Software Risk Factors: A Survey and Software Risk Mitigation Intelligent Decision Network Using Rule Based Technique

* Muhammad Asif, Jamil Ahmed, and Abdul Hannan

Abstract—**Software industry has been rapidly growing from the last couple of decades. Due to this growth and advancement many issues are going to be occurred. There are number of factors that affect the whole software development process. This research has focused on the most important risk factors that affect the overall software project and risk mitigation reviewed from the literature and my own experience. The main aim of this research is to collect the most important risk factors and risk mitigation for the development of intelligent network using an Artificial Intelligence technique known as Rule Based System. This is helpful for the project managers to make decisions. For the accomplishment of this task Firstly an exploratory survey has been conducted to prioritize the twenty requirements and fifty software risk mitigation factors. A sample of 200 respondents is selected out of which 100 are the university students and 100 are the IT professionals of a well reputed software house. Secondly a Rule Based technique is used to generate the risk mitigation intelligent network. Rules have been stored in the form of risk mitigation in relationship with risk factors. The system follows five steps of execution. First of all risk factors has been prioritized based upon the associations of risk factors and risk mitigation. Secondly, it creates new relationships between them. Thirdly, enter these relationships as rules into the Knowledge Base or Rule Base. At the fourth stage, an RBS engine executes these rules and finally an intelligent risk mitigation network is generated.**

Index Terms—**Software Risk Factors, Software Risk Mitigation, Intelligent Decision Network, Artificial Intelligence in Software Engineering, Rule Based System.**

I. INTRODUCTION

SOFTWARE project development is a very complex and critical job in software engineering [1]. Software development lifecycle (SDLC) plays a vital role in any software development activity. It includes different phases such as analysis, design, detail design, implementation and maintenance. The verification and validation is associated with every phase of SDLC. Using Artificial Intelligence in software engineering provides new ways of software development.

There are certain factors that affect the project scope, planning objectives, scheduling, budget and implementation [2]. These factors are known as risks. Risks are basically negative factors that create hurdles during the execution of software project or risks are even those events that can threaten the software success. Risks can be handled or managed but it is necessary to analyze and identify the hypothetical risks. There are possible risk factors that are certain or uncertain and risk mitigation from the literature [1] [2] [3]. Certain risks are those risks that are known and uncertain risks that are unknown [1]. Risks may be dependent or independent in nature. Dependent risks are those that are occurred with the presence of some other risks. And independent risks are those that are generated without the dependency of some other risks [6]. Software project failure causes might lead the project to a closed end of a street. Internal risks are those that come from inside the organization and threaten it. On the other hand external risks always come externally i.e. risks that are hard to handle and control [4].

This research work is based on two phases. Firstly, an exploratory research has been conducted to figure out the risk factors and assigned ranks on the basis of respondent's feedback. Secondly, an intelligent risk mitigation network has been established through an Artificial Intelligence technique generally known as Rule Based System (RBS). It is very difficult to identify and control all of the risks during software development but this research work can make an Artificial Intelligence system which stores the risk factors and risk mitigation in the form of relationships in the Knowledge Base to generate a network to facilitate the project managers for decision making. It reduces the failure factors of projects.

II. SOFTWARE RISK FACTORS

In this section following are the important risk factors collected from the literature review.

A. *Unrealistic deadlines*

Software project may fail when deadlines are not properly set. It includes time and schedule. Project initialization, finishing date and time must be realistic otherwise it may cause a great deal of harm [1] [3] [4].

B. *Improper budget*

Cost estimation of a project is very crucial in terms of project success and failure. Low cost with high expectations of large projects may cause project failure. An organization cannot bear the expenses of a project [1] [3] [5].

Manuscript received December 08, 2013.

*Corresponding Author: Mr. Muhammad Asif (member IAENG, no: 133742) is a PhD Scholar of Abasyn University, Islamabad, Pakistan. (Phone: +92518438320; fax: +92518438325; e-mail: masifkhan_2055 @yahoo.com)

Dr. Jamil Ahmed is the Vice Chancellor of Abasyn University, Islamabad, Pakistan. (email: jamil.ahmad@abasyn.edu.pk).

Mr. Abdul Hannan is a PhD Scholar of Abasyn University, Islamabad, Pakistan. (email: abdul.hannan@abasyn.edu.pk).

C. Lack of resources

Software and hardware resources are not quite up to the mark. Lack of resources in terms of man power is also a critical risk factor of software failure [1] [3] [4].

D. Personnel hiring

Extensive hiring and firing in a software team may lead a software project to a critical stage. Also the staff is not properly organized to proper tasks [6] [7].

E. Understanding problems of customers

Most of the occasion customers are not technical in terms of software terminologies and don't understand the developer's point of view [4] [5].

F. Understanding problem of developers

Sometimes developers perceive different meanings of provided information by the clients [4] [5].

G. Size of the project

Most of the time managers could not justify the project because of its relatively huge size. Large projects with minimum resources may affect the overall success rate [3] [6].

H. Inappropriate design

Software designers have a major role in the success or failure of the project. If project design is not appropriate then it may be disastrous [2] [4].

I. Inappropriate technology

Information technology and computer science both are dynamic in nature. Technology is progressing by leaps and bounds. Selecting an inadequate or obsolete technology for the software development is definitely going towards a project failure [2] [4].

J. Implementation

While implementation there is a need of skillful person. Other real-time factors may also create disturbance during the development such as lack of software or hardware resources in the market [3].

K. Market demand obsolete

Sometimes market demand obsoletes while project is still in progress [1].

L. Improper planning

During analysis, an improper definition of goals and objectives may result project failure. A good plan leads to a success and a bad plan can cause failure [1] [6] [7]. There is a great chance of unclear requirements in the planning and analysis phase.

M. Improper marketing techniques

Most of the software projects have been completed successfully but due to non-technical marketing team, it may affect the whole project badly [4] [6].

N. Higher management decisions

Higher management always plays a vital role in the project success or failure. Strong and positive decisions lead towards the successful achievement of the goals but bad decisions always leads to a zero level [1] [7].

O. Improper scope definition

Project scope should always be defined precisely. Little diversion from a scope ultimately affects the goals [4] [5].

P. Lack of experience of project manager

Project manager is the leader and the most responsible person. Inexperience manager is very harmful and dangerous for the project and the team [4] [7].

Q. Cultural diversity

World is now a global village, so the software teams are full of diverse members in terms of culture, gender, social, language etc. There should be a team with full of coordination and collaboration [3] [4].

R. Lack of motivation

Team members should be motivated for the achievement of the goals otherwise objectives cannot be met according to the plan [6].

S. Government factors

Legislative factors are very critical and continuous. Government and its legislation have been changing frequently. Their rules and regulations affect the software industry [1] [3].

T. Improper feasibility Study

Feasibility report is a backbone of any software project success. Feasibility study aims at SWOT (Strength, Weakness, Opportunity and Threats) analysis [6]. Wrong analysis will lead towards a bad feasibility report. Feasibility refers to five areas TELOS (Technical, Economic, Legal, Operational and Scheduling).

III. SOFTWARE RISK MITIGATION

This section contains 50 risk mitigation factors reviewed from the literature [5] [7] [8] and my own experience against the risk factors in section II.

1) Clear Idea of the requirements: All the requirements should be well defined and unambiguous.

2) Proper Feasibility Report making: A thorough and complete feasibility study is required which then leads to a project success and allows decision makers to strive towards a successful project.

3) Requirements Specification: A complete Software Requirements Specification (SRS) is needed. It is a complete description of the project. It includes functional and non-functional requirements.

4) IT Consultants: An IT consultant is the one who works with all the stake holders. They are also well aware of the latest technologies and implementations.

5) Proper Communication Channel: Communication between employees and stakeholders should be held regularly. e.g. daily or weekly meetings.

6) Retaining and preservation of Good Employees: To keep the experienced and good employees in contact. They are real assets of any organization.

7) Bonuses: A reward in terms of special salary packages and extra pays etc. It helps to boost the motivation of employees.

8) Attractive packages: Attractive salary at which the team members work more effectively.

9) Developers Faithfulness: Developers must be very motivated and very clear direction towards completion of the work and have full confidence on the project manager.

10) Proper Team Structure: Team should be properly structured and managed. Centralized team structure should be adopted to mitigate risks.

11) Proper backup plans: Backup plans should be given preferences because if there is no backup plan then there can be a complete loss.

12) Define Goals and Objectives: Goals and objectives should be focus in a project plan otherwise it is difficult to complete a project successfully. Goals are the desired results which a project manager wants to achieve in a finite time.

13) Ensure Communications and Milestones: Ensure to achieve deliverables and milestones through proper collaboration and communication within a given time period.

14) Leadership: Leadership is very important for the achievement of any goal successfully. Leader should be very creative and innovative.

15) Past Experience: Past experience may lead to a success. Experiences of previous projects are helpful for the successful completion of a project.

16) Proper use of methodologies and Software process models: All the models and methods must be appropriate and up to the mark of the current technologies. e.g.; software process models.

17) Work Unit Culture: The environment and working people are very crucial to any organization. It focuses on team work.

18) On the job and off the job training: On-the job training should be given to employees in the working hours and off-the job training should be away from the working environment.

19) Respect and Honour of Employees: Respect and Honour should be given to all the employees. Dignity and appreciation of the employees should be taken care of.

20) Employee Attitude: Employee attitude towards work should be positive and constructive.

21) Employee Skill: Talented and Skilled employee is the main asset of the organization.

22) Employee Awareness: Employee should know the situations, policies and documents. Employee Awareness training programs must be provided.

23) Continuous Review: Continuous review process is very important in every phase of software development life cycle. Formal Technical reviews should be conducted in regular intervals.

24) Project Scheduling: Project scheduling includes milestones and deliverables on proper dates. e.g.; Activity Network Diagram and Bar Chart Diagram.

25) Prototyping: Make an initial version of the software that meets the user’s requirements.

26) User Involvement: User must be involved in the development process of software regularly.

27) Use Statistical Methods: Always use mathematical models and methods for the solution of issues.

28) Choice of technology after thorough research of available tools and technologies: Tools and technologies should be used properly and flexible with future trends and techniques.

29) Human Resource Role: Human resources are the set of personalities who are responsible for making an organization’s workforce. This work force should be creative and technically skillful people.

30) Proper Testing Techniques: Proper software testing will lead to a successful and mature product. For example Black box testing, White Box testing, Integration testing, Unit Testing etc.

31) Proper Sales Marketing Team: An experienced and qualified marketing team is required.

32) Identification of Success Criteria: The goals and aims that must be achieved.

33) Policy Setting and Enforcement: Clear and fair policies should be implemented and enforce people to follow them.

34) Scrub able requirements: Requirements that are specific and according to the needs of the customers.

35) Top Management Commitment: Higher officials are the decision makers of any organization. They should be very flexible and innovative.

36) Facilitated Application Specification Technique: A technique for requirements elicitation for software development and mediator between client and developer.

37) Centralization: There should be a centralized control mechanism for software project execution.

38) Intuitive and Creative: Innovative thinking and creative ideas is the key to successful achievement of goals.

39) Positive behaviour and problem solving skills: Positive behaviour is a key to success of any project. If direction is focused and attitude is positive then everything goes in right direction without any ambiguities.

40) Security Checklist and Authentication Process: A complete list of checks for the maintenance of security and the verification of product.

41) Set Key Performance Indicator: A key performance indicator (KPIs) is a measure of efficiency and performance.

42) Stress testing: It can determine the stability and efficiency of a system. System is under heavy load during testing to uncover errors and measure efficiency.

43) Regular Updates: Project updates and new technologies should be properly managed with regularities.

44) Assess Past Communications: Information collection or Requirements gathering is crucial in software development phases. Communication should be very clear and strong in order to grasp the ideas. Past problems and their solutions should be taken care of. Assessment of past communication is very critical.

45) Contingency Plan: A plan for possible future circumstances and events.

46) Trouble Shooting: Tracing and correction of errors and faults in a software system.

47) Reusability: Reusability refers to the possibility that a piece of component can be used again. This includes new additional functionalities and modifications. It reduces the bugs and implementation time.

48) Project Tracking and Control: Analyzing project development from the beginning to the end of the project.

49) Impact Assessment: To assess the impact of risks on the software that has to be built.

50) Consistent Commitment: Consistency with commitment is very important towards success. Software Elicitation is a requirement discovery stage.

IV. CASE STUDY: A SURVEY

An exploratory survey has been conducted to prioritize these twenty requirements. This survey has been divided into two parts.

1) Sample of 100 students (Male/Female) of a university has been selected. These 100 students are randomly selected from MIT, MCS, BCS, and BIT sections.

2) Sample of 100 IT professionals from an IT department of a well reputed software house has been randomly selected. These IT professionals include IT managers, Team leaders, Senior Software Engineers, Software Engineers, Designers, Software Analysts and Quality Assurance Engineers.

A survey has been conducted from a sample of 200 respondents. On the basis of their feedback ratings have been assigned to certain software risk factors in terms of percentages accordingly. Open-Ended questions have been asked about these 20 software risk factors from the respondents. This is also an experience survey in which opinions of experts have been collected. These risk factors have been categorized into different rankings based upon the number of occurrences in percentage. These are prioritized from 1 to 20 on the basis of respondent's feedback. Table I. given below shows the ratings of these factors.

TABLE I. SOFTWARE RISK FACTORS RATING

Rating	Software Risk factor	Frequency in Percentage
1	Improper feasibility report	90%
2	Higher management decisions	88%
3	Understanding problems of customers	85%
4	Understanding problem of developers	82%
5	Improper planning	80%
6	Improper scope definition	78%
7	Lack of experience of project manager	76%
8	Government factors	75%
9	Implementation	72%
10	Cultural diversity	70%
11	Lack of motivation	68%
12	Personnel hiring	67%
13	Unrealistic deadlines	60%
14	Inappropriate design	55%
15	Improper budget	52%
16	Inappropriate technology	49%
17	Lack of resources	44%
18	Size of the project	40%
19	Improper marketing techniques	35%
20	Market demand obsolete	30%

V. SOFTWARE RISK MITIGATION INTELLIGENT DECISION NETWORK THROUGH RULE BASE

Rule Based System (RBS) is an Artificial Intelligence technique. An RBS has been used to provide a solution to real world problem by using expert knowledge. Rules are the better representation of expert knowledge [13]. A Rule Based System has three components, A Rule Based Engine (An Inference Engine), A Knowledge Base (KB) and Working Memory (WM) [12]. Software risk factors and risk mitigation are stored in the form of rules in the Knowledge Base. Domain Engineers or Domain Experts stores data in the form of relationships between software risk factors and risk mitigation. Rules are basically conjunction of conditions of risk mitigations. An RBS takes risk factors as inputs and generates an intelligent decision network of risk mitigation nodes by an inference engine. A framework of RBS is given in Fig. 1.

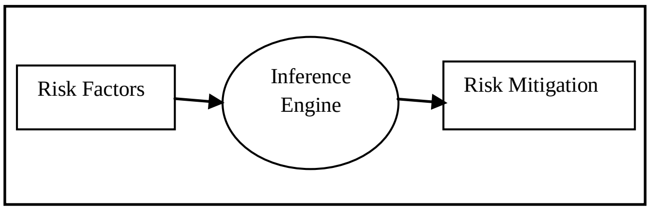

Fig. 1: A Generic RBS Framework

Initially 20 risk factors and 50 risk mitigation factors have been worked out. This system will be effective as knowledge base increases. An RBS engine accepts software risk factors as inputs. It searches already existing rules in the form of relationships between Risk factors and risk mitigation from the knowledge base. If Rule(s) found then an RBS inference engine apply the selected rule(s) and generates a network of nodes that represents software risk mitigation against the risk factors. On the other hand if rule(s) not found then this engine will follow the five steps of execution. a) First of all it prioritizes the risk factors on the basis of already stored knowledge in the form of relationships between risk factors and risk mitigation. b) Create new relationships of the input factors with risk mitigation. These relationships will be created through probability of occurrence of risks. c) Add these relationships in the form of rules into the Knowledge Base or Rule Base. d) Apply these newly created rules from the KB. e) Generate an intelligent risk mitigation network. See Fig.3.

Probability of occurrence of risks is based on two types of risks i.e. Dependent Risks and Independent Risks [6]. Also the relationships between risk factors and risk mitigation are created using probability. Probability of independent risks have been shown mathematically as P(Risk4 and Risk5) = P(Risk4∩Risk5) = P(Risk4)P(Risk5). Probability of dependent risks have been shown mathematically as P(Risk2|Risk1)P(Risk2∩Risk1)/P(Risk1). Risk (Risk Factors and Risk Mitigation) taxonomy of dependent and independent risks is shown in Fig.2.

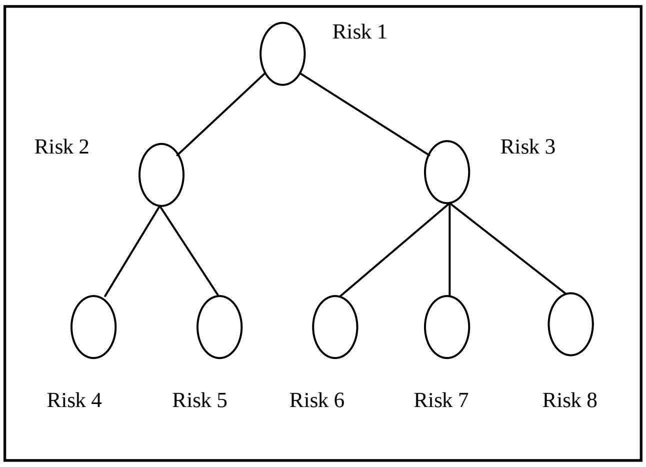

Fig. 2: Risk Taxonomy

There are total no. of 20 rules of software risk mitigation [9] [10] against certain software risk factors. This decision network would be helpful in making decisions as decision support system by Asif, M. [11].

Fig. 3 represents an Intelligent Network Engine.

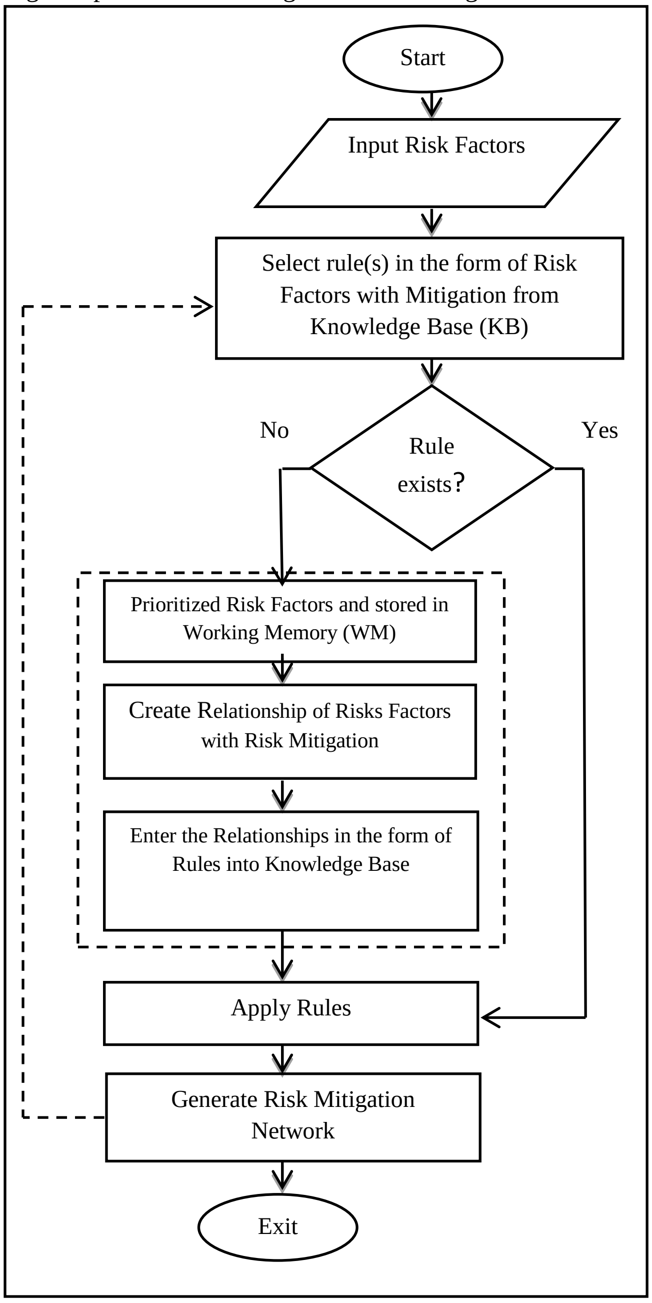
Start
Input Risk Factors
Select rule(s) in the form of Risk Factors with Mitigation from Knowledge Base (KB)
Rule exists?
No
Yes
Prioritized Risk Factors and stored in Working Memory (WM)
Create Relationship of Risks Factors with Risk Mitigation
Enter the Relationships in the form of Rules into Knowledge Base
Apply Rules
Generate Risk Mitigation Network
Exit

Fig.3. An Intelligent Decision Network Engine

Rule 1: Improper feasibility report {Clear Idea of the requirements, Proper Feasibility Report making, Requirements Specification, Scrub able requirements, Choice of technology after thorough research of available tools and technologies, Define Goals and Objectives}

Rule 2: Higher management decisions {IT Consultants, Work Unit Culture, Proper Communication Channel, Top Management Commitment, FAST}

Rule 3: Understanding problems of customers {Proper Communication Channel, Work Unit Culture, Proper Sales Marketing Team, Employee Attitude, Employee Skill, Employee Awareness}

Rule 4: Understanding problem of developers {Retaining and preservation of Good Employees, Bonuses, Attractive packages, Developers Faithfulness, Proper Team Structure, and Centralization}

Rule 5: Improper planning {Proper Team Structure, Proper backup plans, Requirements Specification, Clear Idea of the requirements, Define Goals and Objectives}

Rule 6: Improper scope definition {Goals and Objectives, Ensure Communications and Milestones, Clear Idea of the requirements, Leadership, Continuous Review and Identification of Success Criteria}

Rule 7: Lack of experience of project manager {Leadership, Past Experience, IT Consultants, Proper Team Structure, Proper backup plans, Continuous Review, Proper Communication Channel, Intuitive and Creative}

Rule 8: Government factors {Policy Setting and Enforcement, Positive behaviour and problem solving skills, Security Checklist and Authentication Process}

Rule 9: Implementation {Clear Idea of the requirements, Proper use of methodologies and Software process models, Proper Feasibility Report making, Requirements Specification, Developers Faithfulness, Top Management Commitment, Set KPIs, Ensure Communications and Milestones}

Rule 10: Cultural diversity {Work Unit Culture, Leadership, Proper Team Structure, Stress testing}

Rule 11: Lack of motivation {On the job and off the job training, Retaining of Good Employees, Bonuses, Attractive packages, Respect/ Honour of Employees, Positive behaviour, problem solving skills}

Rule 12: Personnel hiring {Retaining and preservation of Good Employees, Employee Attitude, Employee Skill, Employee Awareness, Work Unit Culture, Human Resource Role, Proper Team Structure}

Rule 13: Unrealistic deadlines {Continuous Review, Project Scheduling, Regular Updates, Proper Feasibility Report making, Proper use of methodologies and Software process models, Assess Past Communications, Prototyping, Contingency Plan}

Rule 14: Inappropriate design {Prototyping, User Involvement, Use Statistical Methods, Employee Skill, and Employee Awareness}

Rule 15: Improper budget {IT Consultants, Choice of technology after thorough research of available tools and technologies, Proper Feasibility Report making, Requirements Specification}

Rule 16: Inappropriate technology {Choice of technology after thorough research of available tools and technologies, Trouble Shooting}

Rule 17: Lack of resources {Retaining and preservation of Good Employees, HR Role, and Reusability}

Rule 18: Size of the project {Testing Techniques, Clear requirements, Goals and Objectives, Ensure Communications/ Milestones, Choice of technology, Project Tracking and Control, Impact Assessment}

Rule 19: Improper marketing techniques {Proper Sales Marketing Team, on the job and off the job training, Consistent Commitment, User Involvement, Use Statistical Methods}

Rule 20: Market demand obsolete {Identification of Success Criteria, policy setting and Enforcement, Impact Assessment, Regular Updates}

Fig. 4 has shown software risk mitigation intelligent decision network and nodes are numbered according to section III.

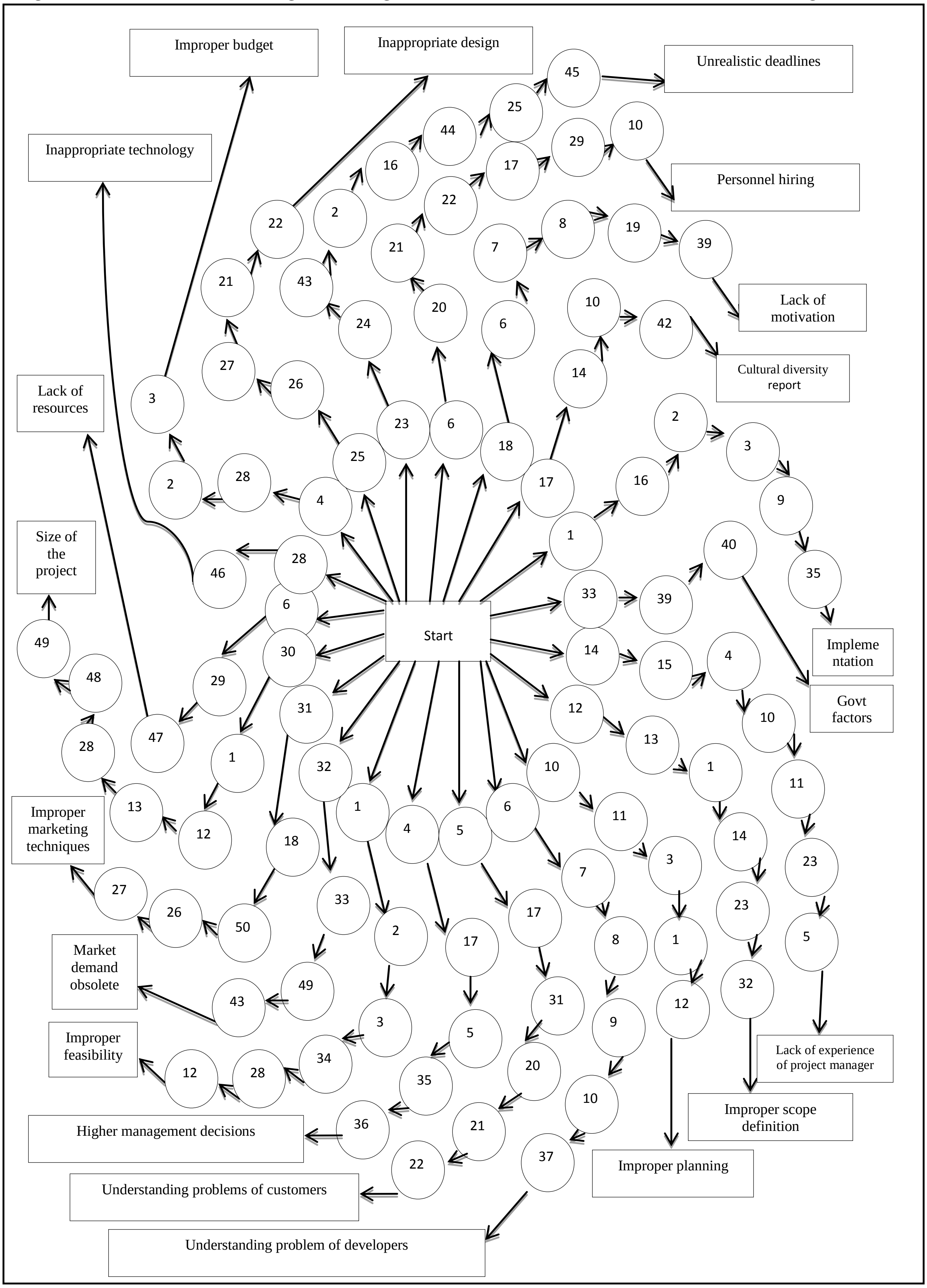

Fig.4. Software Risk Mitigation Intelligent Decision Network

The decision network in fig.4 has been generated through nodes of software risk mitigation as discussed in section III. These connected nodes provide paths to risk factors as a consequence of risk mitigations. Numbers have been assigned to nodes of a network for mitigation of risks. The rectangles in the diagram describe the risk factors specified in section II. The associations between risk factors and risk mitigation have been generated after the execution of rules stored in the knowledge base. E.g. Rule No. 11 Lack of Motivation of software development team members is a risk factor. When there is no motivation of employees from the project manager then there is a great chance of bad work environment. An RBS engine executes this rule and makes decision in conjunction of giving on-the job and off -the job training to employees, Good and skill full employees should be taken care of, Bonuses should be given to the employees, Attractive and smart salary packages should be announced, Respect and Honour of Employees should be maintained, Positive behaviour should be shown to the good employees and manager must have problem solving skills to handle hurdles. And the whole decision network is maintained in this manner.

VI. DISCUSSION AND CONCLUSION

This research work has been fruitful and result oriented. Integration of Artificial Intelligence in Software Engineering has introduced new dimensions for research and development. Senior management of the organization has acknowledged these factors as critical and prominent during the software development stages. Ratings on the basis of software risk factors have been assigned and 50 software risk mitigation factors have also been analyzed. On the basis of these software risks mitigation decision network has been developed using an Artificial Intelligence technique known as Rule Based System. This decision network is the combination of nodes of risk factors in relation with risk mitigation. This network will be helpful to mitigate risk factors and classify the risk factors. First 90% Improper feasibility report, Second 88% Higher management decisions, Third 85% Communication problems with customers, Fourth 82% Understanding problem of developers, Fifth 80% Improper planning, Sixth 78% Improper scope definition, Seventh 76% Lack of experience of project manager, Eighth 75% Government factors, Ninth 72% Implementation, Tenth 70% Cultural diversity, Eleventh 68% Lack of motivation, Twelfth 67% Personnel hiring, Thirteenth 60% Unrealistic deadlines, Fourteenth 55% Inappropriate design, Fifteenth 52% Improper budget, Sixteenth 49% Inappropriate technology, Seventeenth 44% Lack of resources, Eighteenth 40% Size of the project, Nineteenth 35% Improper marketing techniques and Twentieth 30% Market demand obsolete. Causes of software failure will definitely be reduced when the above factors are in the mind of project managers.

ACKNOWLEDGMENTS

The support and encouragement of Abasyn University, Islamabad, Pakistan, Dr. Jamil Ahmed Vice Chancellor Abasyn University and Mr. Abdul Hannan a Ph.D. Scholar is enthusiastically acknowledged. I also acknowledge the encouragement and firm support of my parents (Mr. & Mrs. Sher Afzal) during this research work.

REFERENCES

[1] Arnuphaptrairong, T. “Top Ten Lists of Software Project Risks : Evidence from the Literature Survey”, Proceedings of the International MultiConference of Engineers and Computer Scientists 2011 Vol 1, IMECS 2011, March 16-18, 2011, Hong Kong.

[2] Hoodat, H. and H Rashidi. “Classification and analysis of Risk in Software Engineering”, World Academy of Science , engineering and Technology 56 2009.

[3] Khan, Q. , S. Ghayyur. “Software Risks and Mitigation in Global Software Development”, Journal of Theoretical and Applied Information Technology 2005-2010 Jatit & LLS.

[4] Pressman, R.S. “Software Engineering A Practitioner’s Approach”, Seventh Edition, McGraw Hill Higher Education. ISBN 978–0–07–337597–7. MHID 0–07–337597–7, 2010.

[5] Sakrthidaran, R.T. “How Can An Acquirer Mitigate Risks In Software Engineering Projects”, The open Software Engineering Journal, 201, 4, 64-69.

[6] Uzzafer, M. “A Risk Classification Scheme for Software Projects”, International Journal of Software Engineering and its Applications Vol. 7, No. 1, January, 2013.

[7] Westfall, L. “Software Risk Management”, The Westfall Team. 2001.

[8] Shahzad, B., Yousef, A. and Abdullah, “A. Trivial model for mitigation of risks in software development life cycle”, International Journal of the Physical Sciences Vol. 6(8), pp. 2072-2082, 18 April, 2011.

[9] Hammer, Michael and Champy, James. “Re-engineering the Corporation”, A Manifesto for Business Revolution. New York, NY: HarperCollins Publishers, Inc 2003.

[10] Luftman, Jerry N. “Managing the Information Technology Resource: Leadership in the Information Age”, Upper Saddle River, NJ: Pearson Education, Inc 2004.

[11] Asif, M., Sawar, J. and Abdullah, U, “Design of Decision Support System in Electronic Medical Record Using Structured Query Language”, DOI:10.7763/IPEDR.V63.3 2013.

[12] “Handbook of Measuring System Design”, edited by Peter H. Sydenham and Richard Thorn. 2005 John Wiley &Sons, Ltd. ISBN: 0-470-02143-8.p-910-912

[13] Frederick Hayes-Roth. “RULE-BASED SYSTEMS”, Communications of the ACM, September 1985 Volume 28 Number 9 ACM 000l-0782/85/0900-0921.