STABLE REQUIREMENT SPECIFICATION FRAMEWORK: REQUIREMENT VOLATILITY PERSPECTIVE

Mohammad Faisal[a], Dr.Md.Rizwan Beg[b], Halima Sadia[c]

[a]Department of CA, Research Scholar, Integral University, Lucknow 226026,India
[b]Department of CSE, Professor, Integral University, Lucknow 226026,India
[c]Department of IT, Integral University, Lucknow 226026,India

Abstract: Software development life cycle constitutes of various stages, among which collection of requirements is carried out in an early phase. Requirement provides the foundation of the software development process [1]. Requirements form the basis of various software activities such as cost estimation, planning project schedules as well as for designing and testing specifications [2]. Requirements keep on changing throughout the SDLC. These Volatile requirements are considered as a major risk factor during system development. These changing requirements have a significant impact on project schedules & cost overruns. This paper proposes a framework to manage the requirement volatility in this rapidly changing environment.

Keywords: Software Development, Software Requirement, Requirement Volatility

---

1. Introduction

The development of software is always considered to be a high risk activity with high failure rate. For developing a software project, requirement collection is carried out from the users, developers, and other stakeholders. Successful elicitation of requirements is considered as one of the major risk factors that affect project schedule, budget and quality. Standish Report [3] indicates that requirement, the contract between customers, end users and the software development organization that defines what gets produced, "is a primary source of software project risk and software defects."

Software development as a dynamic activity causes the requirements to be changed even though development is in progress. These recurrent requirements changes may generate significant project uncertainty. Requirements change has been reported as one of the main factors that cause a project to be challenged [2, 3]. It specifies that it's a challenge to manage these requirements change in software development. Requirement Volatility can be defined as the alteration to requirements after the initial set of requirements has been contracted by the stakeholders of the requirements [2].

Poor requirements collection strategies, deficient experience, government policies and many other factors cause the requirements to be volatile. Jones [4] defines Requirement Volatility as "One of the most chronic problems in software development is the fact that application requirements are almost never stable and fixed". A number of factors have been identified in the literatures that cause requirements to change [2, 5, 6]. The requirements make a software project to be a success both financially and functionally [7]. The quality of requirements specification is significantly affected by these recurrent changes to requirements during development life cycle [7, 8 9]. Project risk assessments must be carried out by keeping requirements volatility in mind [10] as it has been recognized as a factor which has a considerable impact on defect density [11].

Requirement volatility can degrade the performance of a software project therefore efficient concern for requirement volatility is vital to achieve. To identify volatile requirement at the early stage is very necessary. At present all the techniques available for assessing requirement based risk are ad hoc in nature [12]. These techniques are very much subjective in nature and are usually performed in an informal and subjective way by an expert team. Conventionally, all types of volatility is gripped after the design phase and testing phase where time and cost to fix the same turned out to be too high. So, it is suggested to software developer to identify and eliminate the volatility as earlier as possible to overcome the extra costing in handling it.

2. PROPOSED FRAMEWORK FOR STABLE REQUIREMENT SPECIFICATION (SRSF)

Software development is dynamic activity and requirements keep on changing [13]. These changing requirements are not the actual problem; the issue is to incorporate these changes at later stage of the development. Volatility that is unidentified earlier in the development process may have widespread repercussion in the later phases of SDLC. Requirements volatility relentlessly results in notable expansion in size from the initial requirements specification to absolute system deployment. The motivation to control the project size is just not to restrict the scope of requirements expansion as a result of volatility. To tackle these issues, a framework to deal with requirement volatility must be present. Even though requirements volatility is a well-deliberated area, however there is a lack of an efficient framework. The existing research is generally involved in analyzing the impact of volatility on various aspects of software project and its attributes, like project performance or risk, software

maintenance [14], and defect density [12].This paper is tended to concentrate on giving a framework to identify early requirement volatility, predict the level of volatility and analyze the impact of these changes it at the same time.

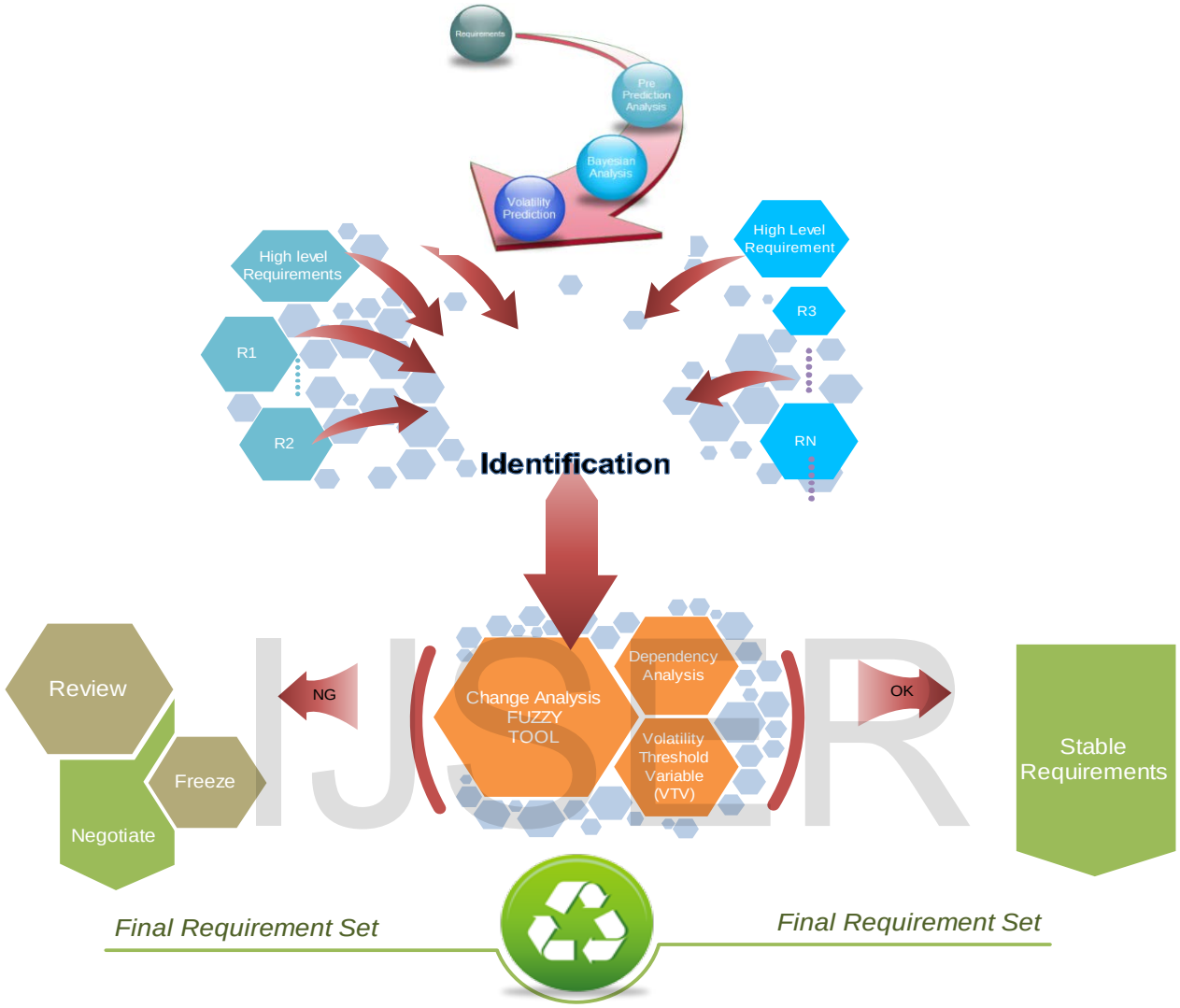

Fig1: A framework for Stable Requirement Specification(SRSF)

Here, a Framework has been proposed for Stable Requirement Specification (Fig 1) consisting three major components: 1) Input component in the form of Initial Requirements 2) Core Process as the combination of Volatility Identification Technique, Dependency Analysis, Change Analysis Tool and Volatility Threshold Variable 3) Output component in the form of Stable Requirement Specification. The proposed framework is basically functional into three categories:

1. Volatility Prediction
2. Volatility Identification
3. Volatility Assessment & Management

Fig (2) shows the workflow of the framework. Initial requirements are provided as Input, the framework predicts level of volatility with the help of Bayesian Analysis. If the predicted value is too high then a review meeting is planned with the stakeholder and requirements are negotiated with respect to given time and budget schedule. Requirements within the range of predicted values of volatility are taken into next stage and Identification of the volatile requirements is performed by implementation of Inspection Techniques [14]. Identified volatile requirements are taken into final stage of the process. Interdependency analysis among the requirements and Change Analysis are performed in this stage. The outcome of this stage is a Volatility Threshold Variable which plays a vital role in fixing up the cost and budget schedule.

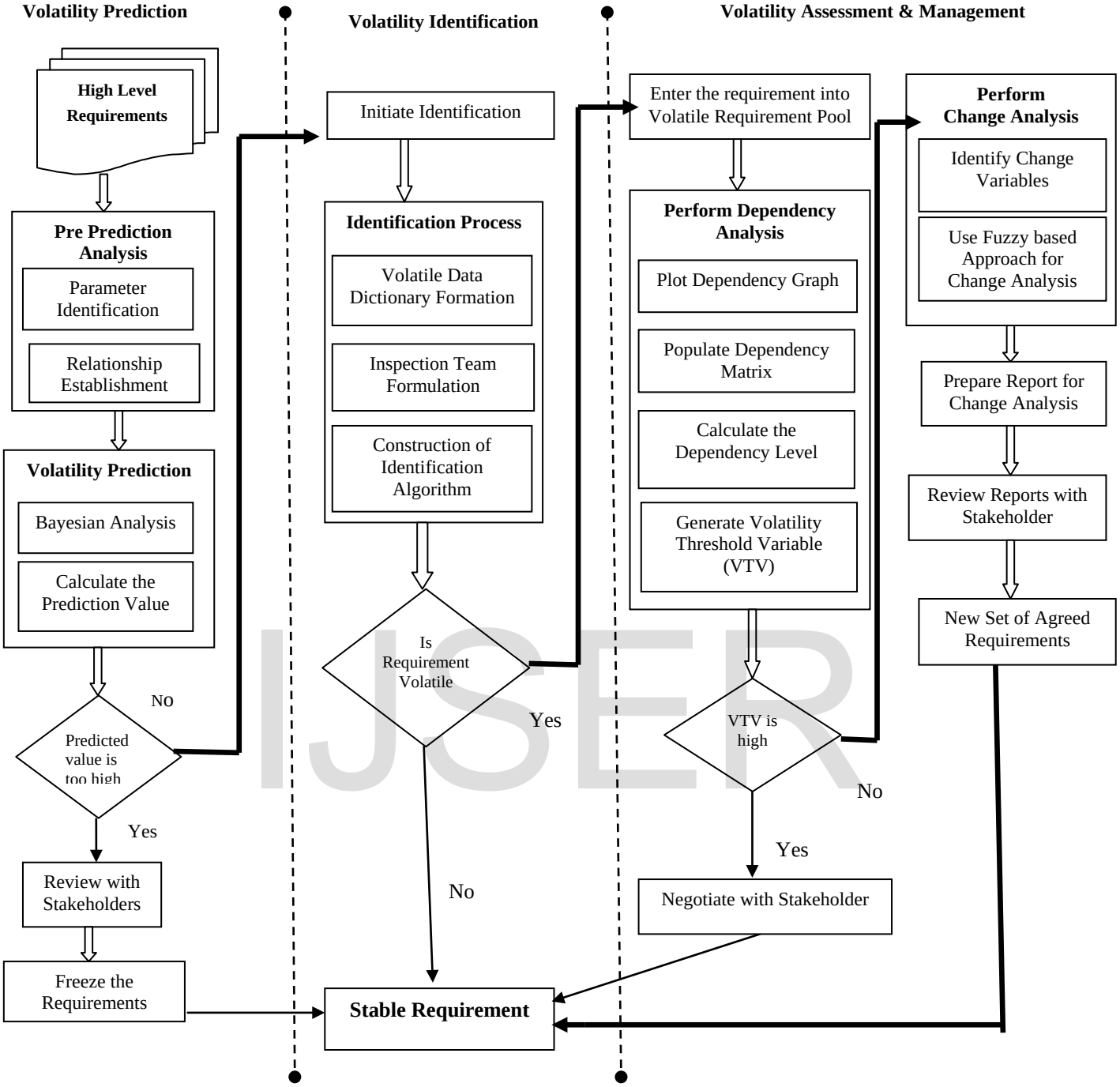

Fig 2: Flow Graph of Proposed Framework

3. VOLATILITY PREDICTION USING BAYESIAN NETWORK

Requirement Volatility is a risk believed to be uncontrollable and outside of a project manager's influence. Although not all volatile requirements can be controlled, they can be managed. Requirement changes that are not critical to achieving a system's objectives can be discarded; development team can create an environment that eliminates the causes for avoidable volatile requirements. Finally, where such methods are not feasible, methods for prediction & quantification of Requirement Volatility must be introduced. The project managers can prepare a more stable process by predicting requirement volatility to curtail project risks. Quantification of volatility can be very effective to handle it as there is a proverb "YOU CANNOT CONTROL WHICH YOU CANNOT MEASURE".

The framework proposes a method that employs Bayesian analysis to predict the volatility level. Bayesian analysis is a statistical method for supporting the decision making process by representing beliefs about the world as probabilities [16, 17]. These probabilities however only represent beliefs that are not definitive, that is some disagreement may exist about the validity and applicability of the results. However, Bayesian analysis gives a reasonable inference about the new data based on the previously fed data. Taking various causes of requirement volatility as input, a Naïve Bayesian has been constructed to predict the level of requirement volatility. A relationship among various volatility causes has been established and accordingly Bayesian Network has been constructed for each category.

5.2 Requirement Volatility Factors

The number of people involved in writing requirements of a system, directly form the factors for requirement volatility as each individual has his\her own perceptions of changing requirements and their specific effects. Each and every requirement volatility factor cannot be evaluated and quantified. So only a standard set of volatility factors which directly affect system development are quantified. These set of requirements is generalized in such a way that they are applicable to a wide variety of software systems. Also the tools using these generalized set of volatile requirements can be applied to a wide variety of programs under development. Additionally, the lessons learned in each program can be retained and used to provide better estimates of requirements volatility in future programs [18].

In this paper, various factors are identified and a relationship among these factors has been established (Fig 3). These relationships provide a basis for the level of requirement volatility prediction in a requirements document.

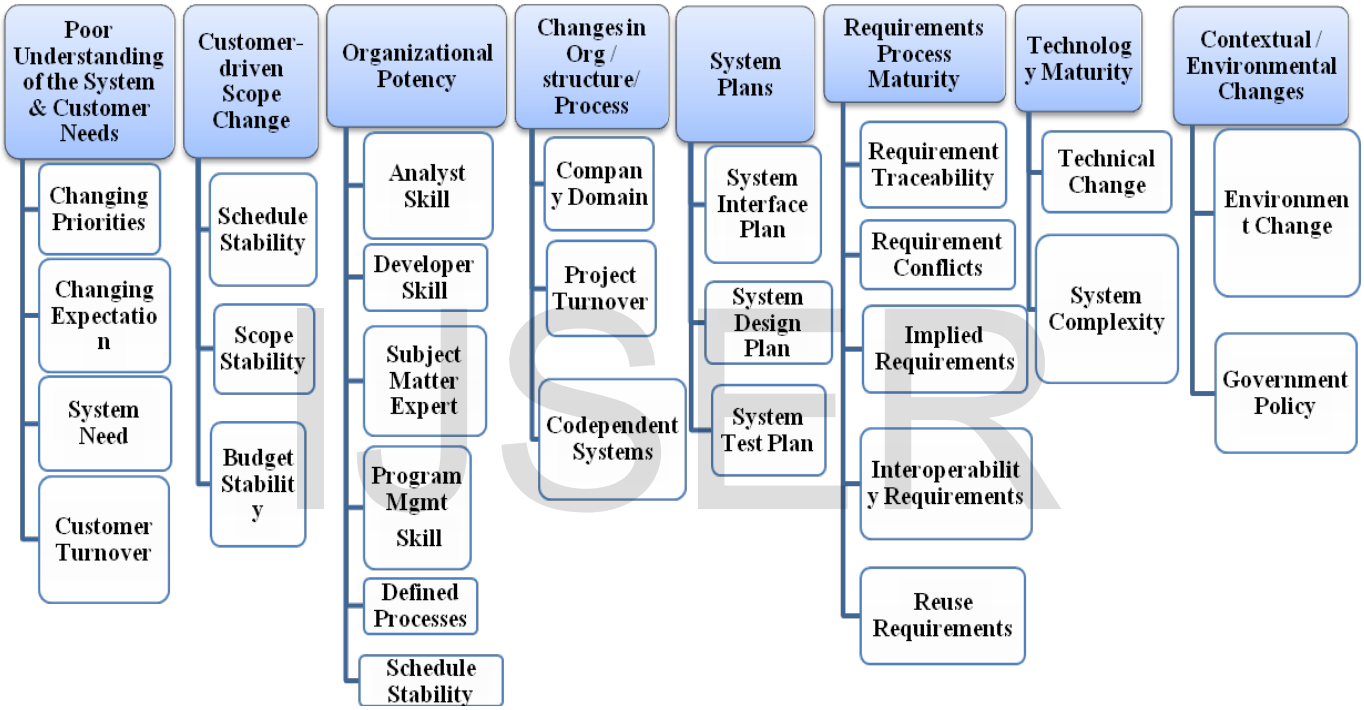

Fig 3: Requirement Volatility Factors

The Requirement Analyst along with his team, records the collective beliefs up to which each requirement volatility factor is present in the requirements document of the system under development. The BN Tool is used here and these collective beliefs are entered as shown in Figure. The proposed model predicts the overall requirement volatility that will affect the system development in later phases.

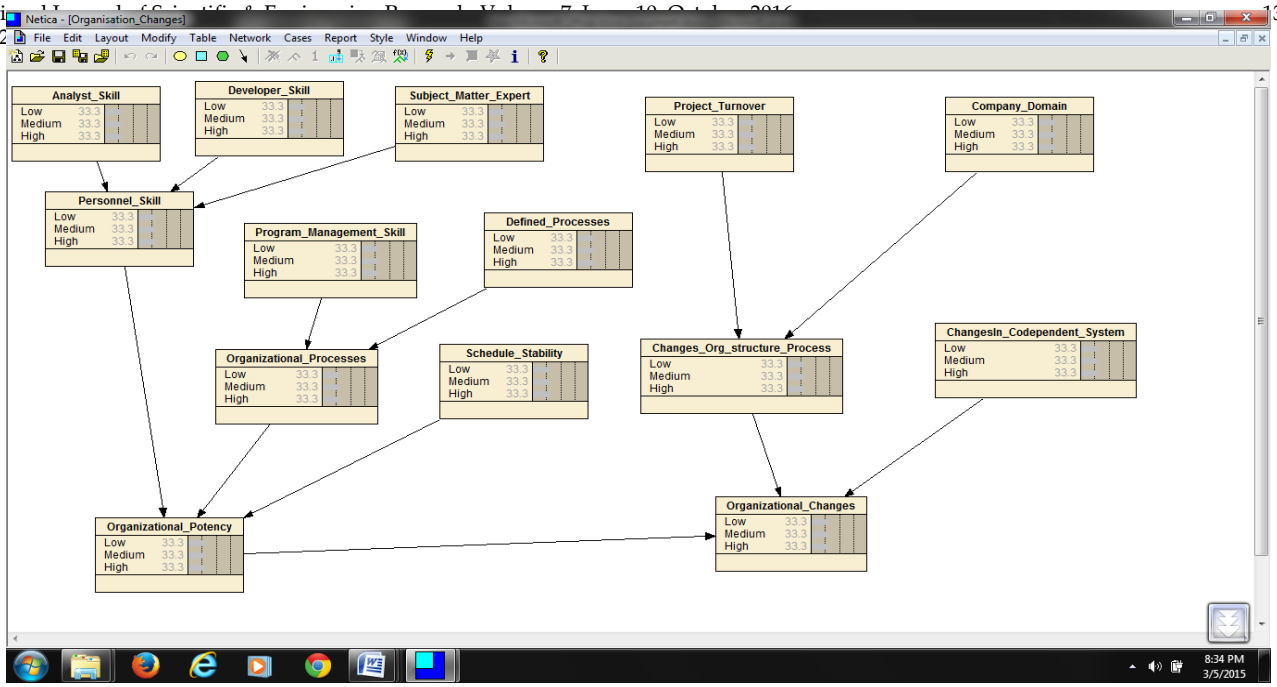

Fig 4: A Bayesian Network Model for Requirement Volatility Prediction

BN networks are developed for all the requirement volatility factors and results are calculated for each category. The results of these BN networks (Fig 4) are then used for calculating the overall probability of occurrence of volatile requirements. Although all the possible requirement volatility factors are used for estimating the prediction class, it can be useful to consider other factors that may also affect whether volatility is deemed acceptable.

For Bayesian Analysis we have used NETICA Tool. Various Requirement Volatility factors have been ranked and the data entry is done in the tool. The NETICA tool returns probability distribution. These predictions are used to tell the level of volatility caused by the standardized factors. The model proposed in this paper, uses a probability distribution ranked between 0 - 10. These results obtained from the model can be combined together with the experience of the requirement analyst team to conclude to a reasonable inference. These serve as a tool for mitigating potential requirements volatility factors affecting system development.

The following case file (Fig 5) was used to input the values of the parent nodes in the Bayesian Network. Only 12 cases were used to test the Bayesian Network.

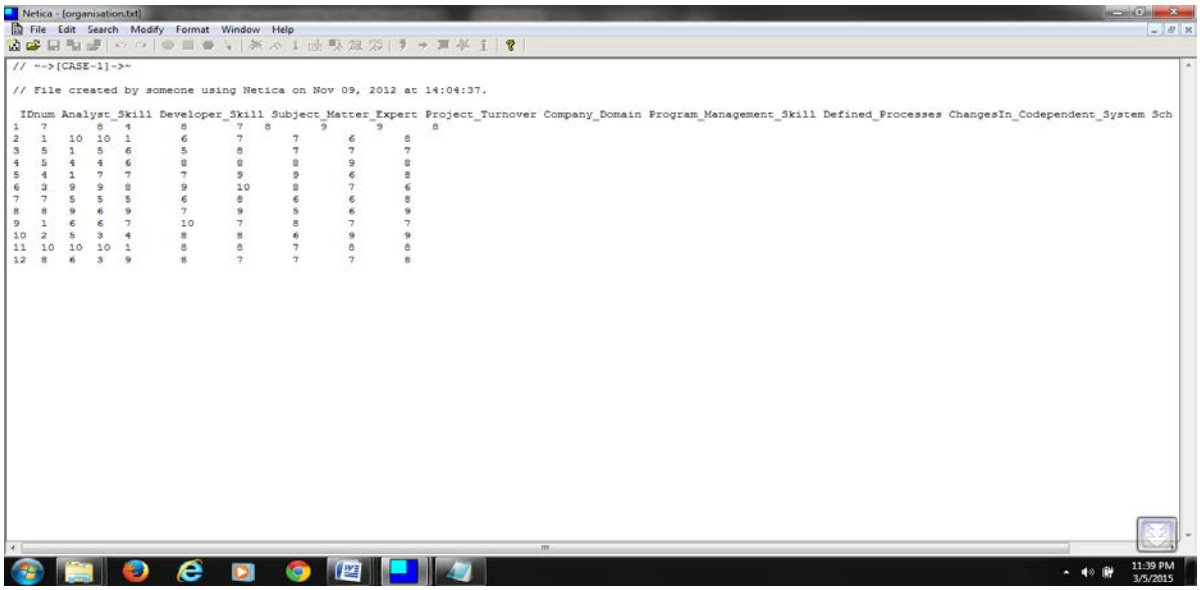

Fig5: Case File for Netica

This paper presented a framework to generate a Stable Requirement Specification which uses the Bayesian Analysis method for prediction of requirements volatility a software project can experience throughout the development life cycle. The Bayesian analysis uses this prediction to provide system stakeholders greater visibility regarding the root causes of volatility. It provides an insight to the various phases of software development that may get affected due to these volatile requirements.

4. CONCLUSION:

Requirement volatility, defined as the alteration in requirements has been acknowledged as a major cause for a software project to experience challenges. This paper proposed a framework to handle the problem of requirement volatility at an early stage of development. The next stages of the research involve development of a multi criteria based fuzzy tool to analyze the impact of changes in a requirement specification**.** It may allow eliminating volatility at early stage so that impacts of requirements volatility during software development life cycle can be minimized.

5. REFERENCES:

1. K. E. Wiegers, “Software requirements”, Microsoft press 1999.
2. G. Stark, A. Skillicorn, and R. Ameele, "An Examination of the Effects of Requirements Changes on Software Releases," CROSSTALK, The Journal of Defence Software Engineering, December 1998.
3. Standish report - CHAOS Summary 2009, http://www1.standishgroup.com/newsroom/chaos_2009.php, assessed on 13th July 2010.
4. C. Jones, Strategies for managing requirements creep, IEEE Computer 29 (1996) 92–94.
5. E. J. Barry, T. Mukhopadhyay, and S. A. Slaughter, "Software Project Duration and Effort: An Empirical Study,"Information Technology and Management, vol. 3, pp. 113-136, 2002.
6. M. Christel and K. Kang, Issues in Requirements Elicitation, Carnegie Mellon University, Pittsburgh September 1992.
7. T. Hammer, L. Huffman, and L. Rosenberg, "Doing Requirements Right the First Time," CROSSTALK, The Journal of Defence Software Engineering, December 1998, pp. 20-25
8. D. Zowghi, and Nurmuliani, "Investigating Requirements Volatility During Software Development: Research in Progress", Proceedings of the 3rd Australian Conference on Requirements Engineering (ACRE98), Geelong, Australia, 1998
9 D. Zowghi, R. Offen, and N. Nurmuliani, "The Impact of Requirements Volatility on Software Development Lifecycle," proceedings of the International Conference on Software, Theory and Practice (ICS2000), Beijing, China, 2000
10. L. Hyatt and L. Rosenberg, "Software Metric for Risk Assessment," proceedings of the 26th Safety and Rescue Symposium, Risk Management and Assessment Session, Beijing, China, 1996.
11. Lane and A. L. M. Cavaye, " Requirements Volatility Enhances Software Development Productivity," proceedings of the 3rd Australian Conference on Requirements Engineering (ACRE 98), Geelong, Australia,1998
12. S. Nayak, R. Khan and R. Beg “Evaluation of Requirement Defects: An Implementation of Identification Technique” ICIET2012, Journal of Procedia Technology
13. Y. Malaiya and J. Denton, "Requirements Volatility and Defect Density," proceedings of the 10th International Symposium on Software Reliability Engineering, Fort Collins, 1998.
14. Bee Bee Chua and June Verner “Examining Requirements Change Rework Effort: A Study” International Journal of Software Engineering & Applications (IJSEA), Vol.1, No.3, July 2010 DOI : 10.5121/ijsea.2010.1304 48
15. Md. Faisal, Md. Rizwan Beg, Halima Sadia “An Efficient Approach for Requirement Volatility Identification”, International Journal of Computer Application, Volume 102, September 2014.
16. D. A. Wooff, M. Goldstein and F. P. A. Coolen, "Bayesian Graphical Models for Software Testing," IEEE Transactions on Software Engineering, vol. 28, no. 5, pp. 510-525, 2002.
17. Armour, F., Catherwood, 8., Beyers c., "A Framework for Managing Requirements Volatility using Function Points as Currency." *International Function Point Users Group Annual Conference*, San Diego, CA, September 2000.
18. A. K. Delic, F. Mazzanti and L. Strigini, "Formalizing a Software Safety Case via Belief Networks," Center for Software Reliability, City University, London, U.K., Technical report 1995.