International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 06 Issue: 08 | Aug 2019 www.irjet.net p-ISSN: 2395-0072

# How Artificial Intelligence Accelerates Software Development

**Divyanshi Kothari**

*Software Engineer*

---

**Abstract -** *Software Engineering is the basic methodology which is used in the development of the software. Software development is a long and time consuming process due to its complexity and the critical analysis required in the creation of the software. This paper surveys the application of Artificial Intelligence (AI) approaches to the software engineering processes. These approaches can accelerate the process, resulting in saving time and resources. This will improve the quality of software systems in general. The paper intends to study the techniques developed in AI from the standpoint of software engineering. This paper will highlight the advantages and risks involved in AI based software systems.*

***Key Words***: Artificial Intelligence Techniques, Artificial Intelligence Design Assistance (AIDA), Natural Language, Automated Software Engineering, Software Development

## 1. INTRODUCTION

Artificial Intelligence (AI) has progressed tremendously in the last few decades but with the recent finding and innovations, the field is undergoing explosive growth. Results have come from large and more complex neural networks, stacked with many layers. This domain specializes in the field of computer usage which creates and constructs computational mechanism for activities that are considered to require intelligence when performed by humans. Artificial intelligence focuses on creating machines that can engage in behaviors that humans consider intelligent.

### 1.1 Artificial Intelligence in Software Engineering

Software engineering is the formal introduction of engineering principles. It deals with the development and the creation of the software. Development of software is a long and complex process which goes through various phases and requires a feasible and executable code. Humans have been developing code for the creation of the software for a long time since no machine algorithm can do it better. AI could bring in a significant amount of advancement in the creation and development of this field.

Categories in which AI can have a great impact on the software products:

a) Ideation to code
b) Software Design
c) Intelligent Programming Assistants
d) Automated Code Refactoring
e) Strategic Decision Making

## 2. SOFTWARE DEVELOPMENT

Software development process usually consists of some traditional techniques and patterns. This includes conceptual specifying, designing, testing the conceptual construct and representation of problems that comprises the software and testing reliability of a representation. The process phases are requirements, design, design testing, coding, coding testing. The 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. This does not meet the expectations of the clients and results in loss of time and resources.

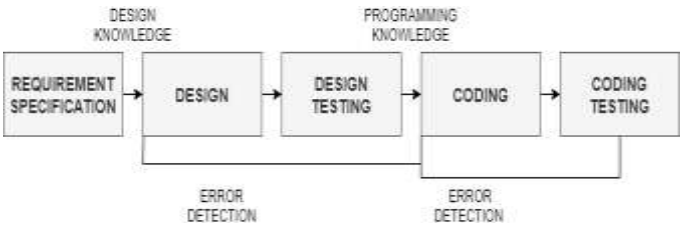


**Fig -1** Software Development Process

Here the focus will be on the major tasks of the development process based on the standards and survey some of the AI techniques used in supporting the tasks of this process. The focus will revolve around the tasks related to requirements analysis, architecture design, coding and testing.

## 3. EXPERT SYSTEM DEVELOPMENT LIFE CYCLE

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. There are nine phases of Expert System Development Lifecycle (ESDL) namely:

a) Problem Identification Phase
b) Feasibility Study Phase
c) Project Planning Phase
d) Knowledge Acquisition Phase
e) Knowledge Representation Phase
f) Knowledge Implementation Phase
g) Verification and Validation Phase
h) Installation/Transition/Training
i) Operation/Evaluation/Maintenance

## 4. SOFTWARE REQUIREMENTS

### 4.1 Requirement Engineering

The requirements needed are initially expressed in natural language. They are usually kept in document format which usually is used for analyzing. The activities that mainly carried in this phase are requirements elicitation, gathering and analysis and their transformation into a less ambiguous representation. Problems arising during this phase related to requirements are:

a) They are ambiguous
b) Incomplete, vague and imprecise
c) Data is volatile
d) Requirements are conflicting

### 4.2 Knowledge Based Systems

Knowledge Based System were used to store design families, upon the development of the requirements, input and outputs of the system's functionality. Ontology is used to reuse, integrate, and merge data and to achieve interoperability and communication among their software systems.

### 4.3 Natural Language Requirements

Natural language is used for writing system requirements and is also used for deriving requirements for users. Since they are more detail, natural language specifications are sometimes hard to understand. The requirements are usually over flexible. The requirements can be said in many different ways and hence leads to the ambiguity of the requirements. To use KBS at the maximum potential, you may have to look at every requirement rather than a group of requirements.
There are three main problem that often arise while writing down the requirement such as:

a) Lack of clarity
b) Requirements confusion
c) Requirements amalgamation

## 5. SOFTWARE ARCHITECTURE DESIGN

Artificial Intelligence uses quality attributes to define a function for an architecture. Some of the quality attributes are mainly used for the design used in developing the architecture for modularity, complexity, modifiability, understandability and reusability. Modularity is usually connected to the concept of coupling and cohesion.

Genetic Algorithms (GAs) can be used to search the space of possible hierarchical decompositions of a system. A fitness function can be used for capturing the data coupling and control coupling between components. The quality attribute in the fitness function is related to the complexity and modularity of the produced architecture.

Product Line Architectures, another AI technique, where variation points are explicitly defined to enhance reusability and modifiability of the code and architecture of the software.

## 6. SOFTWARE CODING AND TESTING

### 6.1 Coding

Software engineers can apply AI techniques to automate the programming process. These techniques can also assist the engineers to create efficient code for the requirements made. With the help of AI, an Expert System can be created to assist the engineers for software development. This is known as Programmer's Apprentice Project. This has the capability of interacting with human programmers the same way the humans would assist. This would in return increase productivity. These systems can generate functions and data structures needed for the entire program.

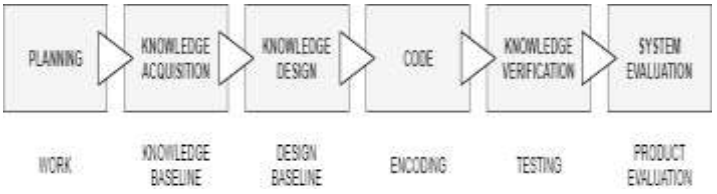


**Fig -2** Expert System Development

### 6.2 Testing

Testing is one of the most expensive tasks in the development phase due to AI automation. These techniques can play a huge role in solving various constraints faced during the development and testing process. Mutation testing, generic algorithms can be developed for automating the process. Generic algorithms are now commonly used for this purpose. Fuzzy logic, another AI technique can also be used to expediate the process and to automate the process completely.

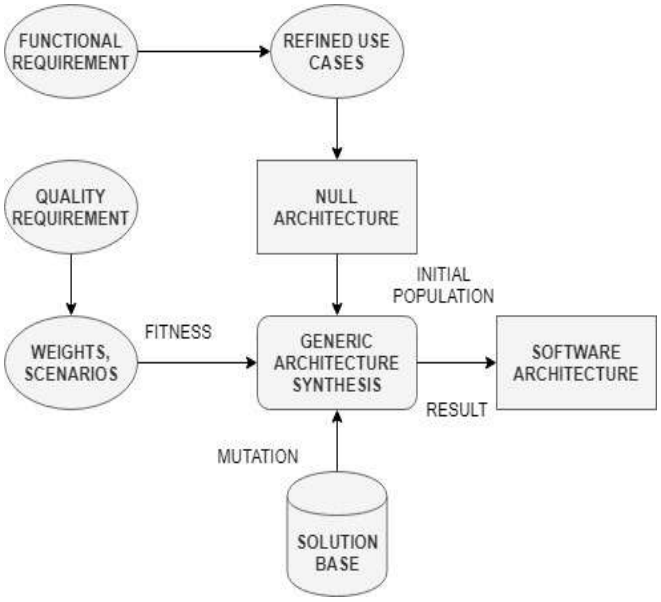


**Fig -3** Evolutionary Architecture for Software Development

## 7. RISK MANAGEMENT

Risk Management is used for identifying the risk involved while establishing methods. These methods are implemented to avoiding those risks and reduce the impact of those risks Process of risk management usually starts during the analysis phase of the life cycle process. AI based systems are free from risks due to the automated technique used. Automatic Programming System (APS) is one of the techniques used to avoid risks. Code generic programming is another technique that allows to solve problem without explicitly programming.

## 8. CONCLUSION

Software engineering helps in building various software products but due to software engineering principles, the development process of the product delays. The quality of the product can be increased by using AI techniques in the software development lifecycle. By using AI based systems with the help of automated programming tools, we can eliminate the risk assessment phase, which in return saves our time in software development and hence builds an effective product. This paper demonstrated promising research work in the field of Artificial Intelligence in terms of Software Engineering and its practices.

## REFERENCES

[1] D. Kornack and P. Rakic, "Cell Proliferation without Neurogenesis in Adult Primate Neocortex," Science, vol. 294, Dec. 2001, pp. 2127-2130, doi:10.1126/science.1065467.

[2] M. Young, The Technical Writer's Handbook. Mill Valley, CA: University Science, 1989.

[3] R. Nicole, "Title of paper with only first word capitalized," J. Name Stand. Abbrev., in press.

[4] K. Elissa, "Title of paper if known," unpublished.

[5] Coulin, C., Zowghi, D., & Sahraoui, A. (2010). MUSTER: A Situational Tool for Requirements Elicitation. In F. Meziane, & S. Vadera (Eds.), Artificial Intelligence Applications for Improved Software Engineering Development: New Prospects (pp. 146-165)

[6] Harmain, H. M., & Gaizauskas, R. (2003). CM-Builder: A natural language-based CASE tool for object-oriented analysis. Automated Software Engineering Journal, 10(2), 157–181

[7] Hewett, Micheal, and Rattikorn Hewett (1994). 1994 IEEE 10th Conference on Artificial Intelligence for Applications.

[8] Hull, E., Jackson, K., & Dick, J. (2005). Requirements Engineering. Berlin: Springer.

[9] Kof, L. (2010). From Textual Scenarios to Message Sequence Charts. In F. Meziane, & S. Vadera (Eds.), Artificial Intelligence Applications for Improved Software Engineering Development: New Prospects (pp. 83-105).

[10] Smith, T. J. (1993). READS: a requirements engineering tool. Proceedings of IEEE International Symposium on Requirements Engineering, (pp. 94–97), San Diego. SSBSE (2010). http://www.ssbse.org, checked 10.5.2011.

[11] Vadera, S., & Meziane, F. (1994). From English to Formal Specifications. The Computer Journal, 37(9), 753–763.