ARTIFICIAL INTELLIGENCE AND SOFTWARE ENGINEERING- POINT OF VIEW

Baswaraju Swathi
Research Scholar (CSE), Jain University, Bangalore

Abstract

Artificial intelligence is a Promising Technology and is progressing into the business marketplace. Artificial Intelligence impact on society is increasing rapidly, in speech and language technology, strategic development and analysis, process and system manage, vision and authentication systems, information retrieval and data-mining and many other areas. The many new realizations frequently redefine which applications we can achieve and push existing technology to Artificial Intelligence limits. Today a knowledge society is emerging, reasoning with knowledge is becoming a vital issue. The mere truth that knowledge is power makes the significance of artificial intelligence indubitable. Software Engineering is a discipline that provides models and processes that guide to the construction of well documented and maintainable software in a mode that is predictable. Artificial Intelligence and Software Engineering deal with modeling real world objects from the real world like business processes, expert system or knowledge Base System, process models and have commonalities.

Index terms: Artificial Intelligence, Expert System, Knowledge Based System, Software Engineering.

1. INTRODUCTION

This paper presents the relative importance of Artificial Intelligence and Software Engineering to solve Software Engineering problems. AI and SE domains are discussed and the later concentrates on the symbiosis of Human Intelligence and Artificial Intelligence, examples of AI techniques applied into SE.

A. Artificial Intelligence- Point Of View

AI is the theory and development of computer systems able to perform tasks usually requiring human intelligence, for instance visual perception, speech recognition, decision-making, and translation among languages. As a field of academic study, many AI researchers reach to realize intelligence by becoming able to construct things of intelligence referred as intelligent behavior. The problem of creating Intelligence includes Reasoning, Problem Solving, Knowledge Representation, Planning, Learning, Natural Language Processing and Perception.

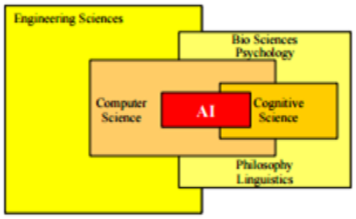

Fig.1. AI in Engineering Science[adapted from[1]]

Artificial Intelligence has been used in various fields which includes medical diagnosis, stock trading, robot control, scientific discovery. The fig 1 shows the AI in Engineering Science.

Software Engineering is the scientific thread orientating towards cognitive science and humanities and in general could be a helpful direction for interdisciplinary research. Certainly, there is a sturdy overlie between SE and the engineering strand of AI. An important part of the latter is KBS.

Richter [2] defines three different levels as essential for describing KBS:

- Cognitive layer –focuses on reasoning and understanding at a higher level, human oriented.
- Representation layer –logical layer
- Implementation layer- machine oriented.

B. Software Engineering- Point Of View

Software Engineering is a cognizance-intensive action, requiring wide-ranging erudition of the application domain and of the objective software itself. Many Software Engineering costs can be credited to the incompetence of current techniques for managing this cognizance, and Artificial Intelligence techniques can avail ease these circumstances. More than two decades of research have led to many consequential speculatively theoretical consequences, but few demonstrations of authentic value. This is appropriate in part to the quantity and diversity of learning required by Software Engineering activities, and the fact that much of the research has been barely focused, missing lots of issues that are of great practical significance. Paramount issues that remain to be addressed include the representation and utilization of domain erudition and the representation of the history of design and implementation of a software system. If solutions to these issues are found, and experiments in practical situations are prosperous, the implicative insinuations for the practice of Software Engineering will be profound, and radically different software development paradigms will become possible.

For a established process it should be possible to establish in advance the time and effort required to engender the final product. This can only be done utilizing data from past experience which requires that one measure the software process. The disciplines of Artificial Intelligence and Software Engineering have many common points. Both deal with modeling real world objects from the real world like business processes, expert cognizance, or process models.

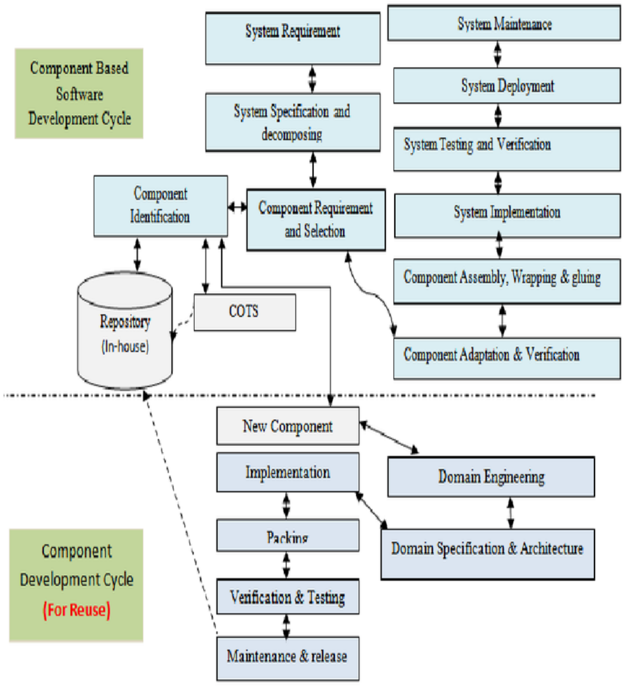
Component Based Software Development Cycle
System Requirement
System Maintenance
System Deployment
System Specification and decomposing
System Testing and Verification
Component Identification
Component Requirement and Selection
System Implementation
Component Assembly, Wrapping & gluing
Repository (In-house)
COTS
Component Adaptation & Verification
New Component
Implementation
Domain Engineering
Packing
Domain Specification & Architecture
Component Development Cycle (For Reuse)
Verification & Testing
Maintenance & release

Fig.2. Software Development Reference Model

C. Artificial Intelligence and Software Engineering

There is a vigorous overlap between SE and the engineering strand of AI[3]. A paramount part are KBS (Knowledge Based System)[4].It can provide the initial technology and first (prosperous) applications as well as a testing environment for conceptions. The inclusion of research fortifies the enabling of human-enacted processes and increases utilize acceptance. AI technology can avail to predicate the overall SE method on a concrete technology, providing adequate detail for the initial method description, and through the available reference technology elucidating the semantics of the respective method. Addition to this, other AI techniques naturally substituting/extending the chosen technology can be used for improved versions of the SE method for amended versions of the SE method.

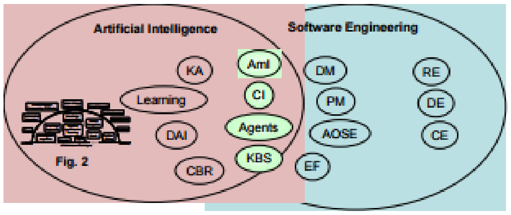

Fig.3. Intersection of AI and SE [adapted from [1]]

II. Human Intelligence and Artificial Intelligence

According to[5] software-engineering activities usually require rigorous human efforts and inflict weight on human intelligence. To reduce human efforts and encumber on human Intelligence in these activities, Artificial Intelligence (AI) techniques, aim to engender software systems that exhibit some form of human Intelligence, have been employed to avail or automate these activities in software engineering.

Current AI techniques employed in software-engineering activities are

- Constraint solving [6].
- Search heuristics [7] in test generation.
- Machine learning [8] in debugging.
- Natural language processing [9] in specification inference,
- Knowledge engineering [10] used in a variety of software-engineering activities along with search-based techniques generally applied in Search-based software engineering [11].

These Techniques suited well for automating repetitive standardized subtasks by incorporating human intelligence in to software engineering activities and also proved that these are able to handle large number of tasks i.e., the scale of the data under analysis. Alternatively, human intelligence has also been influenced in the wide context of human-centric software engineering [12]. In general human's domain knowledge can be provide as starting points for designing AI techniques. In addition, the results of AI techniques are often interpreted or demonstrated by human users. Such user feedback could be incorporated to further progress the AI techniques, forming a constant feedback loop.

III. Intelligent Automated Methods in Software Testing

According to [14] Automation testing has several advantages such as increasing testing speed, quality and reliability, reduce testing resources and costs. In automated testing, the testing phases or sub phases can be performed by intelligent methods, in order to reduce human intervention in the process. This paper classifies the methods in automation software testing and provides AI techniques used in the different phases.

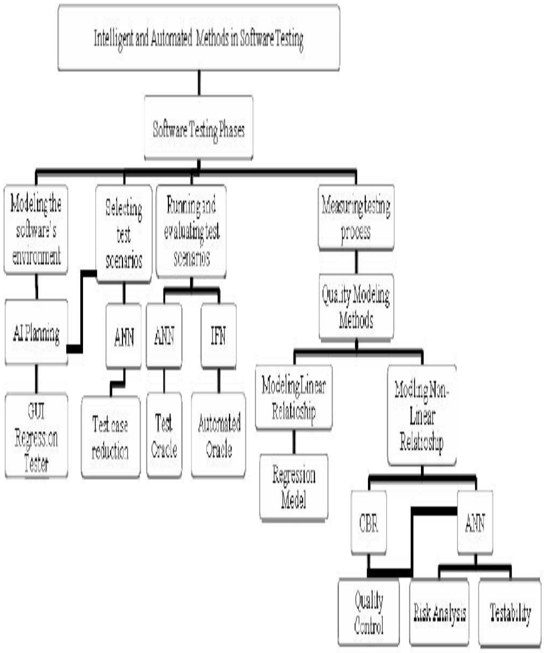

Fig.4. Intelligent Automated Methods in Software Testing.

- Modeling Software Environment uses AI planning (to generate effective test cases), to replicate associations and communications between software and its environment.
- Selecting Test Scenarios uses Artificial Neural Network (to automating I/O analysis by Identifying important attributes and ranking them) to reduce Test Cases.
- Running and Evaluating test scenarios uses Artificial Neural Network and Info Fuzzy Network to develop Knowledge Discovery and data mining.

Research on Automation Testing Framework for Multi Platform Mobile applications[15] discusses about a proposed Cross-Platform Testing Framework i.e., MATF(Multi Platform Automated testing Framework),which can parse the test cases and generate the test scripts for both iOS and Android platforms. Lot more Research can be done for integrating both intelligent automation methods and cross-platform automated framework.

IV. Integrating AI and Test generation

Conventional test generation deals with human intelligence to design test inputs that achieves high code coverage, fault detection. Automated test generation tools [13] automatically generate test inputs which can satisfy testing requirements. In human-assisted computing, test-generation tools are in the driver seat and may face challenges in test generation; for which, users can supervise such as scripting mock objects, or including factory methods for encoding method sequences to construct enviable object states. Human-centric computing, users are in the driver seat and may get supervision from tools on subtasks such as what assertions need to be written and what method sequences need to be written or translating natural-language test descriptions to executable test scripts.

V. Initial Steps on an On-Ground Autonomy Test Environment

The paper [16] demonstrates a case study on OGATE an ESA funded project that aims to facilitate accurate experiments on planning and execution systems for robotics. To control the robots in dynamic environments with autonomy AI controllers are incorporated as one of the top layer in this system.

The advances in the Artificial Intelligence (AI) give the impression to be naturally merged with the control of robotic systems in order to sanction it to engender long term plans without or little human intervention. In this way, developments in orchestrating and scheduling systems, such as task planning [17], CSPs [18], timelines and, recently, the efforts interleaving orchestrating and execution could be very valuable to control robots in dynamic environments with an incrementing degree of autonomy.

The AI controllers proposed here are the top layer of complex tools that are customarily made ad-hoc to control a categorical robotic platform to perform determinate missions. Customarily, to test and verify the correctness of architecture, a minute set of missions are carried out. Additionally, some components of the control system could be evaluated in a standalone manner via particular test beds. But testing the robustness, adequacy and performance for the whole control architecture cannot be that easily done; it requires to a mass and to analyze relevant data from all the components of the control system while the test bed covers more cases than the typical scenarios. To cope with the difficulty have conceived the conception of engendering a software environment to be utilized on-ground to facilitate the demonstration and testing of software for autonomy. Such an environment can represent the seed for a future Knowledge engineering environment for autonomous controllers. Certainly the first objective for such an environment is:

- Facilitating the utilization of autonomous controllers.
- Sanctioning the utilization of different solution for autonomous control.
- Enabling the comparison of different solutions congregates reliable execution data on a given mission.

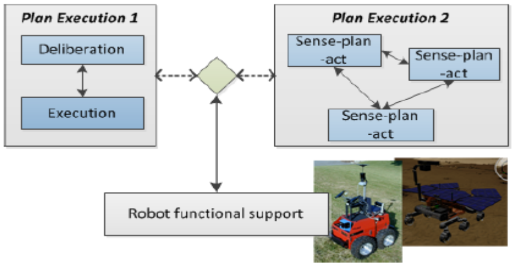

Fig.5. Different plan executions for controlling the same robotic platform.

VI. Conclusion

This paper presents a review of how the Artificial Intelligent Techniques are being incorporated in to Software Engineering and focuses on some of the examples where AI is being incorporated. Examples included Test Generation, Intelligent Automated Testing Methods and an AI Planner for creating an On Ground Autonomy Test Environment applicable for Robotics.

Acknowledgement

I am extremely thankful to my guide Dr. Harshavardhan Tiwari for his guidance and support.

I extend my thanks to Jain University management for their constant encouragement and support for doing the research work.

References

[1]. Wahlster W., "Einführung in die Methoden der Künstlichen Intelligenz," University of Saarbrücken, Germany,
Lecture Notes 2002.

[2]. Richter M. M., "Artificial Intelligence," University of Calgary, Canada, Lecture Notes 2004.

[3]. The relationships of AI to software engineering, D.Partridge ,IEEE Colloquium on Software Engineering and AI(Digest No.087).

[4]. Avram Gabriela “Empirical Study on Knowledge Based Systems” The Electronic Journal of Information Systems Evaluation, Vol. 8, Iss.1,2005.

[5]. The Synergy of Human and Artificial Intelligence in Software Engineering Tao Xie North Carolina State University,Raleigh, NC, USA, 2013 IEEE,RAISE 2013, San Francisco, CA, USA.

[6]. P. Godefroid, N. Klarlund, and K. Sen. DART: Directed automated random testing. In *Proc. PLDI*, pages 213–223, 2005.

[7]. T. Xie, N. Tillmann, P. de Halleux, and W. Schulte. Fitness-guided path exploration in dynamic symbolic execution. In *Proc. DSN*, pages 359–368, 2009.

[8]. Debugging Machine Learning Tasks Aleksandar Chakarov A, Aditya Nori B, Sriram Rajamani B , hayak Sen C, and Deepak Vijaykeerthy D AUniversity of Colorado, Boulder BMicrosoft Research CCarnegie ellon University DIBM Research March 24, 2016.

[9]. Inferring Resource Specifications from Natural Language API Documentation Hao Zhong1,2 , Lu Zhang , Tao Xie, Hong Mei,Key Laboratory of High Confidence Software Technologies (Peking University), Ministry of Education, China 2 Institute of Software, School of Electronics Engineering and Computer Science, Peking University, China 3Department of Computer Science, North Carolina State University, Raleigh, NC 27695-8206, USA.

[10]. 4th Workshop on Knowledge Engineering and Software Engineering KESE 2008 at the 31st German Conference on Artificial Intelligence.

[11]. Search Based Software Engineering: Techniques, Taxonomy, Tutorial Mark Harman , Phil McMinn , Jerffeson Teixeira de Souza , and Shin Yoo.

[12]. Trends in the Quality of Human-Centric Software Engineering Experiments--A Quasi-Experiment. Barbara Kitchenham; Dag I. K. Sjøberg; Tore Dybå; O. Pearl Brereton; David Budgen; Martin Höst; Per Runeson,IEEE Transactions on Software Engineering,Year: 2013, Volume: 39, Issue: 7.

[13]. TesMa and CATG: Automated Test Generation Tools for Models of Enterprise Applications Haruto Tanno NTT Laboratories, Japan. Xiaojing Zhang NTT Laboratories, Japan. Takashi Hoshino NTT Laboratories, Japan. Koushik Sen EECS Department UC Berkeley, CA, USA.

[14]. Intelligent and Automated Software Testing Methods Classification Seyed Reza Shahamiri,Department of Software Engineering Faculty of Computer Science and Information Systems, University Teknologi Malaysia (UTM), Wan Mohd Nasir Department of Software Engineering Faculty of Computer Science and Information Systems University Teknologi Malaysia (UTM).

[15]. Research on Automation Testing Framework for Multi Platform Mobile applications, Da Zun,Tao Qi,Liping Chen, Proceedings of CCIS2016.

[16]. First Steps on an On-Ground Autonomy Test Environment, Pablo Muñoz, Amedeo Cestay, Andrea Orlandiniy and María D. R-Moreno, 2014 IEEE International Conference on Space Mission Challenges for Information Technology.

[17]. R. E. Korf, “Planning as Search: A Quantitative Approach,” Artificial Intelligence, vol. 33, pp. 65–88, 1987.

[18]. A. Cesta, S. Fratini, and A. Oddi, “Planning with Concurrency, Time and Resources: A CSP-Based Approach,” in Intelligent Techniques for Planning, I. Vlahavas and D. Vrakas, Eds. Idea Group Publishing, 2005, pp. 259–295.