Title: A Knowledge-based System to Assist Software Development
Risk Management

Course:
Year: 1990
Author(s): B. Johnson

Report No: P90031

Abstract: This report presents a Knowledge Sharing System (KSS) approach in developing a risk management tool for a company. The purpose of KSS is to promote the automated collection of the firm's historical risk factors and to assist the risk analysis process during program performance.

A KNOWLEDGE-BASED SYSTEM TO ASSIST SOFTWARE DEVELOPMENT RISK MANAGEMENT

B. Johnston

EMP - P9031

A Knowledge-based System to Assist Software Development Risk Management

Brian Johnston, Sequent Computer Systems

To promote the automated collection of Sequent's historical risk factors and to assist the risk analysis process during program performance

A report prepared in partial fulfillment of the requirements for EGMT 510 E, Knowledge Engineering and Management, Winter 1991, under the guidance of Dr. Kiyoshi Niwa.

1.0 System Purpose

1.1 Background

The Software development process at Sequent is culturally ingrained, but lacking definition. Several parallel efforts are underway to define processes. Sequent management is developing a passion for process and a desire to institute continuous improvement mechanisms.

1.1.1 Sequent has implemented a weak matrix organization.

The weak matrix system has arisen due to the entrepreneurial start-up nature of the company. Early technologists (the creative genius responsible for the first product) just did basic R&D and pushed product out the door by shear determination. Very little structure or development life-cycle was present. After several generations of HW/SW product, a more formal mechanisms of program management and development life-cycle was instituted. But the early technologists became the line managers and program management performed a coordination role. Currently, the matrix organization has the following characteristics:

- Line (technical) management staffs project managers and engineers
- Line management holds the budget
- Program/release management coordinates and reviews project performance and manages the release of several products to a given target date.

As a consequence of the weak matrix, the coordination of product development is sporadically successful, and individual contributors tend to listen to the line mangers since their careers, reviews and raises are the responsibility of line management.

1.1.2 No risk analysis mechanism

There exist a token risk analysis section in the original project planning document. This is not a living document. No status or tracking of risk causes and workarounds is performed. Program management believes this is part of their function, but no mechanism exist. As a result, cost overruns, schedule slips, feature deletion and integration problems are not uncommon. No cohesive history kept of project post-mortums and performance against plan. In spite of 30 years history, software development is still an ill-structured problem domain.

1.2 Objectives

The ill-structured problem domain of software development suggests a Knowledge Sharing System (KSS) approach in developing a risk management tool. The objectives of the proposed system include:

- Provide for capture of Sequent SW risk causes and risk solutions from managers who lived through it.
- Develop KSS to allow managers to populate knowledge-base.
- Interpret top ten Sequent historical risks and proposed solutions.
- Allow managers to query for risk and risk mitigation techniques.
- Allow immediate risk update.

1.3 Expected Benefits

The proposed KSS system would aid the process of SW risk analysis by both the program management staff and the project leaders. By developing the top ten Sequent risk factors and the resultant risk mitigation actions, management would be enlightened and process a powerful tool to avoid historical pitfalls.

Additional benefits include:

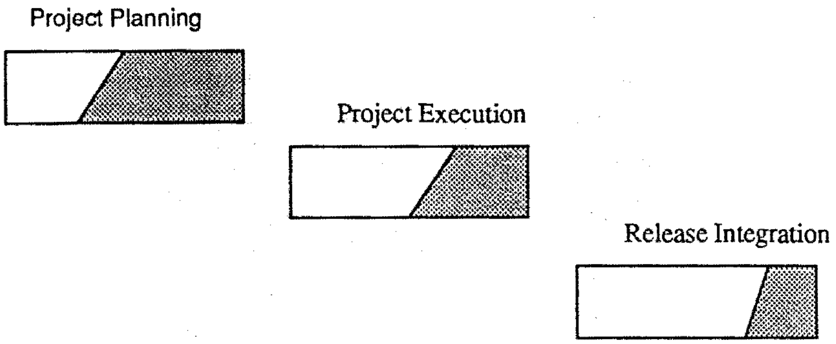
- Greater predictability in project performance, both in cost and schedule
- Automate the maintenance of a project top ten risk items
- A tool for training new program/project management personnel
- Intended for use by program/project management to perform "what-if" scenarios during project planning
- Utilized during project performance to check risk analysis

2.0 System Outline

2.1 System functions

The task of program/project management are illustrated in Figure1.

FIGURE 1 Program management tasks, with risk management areas shaded



2.1.1 System objectives

The functions provided by the system will include:

1. Repository for past Sequent risk factors and risks
2. Correlation of risk factors by forward inference
3. Selection and maintenance of Sequent's top ten risk factors
4. Risk mitigation guidance tied to known or inferred risk factors
5. Browse mode for management to examine risk and risk mitigation actions

2.1.2 System input/output

The user will be able to input the following:

- Request for top ten Sequent SW risk factors and mitigation Stategies
- Choose risk factor from scroll region and request backward inferences
- Be prompted for risk factor existence and receive an output of possible risks
- Choose risk from scroll region and receive mitigation Stategies
- Add risk factor and risk association to knowledge base

2.2 Knowledge representation

The knowledge base will be constructed utilizing both a risk mechanism model and the standard work package method.

2.2.1 Risk Mechanism Model

Using this model, the risk factors will be classified into several categories and the acquisition process will attempt to input observed risks into a knowledge base. The categories being considered include:

- Performance
- Quality

- Cost
- Schedule

2.2.2 Standard Work Package Method

This method involves two matrixes bounded by the same two axes as explained in the textbook.

2.2.3 Structure of knowledge base

The proposed system will utilize a commercial tool, Knowledge Engineering System (KES), marketed by Software Architecture & Engineering, Inc., Arlington VA. KES is an expert system development tool that provides capabilities for building, maintaining and using expert systems. The tool will use an ingerence technique based on Production Rules. The domain knowledge is represented in the form of IF-THEN rules. The Production Rule Engine of KES is an object only system that uses both forward and backward chaining. KES is written in the C programming language. The system includes a library of C functions that enable the Knowledge base author to link the interface engine with other applications, forming one executable file.

The structure will look like:

IF <risk factor> and/or <risk factor> then risk

IF <risk> and/or <risk factor> then risk

Also:

IF <risk> then <risk mitigation actions>

2.2.4 Knowledge acquisition

The proposed KSS system will use interviews and questionnaires in the knowledge acquisition process. The senior managers will be interviewed in a non-threatening environment. An attempt will be made to postulate some risk as an initial forum for discussion. The interviewee will be prompted to discuss factors and situations that have led to schedule delays, additional costs, wasted resources, a decrease in the quality as judged by the user, missed market requirements, etc.

Possible topics for discussion:

- Timeliness of marketing requirements
- Unclear performance goals
- Lack of resource commitment
- Unavailability of perceived key resource
- Lack of enforced due dates for project plans
- Lack of technical knowledge by program management
- Vague marketing requirements
- Functional management redirection of resources
- Vague/incomplete software design

- Implementation started before design approved.
- Lack of testing by developer

It is expected that the process of gather risk factors, risk and risk mitigation actions will be a long process, consuming six months or longer.

2.3 Example Usage

2.3.1 A user wants to find mitigation strategies.

He will pick a risk from scroll region

RISK: Personnel shortfalls

The system will respond:

MITIGATION: Staffing with top talent, job matching; team building; morale building; cross training; pre-scheduling key people

Process:

Data stored/developed as

IF <risk> then <risk mitigation action>

2.3.2 User requests top ten historical Sequent SW risks

After clicking on the TOP 10 button (this is a windows systems being described in ascii)

The system will print out the top ten SW risk along with the successful mitigation actions.

The data is simply chained together in the knowledge base

2.3.3 User selects a risk and wants to know contributing risk factors

User selects risk for scroll region and request backward chaining.

Derived chaining is displayed on screen

Will display:

Risk n

If <risk factor y> and <risk a> then <risk n>

If <risk factor z> then <risk a>

3.0 System Development

3.1 Project Organization

The project will consist of a knowledge engineer who will double as the project leader. The knowledge engineer will be responsible for requirements definition and knowledge acquisition. This person should have a background in SW development management. A SW engineer will perform the tasks of tool development and knowledge merging with the development tool. Finally, an MMI engineer will prototype and develop the man-machine interface based on Motif and X-windows

3.2 Development stages

Figure 2 presents a PERT chart of the development stages.

FIGURE 2

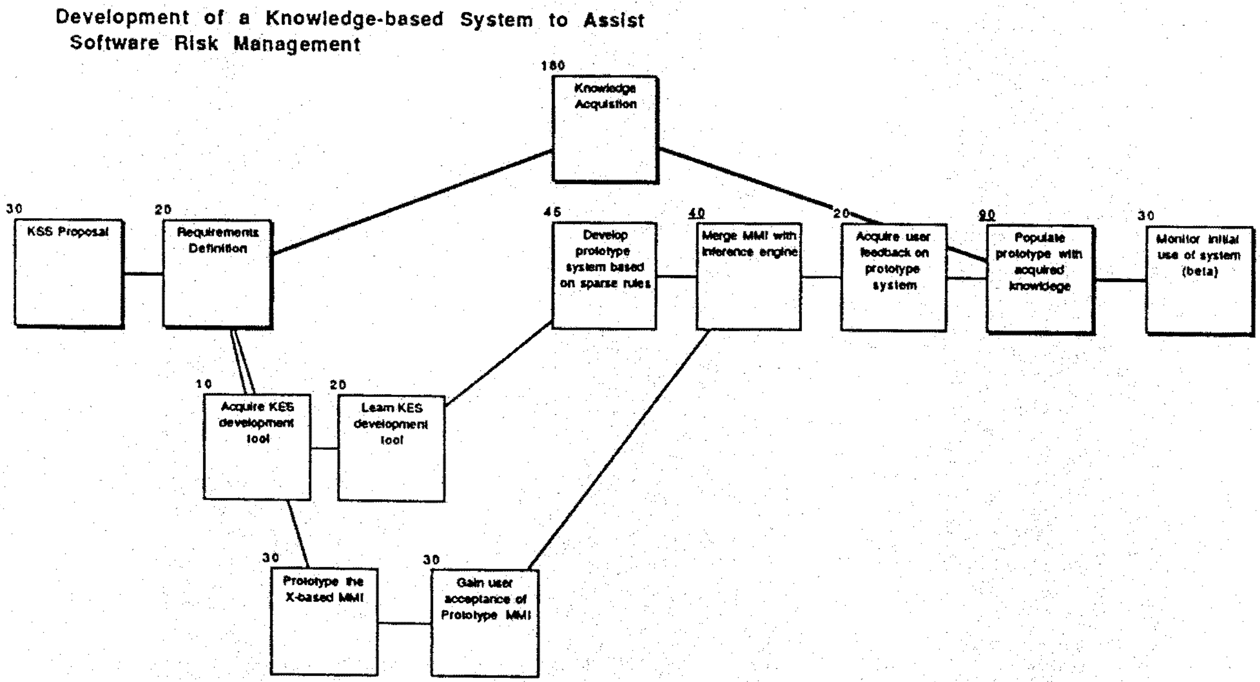

Development of a Knowledge-based System to Assist Software Risk Management
180
Knowledge Acquistion
30
KSS Proposal
20
Requirements Definition
45
Develop prototype system based on sparse rules
40
Merge MMI with inference engine
20
Acquire user feedback on prototype system
90
Populate prototype with acquired knowldege
30
Monitor initial use of system (beta)
10
Acquire KES development tool
20
Learn KES development tool
30
Prototype the X-based MMI
30
Gain user acceptance of Prototype MMI

Figure 3 illustrates a GANT chart of the development timeline

FIGURE 3 GANT chart of development timeline

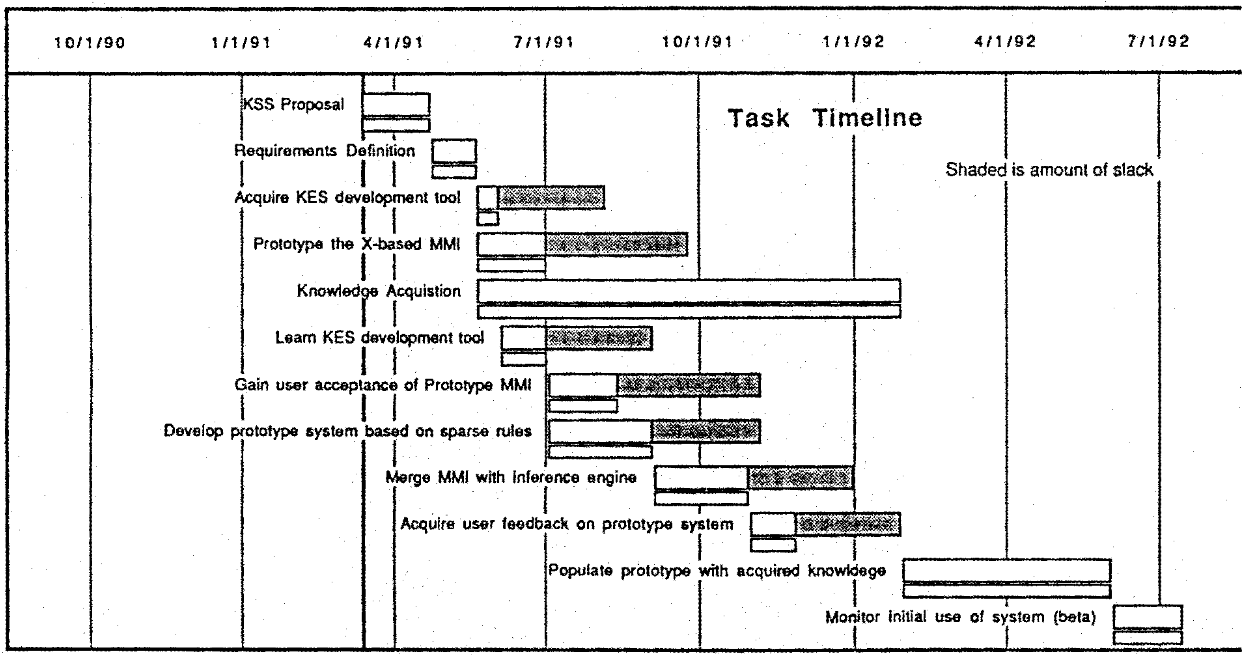
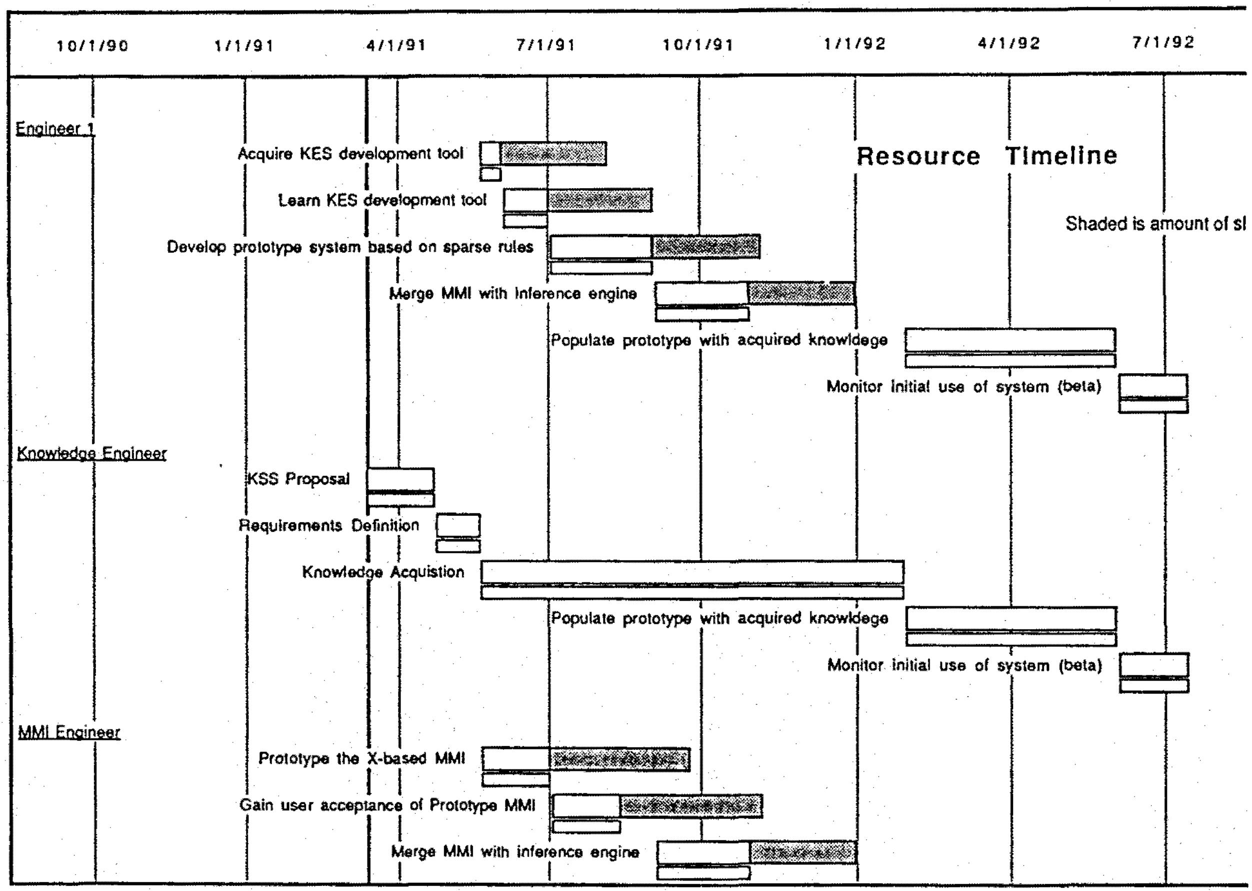


Figure 4 shows a resource allocation

FIGURE 4


10/1/90
1/1/91
4/1/91
7/1/91
10/1/91
1/1/92
4/1/92
7/1/92
Resource Timeline
Shaded is amount of sl
Engineer 1
Acquire KES development tool
Learn KES development tool
Develop prototype system based on sparse rules
Merge MMI with inference engine
Populate prototype with acquired knowldege
Monitor initial use of system (beta)
Knowledge Engineer
KSS Proposal
Requirements Definition
Knowledge Acquistion
Populate prototype with acquired knowldege
Monitor initial use of system (beta)
MMI Engineer
Prototype the X-based MMI
Gain user acceptance of Prototype MMI
Merge MMI with inference engine