Artificial Intelligence Techniques in Software Engineering (AITSE)

Engr.Farah Naaz Raza

Abstract—*Software development process is a very complex process that, at present, is primarily a human activity. Programming, in software development, requires the use of different types of knowledge: about the problem domain and the programming domain. It also requires many different steps in combining these types of knowledge into one final solution. This paper intends to study the techniques developed in artificial intelligence (AI) from the standpoint of their application in software engineering. In particular, it focuses on techniques developed (or that are being developed) in artificial intelligence that can be deployed in solving problems associated with software engineering processes. This paper highlights a comparative study between the software development and expert system development. This paper also highlights absence of risk management strategies or risk management phase in AI based systems.*

***Index Terms*—Knowledge intensive activity, Programmer's apprentice, automated programming, genetic code.**

I. Introduction

Artificial Intelligence is concerned with the study and creation of computer systems that exhibit some form of intelligence and attempts to apply such knowledge to the design of computer based systems that can understand a natural language or understanding of natural intelligence. Software Engineering is a knowledge-intensive activity, requiring extensive knowledge of the application domain and of the target software itself. Many Software products costs can be attributed to the ineffectiveness of current techniques for managing this knowledge, and Artificial Intelligence techniques can help alleviate this situation.

The goal of this research paper is to give a comparative study of how expert programmers analyze, synthesize, modify, explain, verify and document programs, and to apply that theory towards automating the programming process. Recognizing that the long-term goal of totally automatic programming is very far off, we are presently concentrating on applying our research towards developing an intelligent computer assistant for programmers, called the Programmer's apprentice. One of my key observations is that expert programmers rely heavily on a large body of standard implementation methods and program forms. A central part of the research has therefore been to identify and codify these standard forms. For this purpose many expert system programming languages have been developed in which these standard forms can be written down in a canonical and abstract way, and used by an automatic programming system.

Basically Conventional programming is a sequential, three step process: Design, Code, Debug. Knowledge engineering,

which is the process of building an expert system, also involves assessment, knowledge acquisition, design, testing, documentation and maintenance. However, there are some key differences between the two programming paradigms. Conventional programming focuses on solution, while ES programming focuses on problem. [1]

II. Software Engineering and Artificial Intelligence

Software development

Software development problems includes conceptual specifying, designing , testing the conceptual construct and representation problems that comprising representing software and testing the reliability of a representation.

The traditional view of software development process begins at the requirements specification and ends at testing the software. At each of these stages, different kinds of knowledge (design knowledge at design stage and programming and domain knowledge at the coding stage) are required. At each of the two stages: design and coding, exist a cycle: error recognition and error correction. Experience shows that errors can occur at any stage of software development. Errors due to coding may occur because of faulty design. Such errors are usually expensive to correct [2].
A basic problem of software engineering is the long delay between the requirements specification and the delivery of a product. This long development cycle causes requirements to change before product arrival.

In addition, there is the problem of phase independence of requirements, design and codes. Phase independence means that any decision made at one level becomes fixed for the next level. Thus, the coding team is forced to recode whenever there is change in design

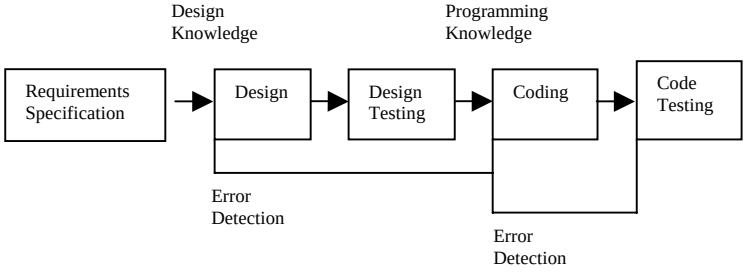

Fig. 1: Traditional software development process

Expert system development

Expert system use knowledge rather than data to control the solution process. Knowledge engineers build systems by eliciting knowledge from experts, coding, that knowledge in an appropriate form, validating the knowledge, and ultimately constructing a system using a variety of building tools.

The main phases the expert system development processes are:-

- Planning
- Knowledge acquisition and analysis
- Knowledge design
- Code
- Knowledge verification
- System evaluation

Planning phase involves feasibility assessment, resource allocation, task phasing and scheduling Requirements analysis. Knowledge acquisition is the most important stage in the development of ES. During this stage the knowledge engineer works with the domain expert to acquire, organize and analyze the domain knowledge for the ES. The goal of knowledge analysis is to analyze and structure the knowledge gained during the knowledge acquisition phase. After knowledge analysis is done, we enter the knowledge design phase. At the end of design phase, we have Knowledge definition, detailed design, and decision of how to represent knowledge decision of a development tool. Consider whether it supports your planned strategy, internal fact structure, Mock interface. Coding this phase occupies the least time in the Expert System Development Life Cycle. It involves coding, preparing test cases, commenting code, developing user's manual and installation guide.

Knowledge-based techniques in AI can be used to modify this traditional approach the AI technique that handles this problem is automated programming which results in reusable code.[4][5] Thus, when a change is made in the design, that part of the design that does not change remains unaffected. Thus, automated tools for system redesign and reconfiguration resulting from a change in the requirements will serve a useful purpose. This technique requires constraint propagation technique. With the help of automated programming approach AI based systems are free from risk management strategies.

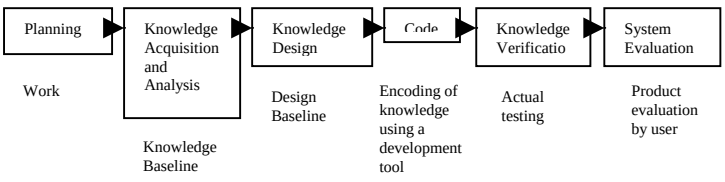

Fig.4: Expert System development

III. RISK MANAGEMENT

The Risk Management process is a method of identifying risks in advance and establishing methods of avoiding those risks and /or reducing the impact of those risks should they occur.
The process of risk management begins during the analysis phase of software development life cycle. However, the actual process of managing risks continues throughout the product development phase.

The given Figure displays the steps of the risk management process. Formally, articulated, risk management process consists of three steps:

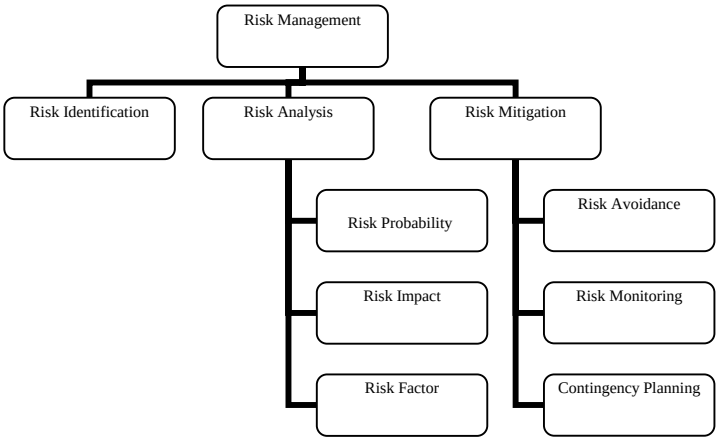

Fig. 2: Risk Management Process

AI based systems are free from risk management strategies because of automated programming techniques making data structures flexible [5]. Automatic programming is the generation of programs by computer, usually based on specifications that are higher-level and easier for humans to specify than ordinary programming languages.

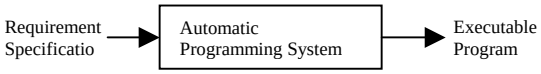

Fig. 3: Automatic Programming System (APS)

The goal is to make the specification smaller, easier to write, easier to understand (closer to application concepts), Less error-prone better than programming languages.

Genetic Code

Genetic programming is a technique which enables computers to solve problems without being explicitly programmed. It works by using genetic algorithms to automatically generate computer programs.

Evaluate each of the attempted solution
Keep the best solutions
Produce next generation from these solutions (using mutation and crossover)
Quit when you have a satisfactory solution (or you run out of time) [7]
Genetic algorithm contains a population of trial solutions to a problem. In genetic programming the individuals in the population are computer programs.

IV. Conclusion

Risk management strategies utilize lot of developer time and in software development phases there is a link between all the phases by introducing a isolation phase among the phases we can reduce the time in development by revisiting each phase after changes in requirements. By using AI based systems with the help of automated tool or automated programming tool we can eliminate risk assessment phase saving our time in software development. Because of AITSE we can reduce the development time in software development. **Coding phase in software development process can be changed into Genetic Code.**

Acknowledgment

Engr. Farah Naaz Raza thanks the cooperation of the management of Iqra University.

References

[1] Ian Sommerville, Software Engineering (6th Edn.) (Addison Wesley Publishers, New York, New York, USA) 2000

[2] Roger S. Pressman, Software Engineering: A Beginner's Guide (McGraw Hill Higher Education Publishers, New York, New York, USA) 1988.

[3] Seth Hock, Computers and Computing (Houghton Mifflin College Publishers, Boston, Massachusetts, USA) 1989.

[4] M.L. Emrich, A. Robert Sadlowe, and F. Lloyd Arrowood (Editors), Expert Systems And Advanced Data Processing: Proceedings of the conference on Expert Systems Technology the ADP Environment (Elsevier-North Holland, New York, New York, USA) 1988.

[5] Shari Lawrence Pfleeger, Software Engineering: theory and Practice (Prentice Hall Publishers, Upper Saddle River, New Jersey, USA) 1998.

[6] C.S. French, Data Processing and Information Technology (10 edition), (Letts Educational Publishers, London, United Kingdom) 1996.

[7] Artificial Intelligence (3rd Edition) Prentice Publisher, Henry Petrick Winston