International Journal of Innovation and Scientific Research
ISSN 2351-8014 Vol. 14 No. 1 Mar. 2015, pp. 10-20
© 2015 Innovative Space of Scientific Research Journals
http://www.ijisr.issr-journals.org/

Effective Software Change Control Model for Minimizing Change Effects

Bilal Hussain, M. Yahiya Saeed, Syed Sajjad Naeem Shah, Adnan Ahmad Malik, and Atif Mahmood

Computer Sciences,
CS Department University of Agriculture Faisalabad,
Punjab, Pakistan

ABSTRACT: The purpose of this research was to satisfy the customer by listing customer voice and to response positively on their demands. The software is designed for a system or to assist a system, the software development process is based on the analysis phase the next phases are design, code and testing. The system acceptability and usability teaches people to demand more that is why changes occurs during the developing cycle and after the delivery of product, so if change is not managed properly then it can affect the software overall performance and workload on the development team that have no benefit. To reduce these problem development team needs an effective change control model that have defined set of rules to implement change. The current study has been based over change indicators and their role in the general software change development. In this research an effective change control model is proposed, this model is based over the classification of the change, that model calculates the impact of a change and reduce time and cost of maintenance process. This model makes the software more customizable and adaptable moreover it increases the customer confidence on the software development team.

KEYWORDS: software development process, customer voice, Change Control Process, Change Impact, Effective Change Control Model, Change Indicators.

1 INTRODUCTION

The information technology is most growing field now days. Most of the organizations have changed their working process manual to computerized due to the fast processing of data and less error occurrence with the passage of time software customer is becoming more IT literate, now they understand their needs due to this customer demanding more than before that's way changes now a day comes more often from customer side during the development process and after development.

Through an effective change control mechanism risk can reduced and product failure chances can be minimized in projects like space, transportation, defense avionics but in medical field the devices are used have more importance regarding a change because these devices have an direct effect on the human life [1].

Changing requirements for software development projects are the main source of risk. To minimize the change impact there are number of change model already proposed. It is suggested that the two types of threats as the main method for predicting sensitivity analysis: change in design requirements unstable and where the most vulnerable areas identified needs. Processes of change impact analysis support the decision-making process, and also to assess the risks of the potential impact of change is used to predict [2].

Research is based on the over the overall study of software change management process the software developers develop software for different situation and for different organizations that might have different business cycle that cycle can be change after some time or by adopting some new strategies or change can be occur by change in the organizational structure there is any other reason. Development team's main focus to satisfy customer, customer will be more satisfied when he has its desired feature in his product. The success of software is not easy to measure there is clear indication in two categories first success is the software success which is primary success (software functionality) and second success is its future success (change adaptation).

1.1 SOFTWARE CHANGE

In today world where trends are changing rapidly in the development of software systems, it is very rare that the initial design or first deliverable is considered as the end product. It is inevitable that the software design will require changes during the software life cycle. in that scenario, the provided solutions that should be flexible enough to adopt changes in the future [3].

Major issue for software development team is maintenance of software after the delivery of the software. This may includes new customer requirements, error correction software update. Maintenance management lies in the information technology products services, representing more than half of the total software development cost. A key reason for the high cost is software updates and error corrections. Software development and management services in the field of contradictions that occur when changes inevitably result in further price increases, which leads to more maintenance problems, confusion and misunderstanding [4].

1.2 ROLE OF CHANGE AGENTS

Many people in an organizations is involved in promoting change with some justification and having the knowledge of the range of change these people known as agents of change. Change agents are those persons who are directly or indirectly relate with the system. It is not generally categorized that change must be proposed by the sponsor, champion [5].

Table 1. People and their roles in change control

Role	Activities
1. Change Initiator (anyone in the organization)	• Proposes a change based on need • Documents the change needed, and submits it using the established practice or process • Justifies the change, and acts as its proponent • Promotes the change until approved
2. Change Agent	• Provides insight into the change process • Provides advice that addresses barriers, resolves issues, and generates results • Acts as an independent set of eyes and ears as the change process unfolds
3. Champion (a project pilot or other mid-level manager)	• Promotes the change at the middle management level • Provides the resources to implement the change • Removes obstacles that are under his or her control • Addresses issues, and generates results
4. Sponsor (an executive-level supporter)	• Promotes the change at the senior management level • Provides executive support and coaching • Buffers the team against interference from above • Provides the budget, and protects it against pilfering
5. Opinion Leaders (technical leaders who are respected by their peers)	• Makes the technical case for change (shows that it's the right thing to do) • Convinces others that the change makes sense • Takes charge of overseeing the work
6. Infrastructure Managers (process group leads or marketing personnel)	• Adjusts the infrastructure to accommodate change • Promotes the change through actions and deeds • Promotes the change through infrastructure resources (such as newsletters, websites, and success stories)
7. Implementation Team	• Implements the change • Verifies that the change does what it was supposed to do • Believes that the change is the right thing to do • Provides evidence that the change is beneficial and meets quantitative • objectives, and if it does not, documents failure, declares success, and moves on

1.3 CLASSIFICATION OF CHANGE

In this research the change is classified into three types on the basis of the effect of change on the other system, two types of modifications, emergent and initiated changes defined by [6].

Emergent changes are “caused by the design’s state, where error occur across the whole design of the software and throughout the software lifecycle can lead to implement change”. In other words, unanticipated problems or knock-on effects require additional, emergent design modifications. Given this perspective, design errors due to human mistakes, unlike more fundamental flaws in the design, are not considered emergent changes since they are not explicitly caused- by the design’s state. In turn, Eckert define initiated changes as “change rising from an outside source, generally a new requirement from customers, or change can be initiated by the manufacturer”. Such initiated changes may be difficult to control, although, in some cases, negotiation with customers and manufacturers can help to manage these modifications. If the complete impact of an initiated change is not determined, unexpected, emergent changes can certainly arise [7].

1.4 COSMETIC CHANGES

These change are very beginning level change not required to much efforts to implement and it also don’t have any influences on the other working of the system, these include changes like:

- Screen layout
- Adding or removing field on output report
- Changing font size or color scheme etc

1.5 LOCAL CHANGES

Local changes have local effect on the system development team have a little concern before implementing that change but not need to worry too much about the effects of that change these changes include:

- Adding or removing new field in the database
- Changing the input fields on the data entry forms
- Regrouping of the data entry form such as instead of two forms getting information on one or vice versa.

1.6 GLOBAL CHANGES

Global changes have a major effect on the working of the system these change are should be accepted through a Change Control Board that changes include:

- Major alteration in the code
- Cross or dependent function change
- New functionality adding to the system

- Cosmetic - trivial
- Local - relatively local impact on the system
- Global - may have significant impact on other
- parts of the system

Fig. 1. Classification of change

2 BACKGROUND

Bohner [3] presented a theory with the development of software engineering practice, the search platforms and heterogeneous distributed respond to requests for changes to the software middleware in terms of players and parts. Dealing with dependency now points more inevitable impact of software changes and expression analysis. Software modifications including middleware components, such as a Web service exposes these organizations to serve the unforeseen chain reaction, often leading to failure of the system software. The current software change impact analysis model does not adequately deal with the situation. In addition, with the expansion of the size and complexity of software programs, give information on their web comprehension of software engineers. Preparing the research paper examines the expansion of the existing software change impact analysis to be interoperable distributed applications to resolve dependencies and to examine the three dimensional (3D) visualization software to more efficient navigation.

Stojanov [8] described that a basic goods have to comply with changes in the life cycle of software. Main problem in the management of change process specified by the customer requests a change after processing software delivery. Change the client requests using the developer's website is often presented. This article explains where the customer produces a change request to produce a new way to help change request. Model Send change requests generated includes three views: the path of the application, it may request the generator and change, change requests based model itself and generate change requests model integration. Activate the switch requests a specific XML format, send the developer's site via HTTP.

Sherriff and Williams [9] described that verification and validation techniques often make different kinds of software development artifacts. Build a shift from ensuring and validation work records show how the files tend to change in response to correct mistakes and failing to see them together. We propose to determine the record software capabilities to analyze the impact of the new system changes by adjusting the unity value. This approach builds clusters file, history has a tendency to change the address mistake and a failure of the library code. We conducted a case study using this process after five open source software systems. We have determined that our strategies are effective in identifying the stakeholders from the file system changes from the developers always tend to make it smaller, attack the source of updates. We also compare our analysis techniques technology and the other two results; we find our technology provides the same results, but also to ensure that it can be influenced by changes in non-source files.

Mayhew, Worsley [7] described that the development of the system, is currently widely discussed, although they were a bit more. The main reason for this situation is project manager worried about how to control the process is iterative development mode example. This paper describes a method for controlling the power based prototyping using the normal changes occurring within the framework. This change in stages for controlling the pattern is passed, the development of monitoring and evaluation for commercial Royal Dockyard, while the English were evaluated.

3 EFFECTIVE CHANGE CONTROL MODEL

After studding the already purposed model for change development and analyzing the drawback of the previous work a research and change control model is developed that effectively manage change, for small and large industries that change model works perfectly it requires less effort to incorporate change and also to make decision that a change going to accept or reject quickly [10].

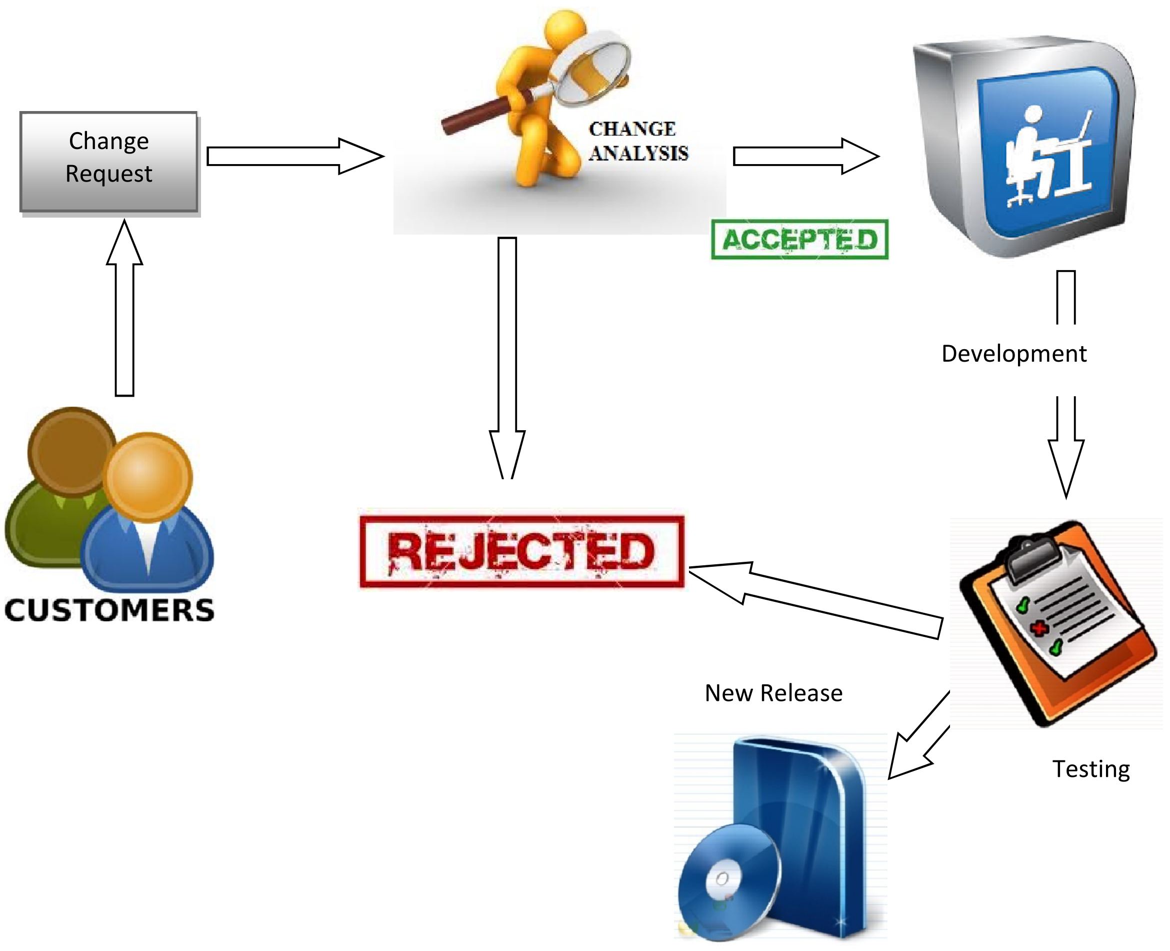


Fig. 2. Effective change Control Model

3.1 SUBMITTING CHANGE OR CHANGE REQUEST

Now a day's customer is more demanded as discussed before in section no 1 most of the time change is occurred from the customer end most but is not necessary that each change is feasible to implement by the development team to verify a change there is an change control board in a development environment.

3.2 SOFTWARE CHANGE CONTROL BOARD

Control Change (CCB) or software change Control Board (SCCB) of the Committee of the Board, so that the decision-making on the proposed changes or software project should be implemented. In short, any change in the basic requirements agreed with the customer in, approved by the project team should be taken in this committee. If any changes are made to the committee, which is communicated to the project team and customer needs change baseline consent. Change Control Board by the project stakeholders or their representatives. Authorized change control board can be from different projects and different projects, but is determined by the often accepted as final and binding control changes made to the board of directors. The decision to accept the change also depends on the stage or stages of the project. CCB or SCCB main purpose is to ensure that the project acceptance (delivery) customers [11].

3.3 ANALYSING CHANGE

Indicate that changes size has direct relation to changes impacts, and moreover small changes has lesser impact than larger changes. Therefore, change types in our study are based on the size of the requested change. We classified the requirement change types that affect the change impact size in nine change types, each change type has a change type factor (CTF) value according to the nature of the requirement change type. The CTF value is a coefficient of correlation value between -1 and 1. It is used to predict the effect of the change type on the change impact size. Table I shows the requirement change types and their nominated CTF values. Furthermore, here are a set of brief descriptions for the effect of each requirement change type [12].

1) Addition:

It is a new requirement, and all new code will be added to the original code.

2) Modification-Major Grow:

Three quarter of the estimated impact code will be added to the original code.

3) Modification-Grow:

Only half of the estimated impact code will be added to the original code.

4) Modification-Minor Grow:

One quarter of the estimated impact code will be added to the original code.

5) Modification-Negligible:

The change is so insignificant that code size will not be changed much.

6) Modification-Minor Shrink:

One quarter of the estimated impact code will be removed from the original code.

7) Modification-Shrink:

Only half of the estimated impact code will be removed from the original code.

8) Modification-Major Shrink:

Three quarter of the estimated impact code will be removed from the original code.

9) Deletion:

The requirement will be deleted, and the codes generated by this requirement will be removed from the

Table 2. Assigning change type value

Change Type	CTF
Addition	1.00
Modification-Major Grow	0.75
Modification-Grow	0.50
Modification-Minor Grow	0.25
Modification-Negligible	-0.10
Modification-Minor Shrink	-0.25
Modification-Shrink	-0.50
Modification-Major Shrink	-0.75
Deletion	-1.00

For evaluating this empirical research results and measuring the accuracy of the proposed approach an explanatory case study evaluation method with several controlled change experiences is used. According to the guidelines described in[13], the five steps for evaluation with case study method are:

(1) Design
(2) Preparation
(3) Collection,
(4) Analysis
(5) Report.s

These steps have been performed thoroughly according to the guidelines[14].

Table 3. change Request Types

C.R No	Change Type
CR 1	Addition
CR 2	Modification Grow
CR 3	Modification-Negligible
CR 4	Modification-Shrink
CR 5	Deletion
CR 6	Modification-Grow & Shrink
CR 7	Modification-Grow & Negligible
CR 8	Addition & Modification
CR 9	Deletion & Modification-Shrink
CR 10	Deletion & Addition

3.4 Development

The development process start after analysis of the change and after an accepting change a complete plan is designed for implementing change in this design and builds iterations.

3.5 Testing

Testing the product after incorporating changes is also important in this phase not only new developed or modified portions is test but it include almost all the product to check the product is working in an order if errors or bugs inducted then make adjustments in the code if possible if not then reject the changes

3.6 New release

Update the documentation by adding change implementation detail assign a new release number to the document that will helps to guide in making future changes to the product and after that changed software or product delivered to the customer after delivery of the product customer response also noted down

4 Results

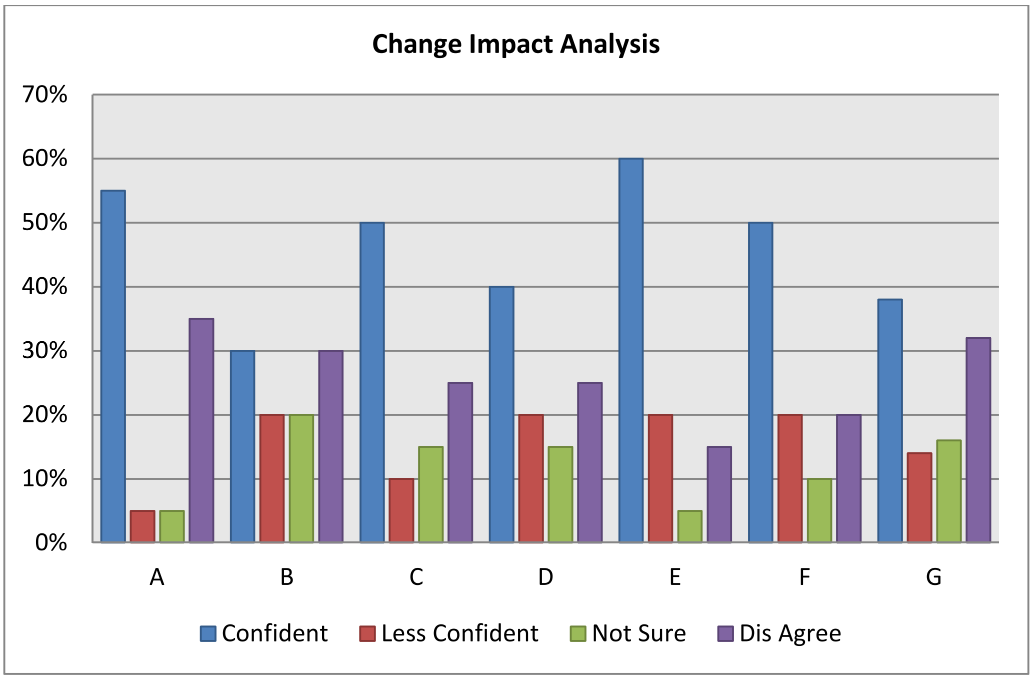
Survey Questionnaire

It is states that questionnaires can be used to reach a wider participant group, it takes less time to administer, and it can be analyzed more rigorously. It can also be administered at various points in the design process, including during requirements capture, task analysis and evaluation, in order to get information on the user’s needs, preferences and experience. Given that the evaluator is not likely to be directly involved in the completion of the questionnaire, it is vital that it is well designed. The first thing that the evaluator must establish is the purpose of the questionnaire: what information is sought? It is also useful to decide at this stage how the questionnaire responses are to be analyzed. For example, do you want specific, measurable feedback on particular interface features, or do you want the user’s impression of using the interface? There are a number of styles of question that can be included in the questionnaire. These include general, open-ended, close-ended or multiple-choice, scalar, ranked,[15].

In this section presents the results and findings of Change Control Model already presented and the deficiencies these models have in it conducted by the author. The questionnaire methodology was used to gather information for evaluation purpose [16].

Table 4. change impact analysis

	Change Impact Analysis	Strongly Agree	Agree	Not Sure	Dis.. Agree
A	Change size should be calculated for planning the change implementation?	55%	5%	5%	35%
B	Does u think that the dependencies matrices are good tool for analyzing change impact?	30%	20%	20%	30%
C	There is a need of new techniques to analyze the change impact?	50%	10%	15%	25%
D	No of Error occurrence chances can be reduced with good Impact Analysis?	40%	20%	15%	25%
E	With the help of impact analysis development team gets an idea what are others areas need to be changed?	60%	20%	5%	15%
F	Do we need a new impact analysis technique for batter estimation?	50%	20%	10%	20%
G	Impact Analysis can reduce the development time?	38%	14%	16%	32%

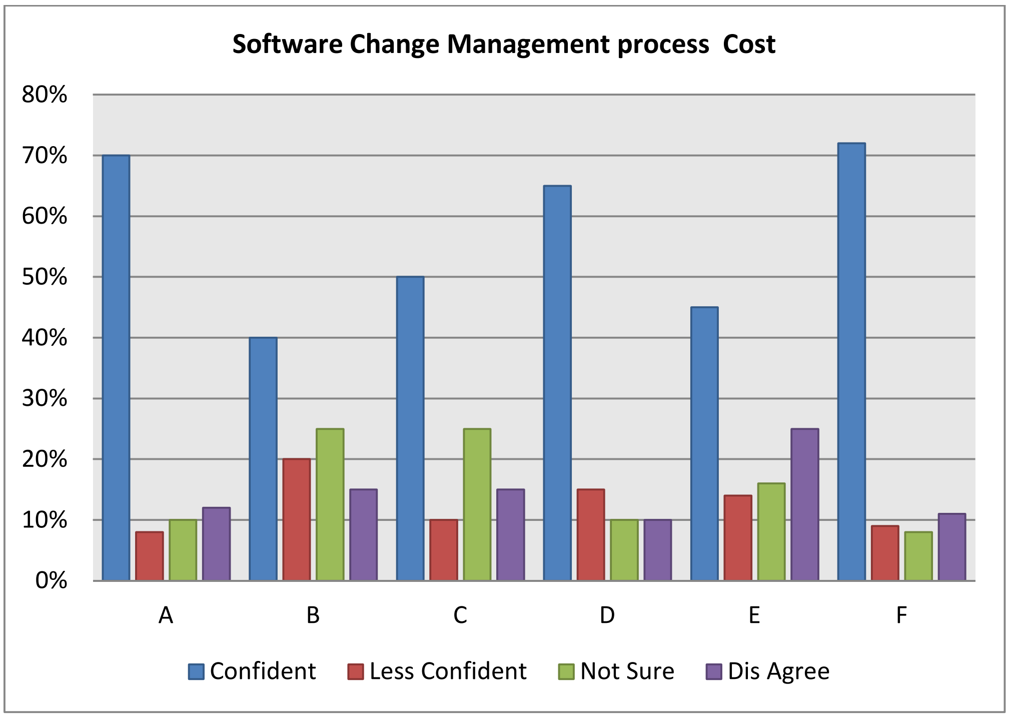


Graph.1. Graphical Representation of response change impact analysis

The above is the bar graph of the change impact analysis. The graph shows the trend and awareness of the professionals regarding change impact analysis. Colored Lines are shown to clear the true status. The level1 is quite dominant in the above graph.

Table 5. Software Change Management process Cost

	Software Change Management process Cost	Strongly Agree	Agree	Not Sure	Dis.. Agree
A	Software maintenance cost is grater then its development cost?	70%	8%	10%	12%
B	Cocommo cost estimation model calculate cost efficiently?	40%	20%	25%	15%
C	Change cost can be estimated well from an experience of past projects?	50%	10%	25%	15%
D	Do you think change process should cost estimation in it?	65%	15%	10%	10%
E	Do you think some changes get rejected because of greater cost?	45%	14%	16%	25%
F	Cost is added for the new requirement?	72%	9%	8%	11%

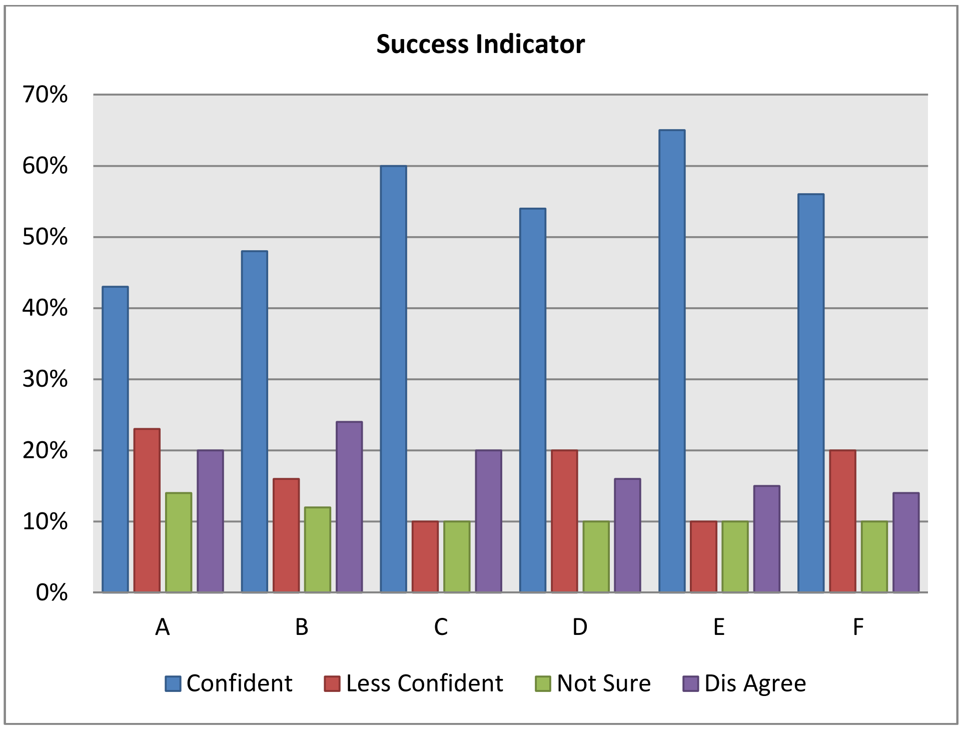


Graph.2. Graphical Representation of response Software Change Management process Cost

The above is the bar graph of the **Software Change Management process Cost**. The graph shows the trend and awareness of the professionals regarding **Software Change Management process Cost**. Colored Lines are shown to clear the true status. The level1 is quite dominant in the above graph.

Table 6. success indicator

	Success Indicator	**Strongly Agree**	**Agree**	**Not Sure**	**Dis.. Agree**
A	Change success indicators are points of change managed properly?	43%	23%	14%	20%
B	With the help of success indicator one can easily judge the quality level do change?	48%	16%	12%	24%
C	Success indicator describes the level of customer satisfaction?	60%	10%	10%	20%
D	Success indicator based on the number of steps requires implementing a change?	54%	20%	10%	16%
E	If a change is passed through all success indicator it will got 100 % score?	65%	10%	10%	15%
F	success indicator metrics increase the customer trust on the development team	56%	20%	10%	14%



Graph.3. Graphical Representation of response Success Indicator

The above is the bar graph of the **Success Indicator**. The graph shows the trend and awareness of the professionals regarding **Success Indicator.** Colored Lines are shown to clear the true status. The level1 is quite dominant in the above graph.

5 Conclusion

The key objective of this research was to introduce a effective software change control model. This research is expected to be help for decision making so that users who is in software development field could take benefit of that model and research. For this purpose almost the entire previous model was studied and discussed with the experts. This research concluded that. The Previous Software Change Models has some deficiencies like process is too long for implementing change, change is not properly handled it creates lots of hurdles for development team and users confidence level is also effects with the attitude of the development team when their demands or requirement rejected down more over small organization don't have a proper team for analyzing change due to lack of resources. Most of the developer does not get satisfactory response by using old models. This is leading towards the dissatisfaction from this old change control processes. After analyzing the customers and developer trends a Effective Change Model is proposed and evaluated, and response is satisfactory.

Acknowledgement

The work presented in this manuscript is accomplished under the sympathetic attitude, animate directions, observant pursuit, scholarly criticism, cheering perspective and enlightened supervision of **Mr. NayyarIqbal, Lecturer Department of Computer Science University of Agriculture, Faisalabad**. His effort towards the inculcation of spirit of handwork and maintenance of professional integrity besides other valuable suggestions always serves as a beacon light through the course of my life.

REFERENCES

[1] Neumann, P., Computer-Related Risks. 1995. Reading: Addison-Wesley, 1995.

[2] Strens, M. and R. Sugden. Change analysis: *a step towards meeting the challenge of changing requirements. in Engineering of Computer-Based Systems*, Proceedings., IEEE Symposium and Workshop on. 1996. IEEE.

[3] Bohner, S.A. *Software change impacts-an evolving perspective. in Software Maintenance*, 2002. Proceedings. International Conference on. 2002. IEEE.

[4] Ping, L. and L. Yang. *A Change-Oriented Conceptual Framework of Software Configuration Management. in Service Systems and Service Management*, International Conference on. 2007. IEEE.

[5] Mosley, V., *How to assess tools efficiently and quantitatively.* Software, IEEE, 1992. 9(3): p. 29-32.

[6] Eckert, C., P.J. Clarkson, and W. Zanker, *Change and customisation in complex engineering domains*. Research in Engineering Design, 2004. 15(1): p. 1-21.

[7] Mayhew, P., C. Worsley, and P. Dearnley, *Control of software prototyping process: change classification approach*. Information and Software Technology, 2009. 31(2): p. 59-66.

[8] Stojanov, Z. *Model of Change Request Generator Integrated into Business Application. in Intelligent Systems and Informatics*, 2007. SISY 2007. 5th International Symposium on. 2007. IEEE.

[9] Sherriff, M. and L. Williams. *Empirical software change impact analysis using singular value decomposition. in Software Testing, Verification, and Validation*, 1st International Conference on. 2008. IEEE.

[10] Ebner, G. and H. Kaindl, *Tracing all around in reengineering*. IEEE software, 2002. 19(3): p. 70-77.

[11] Mrozek, Z. *Design of the mechatronic system with help of UML diagrams*. in Proc. 3-rd Workshop Robot Motion and Control, Bukowy Dworek, Poland. 2002.

[12] Shao, D., S. Khurshid, and D.E. Perry. *Semantic Impact and Faults in Source Code Changes*: An Empirical Study. in Software Engineering Conference, 2009. ASWEC'09. Australian. 2009. IEEE.

[13] Runeson, P. and M. Höst, Guidelines for conducting and reporting case study research in software engineering. Empirical software engineering, 2009. 14(2): p. 131-164.

[14] Breech, B., M. Tegtmeyer, and L. Pollock. *A comparison of online and dynamic impact analysis algorithms*. in Software Maintenance and Reengineering, 2005. CSMR 2005. Ninth European Conference on. 2005. IEEE.

[15] Dix, A., Human-computer interaction. 2009: Springer.

[16] Parsons, K., et al., *Determining employee awareness using the Human Aspects of Information Security Questionnaire* (HAIS-Q). Computers & Security, 2014.