clear correlation between activities, artifacts and roles/actors. However, current processes of change impact analysis have less support to the software change requirement management. This situation calls for a change impact analysis process improvement in a lightweight manner and developing a detailed approach that can support practitioners to use for their software change processes.

REFERENCES

1. Rajlich, V. and P. Gosavi, 2004. Incremental change in object-oriented programming, IEEE Software, 21(4): 62-69.
2. Bohner, S.A., 2000. Impact analysis in the software change process: a year 2000 perspective, in Proceedings of the 12th International Conference on Software Maintenance (ICSM'96), Monterey, CA, pp: 42-51.
3. Chen, Chung-Yang and Pei-Chi Chen, 2009. A holistic approach to managing software change impact. Journal of Systems and Software, pp: 2051-2067.
4. Nuseibeh, Bashar, Jeff Kramer and Anthony Finkelstein, 2003. ViewPoints: meaningful relationships are difficult, Software Engineering. Proceedings. 25th International Conference on. IEEE.
5. Davis, Jesse and Mark Goadrich, 2006. The relationship between Precision-Recall and ROC curves. Proceedings of the 23rd international conference on Machine learning. ACM.
6. Karlstrom, D., Per Runeson and Claes Wohlin, 2002. Aggregating viewpoints for strategic software process improvement-a method and a case study. Software, IEE Proceedings, 149(5). IET.
7. Leffingwell, Dean and Don Widrig, 2000. Managing software requirements: a unified approach. Addison-Wesley Professional.
8. Morkos, B., 2009. Analysis of DOORS.
9. Moore Gary, C. and IzakBenbasat, 1991. Development of an instrument to measure the perceptions of adopting an information technology innovation. Information Systems Research, pp: 192-222.
10. INCOSE, S.E.H., 2006. A Guide for System Life Cycle Processes and Activities. no. TP-2003-2002-2003 in Version 2003, Ed.
11. Hoffman, *et al*., 2004. Hoffmann, M., Kuhn, N., Weber, M. and Bittner, M. Requirements for Requirements Management Tools. In Proceedings of the 12th IEEE International Require-ments Engineering Conference (RE '04),
12. VanitaShroff, 2001. Requirements Management Metrics. Department of Electrical and Computer Engineering. University of Calgary, Calgary, Alberta, CANADA. Master's in Software Engineering Comprehensive Project Report.
13. Cant, T., J. McCarthy and R. Stanley, 2006. Tools for Requirements Management: a Comparison of Telelogic DOORS and the HIVE, Australian Government Department of Defense, Defence Science and Technology Organisation.
14. Gallagher, Keith Brian and James R. Lyle, 1991. Using program slicing in software maintenance. Software Engineering, IEEE Transactions, 751-761.
15. Williams, L.G., 1989. A Behavioral, Approach to Software Process Modeling, ACM SIGSOFT Software Engineering Notes, 14(4): 167-170.
16. Feiler Peter, H. and Watts S. Humphrey, 1993. Software process development and enactment: Concepts and definitions. Software Process, Continuous Software Process Improvement, Second International Conference on the. IEEE.
17. Hattori, L., D. Guerrero, J. Figueiredo, J. Brunet and J. Damasio, 2008. On the Precision and Accuracy of Impact Analysis Techniques. in 7th IEEE/ACIS International Conference on Computer and Information Science, Portland, Oregon, USA, IEEE Computer Society.

18. Curtis, B., M.I. Kellner and J. Over, 1992. Process Modeling, Communication of the ACM, 35: 75-90.
19. Dowson, Mark, Brian Nejmeh and William Riddle, 1990. Concepts for process definition and support. Software Process Workshop, 'Support for the Software Process', Proceedings of the 6th International. IEEE.
20. Lonchamp Jacques, 1993. A structured conceptual and terminological framework for software process engineering. Software Process, Continuous Software Process Improvement, Second International Conference on the. IEEE.
21. Lavazza Luigi and Giuseppe Valetto, 2000. Enhancing requirements and change management through process modelling and measurement. Requirements Engineering, Proceedings. 4th International Conference on. IEEE.
22. Lam, W., *et al*., 1998. Change analysis and management: a process model and its application within a commercial setting. Application-Specific Software Engineering Technology, ASSET, Proceedings. IEEE Workshop on. IEEE.
23. Makarainen, M., 2000. Software change management process in the development of embedded software, Dissertation, VTT Technical Research Center of Finland, ESPOO.
24. Silvia, A.T. and F. Xavier, 2001. Software Process Modelling. World Multiconference on Systemics, Cybernetics and Informatics SCI, Orlando, FL,(USA), pp: 25.
25. Nadi, S., 2009. DRACA, Decision-support for Root Cause Analysis and Change Impact Analysis, in Computer Science University of Waterloo: Waterloo, Ontario, Canada, pp: 87-98.
26. Goknil, A., I. Kurtev and K. Van den Berg, 2008. Change Impact Analysis based on Formalization of Trace Relations for Requirements. in ECMDA, Traceability Workshop, ECMDA-TW, Berlin, Germany, pp: 59-75.
27. Olsen Neil, C., 1993. The software rush hour (software engineering). Software, IEEE, pp: 29-37.
28. Lock, S. and G. Kotonya, 1999. An integrated framework for requirement change impact analysis. Proc. of the 4th Australian Conference on Requirements Engineering, Sydney, Australia.
29. Mäkäräinen Minna, 2000. Software change management process in the development of embedded software. VTT, Technical Research Center of Finland, ESPOO, pp: 699-712.
30. Prikladnicki, Rafael, Jorge Luis Nicolas Audy and Roberto Evaristo, 2006. A reference model for global software development: findings from a case study. Global Software Engineering, ICGSE'. International Conference on. IEEE.
31. Bohner Shawn, A., 1996. Impact analysis in the software change process: A year 2000 perspective. Software Maintenance, Proceedings. International Conference. IEEE.
32. Zhang Sai, *et al*., 2008. Celadon: a change impact analysis tool for aspect-oriented programs. Companion of the 30th international conference on Software engineering. ACM.
33. Ramzan, S. and N. Ikram, 2006. Requirement change management process models: Activities, artifacts and roles. Multitopic Conference, INMIC. IEEE.
34. Ramzan, S. and N. Ikram, 2005. Making Decisions in Requirements Change Management. Information and Communication Technologies. ICIT First International Conference, 27-28: 309.
35. Ajila, Samuel, 1995. Software maintenance: an approach to impact analysis of objects change. Software: Practice and Experience, pp: 1155-1181.
36. Abma, B.J.M., 2009. Evaluation of requirements management tools with support for traceability-based change impact analysis, Software Engineering Faculty of Electrical Engineering, Mathematics and Computer Science University of Twente.
37. Bohner, S. and R. Arnold, 2002. Software Change Impact Analysis. IEEE Computer Society Press, Los Alamitos, CA, USA, pp: 246-256.
38. Vallabhaneni S. Rao, 1987. Auditing the maintenance of software. Prentice-Hall, Inc.
39. Appleton James, J., *et al*., 2006. Measuring cognitive and psychological engagement: Validation of the Student Engagement Instrument. Journal of School Psychology, pp: 427-445.
40. Moore Gary, C. and Izak Benbasat, 1991. Development of an instrument to measure the perceptions of adopting an information technology innovation. Information Systems Research, pp: 192-222.
41. Boyar Scott, L., *et al*., 2003. Work-family conflict: A model of linkages between work and family domain variables and turnover intentions. Journal of Managerial Issues, pp: 175-190.