**Coders Compass: An Expert System Helping Navigate the Software Development Landscape**

Ahmed Mohamed Ismail[[1]](#footnote-1) and Khaled Gamal ElLethy[[2]](#footnote-2)

Computer Engineering Department, Faculty of Engineering, Cairo University, Egypt

**ABSTRACT**

In the realm of software development, the selection of appropriate frameworks and tools is crucial for project success. However, with the myriad of options available, developers often face challenges in making informed decisions. To address this issue, this research proposes the utilization of expert systems for recommending frameworks and tools tailored to specific project requirements and developer preferences. While the evaluation of recommendation systems poses challenges due to subjectivity, efforts have been made to validate the approach. Future work will focus on enhancing the system's capabilities, including continuous updates to the knowledge base and the potential integration of machine learning techniques for improved recommendations.

KEYWORDS: expert systems, framework recommendation, tool recommendation, software development, rule-based inference, personalized recommendations

**1. INTRODUCTION**

In today's dynamic technological world, the selection of an appropriate programming language is a crucial decision for developers starting a new project. The myriad of programming languages available, each tailored to specific domains, platforms, and learning curves, presents a formidable challenge. What makes it more problematic is that each programming language comes with its set of frameworks and tools, each created for a specific need.

In response to this complexity, we propose the usage of expert systems as a promising approach to assist developers in the decision-making process of selecting a framework and tools to use in their upcoming projects.

This research aims to explore the effect of employing expert systems in the domain of framework and tools recommendation for various projects. The study aims to develop a framework that integrates multiple criteria essential for language selection, including domain suitability (e.g. web development, mobile development, etc.) and learning curve considerations (e.g. steep learning curve). Additionally, the system will take into account users' previous knowledge and familiarity with programming.

By systematically analyzing these criteria and leveraging expert knowledge, the proposed system seeks to provide personalized and informed recommendations to developers, making the decision-making processes easier and ultimately enhancing the success and quality of software development projects.

**2. RELATED WORK**

In our research, we encountered limited or no prior studies that has the same objective as our approach. This is because our approach is a new application of expert systems. Nonetheless, we searched for similar research that is almost comparable to our approach. In 2011, (Shishehchi et al., 2011)used an ontology-based system for recommending learning materials for the Visual Basic.NET programming language, employing a semantic recommender system that considers the learner's knowledge level and requests. The system comprises several subsystems and components, including a knowledge-based system that covers the VB.NET ontology, learner performance evaluation, recommendation system, availability checker, knowledge evaluator, exam generator, request analyzer, and user interface.

Another interesting study is(Odiete et al., 2017). This research employed graph theory to recommend which programming language should be learned. It utilized social network analysis techniques to suggest programming languages to both new and experienced programmers. Additionally, the study utilized an expertise graph to determine the significance of a programming language within a community. The experts involved were sourced from the Stack Overflow community.

**3. PROPOSED APPROACH**

**3.1 Domain Definition**

When defining our approach, we aimed for a narrow domain (Waterman, 1985). To achieve this, we began by outlining the fields that users could select from. The fields we defined include: front-end development, back-end development, mobile development, automation testing, and DevOps (Development Operations), all within the software development domain.

Although the realm of software development is vast, we opted for the most popular fields to narrow down the domain as much as possible, whilst making it valuable for both novice and experienced developers.

**3.2 Knowledge Base**

In this study, we adopt a rule-based representation for the knowledge base, which allows us to effectively capture and utilize the information in our domain. The information we captured in our knowledge base is the following:

1. Field: Each framework in the knowledge base is tied to one field only. For example, Django framework is a backend framework that is written in Python.
2. Previous knowledge: we need to take into consideration the experience level of the user. That is why we embedded the experience level recommended with each framework and tool in the knowledge base. The previous knowledge can be on of the following values: (Beginner, Intermediate and Advanced)
3. Learning curve: When choosing resources for a new project, it is crucial to take into consideration the deadline of the project. For that reason, we added the learning curve as some frameworks are harder to learn due to their complexity, while others can be used to put together an interesting demo. The values of the learning curve are: (Gentle, Normal, and Steep).

Currently, the knowledge base contains information about 41 frameworks and tools from the fields mentioned above.

**3.3 Inference Engine**

In our proposed approach, we utilize an inference engine to process the rules and information stored in the knowledge base as well as the input from the user. The inference engine operates by applying logical rules to the available data to draw conclusions or make recommendations. We employ a rule-based approach for the inference engine, leveraging the Experta library in Python[[3]](#footnote-3).

The Experta library provides a flexible and efficient framework for defining rules and executing them based on the provided facts. Each rule consists of conditions and actions, where conditions specify the criteria that must be met for the rule to be applicable, and actions define the consequences or recommendations resulting from the rule.

Within our system, the inference engine utilizes the rules defined in the knowledge base to make recommendations tailored to the user's specific requirements and context. For example, if a developer indicates a preference for front-end development with a gentle learning curve and intermediate previous knowledge, the inference engine will evaluate the available frameworks and tools to suggest options that align with these criteria.

By employing an inference engine powered by expert rules, our approach can efficiently process complex decision-making scenarios and provide personalized recommendations based on the user's input and the expertise encoded in the knowledge base. This enables developers to make informed decisions when selecting frameworks and tools for their projects, ultimately contributing to improved project outcomes and developer satisfaction.

**3.4 Interface**

To ensure accessibility and ease of use for developers, we have designed a simple yet effective terminal interface for interacting with the expert system. The interface allows users to input their project requirements and preferences, prompting the system to provide tailored recommendations based on the provided information.

Upon launching the interface, users are greeted with a clear prompt inviting them to input their project details. The interface guides users through the process, prompting them to specify the development field they are working in (e.g., front-end, back-end, mobile, etc.), their level of previous knowledge (beginner, intermediate, advanced), and their desired learning curve (gentle, normal, steep).

Once the user has provided all necessary inputs, the interface triggers the inference engine to process the information and generate recommendations. The recommendations are then displayed in a structured format, presenting the suggested frameworks and tools that best match the user's criteria.

**4. RESULTS**

Given the subjective nature of recommendation systems, evaluating their outputs can be challenging. Nonetheless, we made tremendous efforts to validate the approach's results to the best of our ability. Initially, we relied on our own assessment, which yielded favorable outcomes. However, we recognized the bias in evaluating our own creation objectively.

Consequently, we determined that user evaluation is crucial. Unfortunately, no users have had the opportunity to test the model since this is still research. Therefore, we highly encourage readers to engage with the model and share their feedback with us via our contact information.

**5. CONCLUSIONS**

In conclusion, this research introduces a novel approach utilizing expert systems for recommending frameworks and tools in software development projects. By using expert knowledge and a rule-based inference engine, the proposed system aims to assist developers in making informed decisions tailored to their specific project requirements and preferences.

Through the development of a knowledge base containing information about various frameworks and tools, along with criteria such as domain and learning curve, the system can generate personalized recommendations based on user input. The inference engine processes these recommendations efficiently, providing users with valuable insights to aid their decision-making process.

**6. FUTURE WORK**

Regarding future work, it primarily focuses on enhancing the capabilities and effectiveness of the expert system in recommending frameworks and tools for software development projects. One key aspect we aim to address is the continuous updating and expansion of the knowledge base to ensure its relevance and comprehensiveness. Another area could be assembling a dataset that can serve as a benchmark for future work or even be used as training data for approaches that utilize machine learning.

**REFERENCES**

1. Odiete, O., Jain, T., Adaji, I., Vassileva, J., & Deters, R. (2017). Recommending Programming Languages by Identifying Skill Gaps Using Analysis of Experts. A Study of Stack Overflow. *Adjunct Publication of the 25th Conference on User Modeling, Adaptation and Personalization*, 159–164. https://doi.org/10.1145/3099023.3099040
2. Shishehchi, S., Banihashem, Y., Azman, S., & Noah, M. (2011). Learning Content Recommendation for Visual Basic.Net Programming Language based on Ontology. *Journal of Computer Science*, *7*(2), 188–196.
3. Waterman, D. A. (1985). *A guide to expert systems*. Addison-Wesley Longman Publishing Co., Inc.

1. ahmedmoh123@hotmail.com [↑](#footnote-ref-1)
2. ellethykhaled2@gmail.com [↑](#footnote-ref-2)
3. Link to the library: https://github.com/nilp0inter/experta/ [↑](#footnote-ref-3)