Roadmap for Software Engineering Education using ChatGPT

Aly Maher Abdelfattah
Faculty of computer Science and Engineering
Galala university
Egypt
aly.Abdelrahman@gu.edu.eg

Nabila Ahmed Ali
Faculty of computer Science and Engineering
Galala university
Egypt
Nabila.Hafez@Gu.edu.eg

Mohamed Abd Elaziz
Faculty of computer Science and Engineering
Galala university
Egypt
m.elsaied@gu.edu.eg

Hany H Ammar
Faculty of computer Science and Engineering
Galala university
Egypt
hany.ammar@gu.edu.eg

2023 International Conference on Artificial Intelligence Science and Applications in Industry and Society (CAISAIS) | 979-8-3503-1478-6/23/$31.00 ©2023 IEEE | DOI: 10.1109/CAISAIS59399.2023.10270477

***Abstract*—This paper presents a novel method for teaching software engineering using the AI tool, ChatGPT, to create an engaging and immersive learning platform. The technique emphasizes understanding requirements engineering principles via interactive exercises and hands-on examples. The approach involves ChatGPT assisting in collecting user stories, creating a use case and class diagrams, and formulating sequence diagrams. This method employs an agile strategy focusing on select user stories and encourages student interaction with ChatGPT for a deeper understanding of the subject. The goal is to demonstrate the potential of AI tools to transform software engineering education by providing practical applications and real-life scenarios. The research outlines a comprehensive plan for integrating ChatGPT into a software engineering syllabus, focusing on requirements engineering. The findings could significantly influence the teaching and understanding of software engineering principles, benefiting educators and students.**

Keywords: Software engineering education; Interactive learning environment; ChatGPT; Requirements engineering, Agile process; AI in education.

I. INTRODUCTION

Software engineering education is vital in preparing students for the multifaceted field of software development, with a key focus on mastering requirements engineering [1,2]. Recently, language models such as ChatGPT-3 and ChatGPT-4 have come to play a crucial role in this process. The introduction of these intelligent models, particularly ChatGPT-4, has ushered in a new era in natural language processing, exhibiting significant enhancements over previous iterations like ChatGPT-3, especially in the realm of software engineering education. ChatGPT-4 has been greatly improved in terms of accuracy over its predecessor, ChatGPT-3, due to advancements in its core architecture and training mechanism. These improvements particularly benefit software engineering education, as the model's ability to comprehend and respond to related scenarios is significantly amplified. This results in more effective interactive student learning [3]. Furthermore, ChatGPT-4 improves not only in accuracy but also in performance. The system's response generation is faster and more efficient, reducing latency and providing a smoother interaction between students and the model. This creates a real-time conversational learning experience that is more dynamic. The enhanced accuracy and performance of ChatGPT-4 substantially benefit the interactive learning process. It provides more accurate and relevant software engineering information, allowing students to gain a deeper understanding of the subject. Also, the prompt interaction with the model creates a highly engaging learning environment, which facilitates quicker knowledge acquisition. Despite the improvements in ChatGPT-4, the future promises even more advancements in language models, leading to further enhancements in output quality and efficiency. Future language models could be increasingly integrated into software engineering education, continually enriching students' learning experiences. ChatGPT is used as an intelligent assistant, aiding students in the creation of user stories, use case diagrams, class diagrams, and sequence diagrams. This approach improves students' understanding of requirements engineering and promotes an interactive learning environment. Students can engage directly with requirements engineering, providing practical, experiential learning. To manage the complexity of the field, an agile process is adopted, focusing on developing one or two user stories at a time and keeping generated diagrams simple. This interactive learning setting encourages active student engagement, fostering critical thinking, problem-solving skills, and a deeper comprehension of requirements engineering concepts.

This paper provides a step-by-step guide for utilizing ChatGPT in the software engineering educational process. It outlines the methodology for defining user stories, generating use case diagrams, developing class diagrams, and creating sequence diagrams. Additionally, it showcases examples of user stories and diagrams derived from the project, demonstrating their relevance to the software engineering domain. In conclusion, integrating ChatGPT into software engineering education opens up new possibilities for interactive and effective learning [4,5]. By providing a virtual assistant that assists in the requirements engineering process, students can acquire practical skills in a dynamic and engaging manner.

The rest of this paper is organized as follows: the background and related work are given in section II. Section III presents the methodology. section IV presents the discussion section. The conclusion and future work is presented in section V.

Authorized licensed use limited to: Hochschul- und Kreisbibliothek Bonn-Rhein-Sieg. Downloaded on April 09,2024 at 16:57:12 UTC from IEEE Xplore. Restrictions apply.

II. BACKGROUND AND RELATED WORK

Requirements engineering plays a crucial role in software development, serving as a foundation for successful project outcomes. It encompasses activities such as elicitation, analysis, documentation, and management of stakeholder needs and expectations. Effective requirements engineering ensures that the software system satisfies user requirements and aligns with organizational goals. Traditionally, requirements engineering relies on techniques like interviews, surveys, and document analysis to gather and shape requirements. However, these methods often lack interactivity, resulting in inadequate student engagement during the learning process. As a result, there is an increasing demand for innovative techniques that aim to enhance the educational experience of software engineering students in requirements engineering. Also, Requirements modeling is a critical component of requirements engineering, enabling stakeholders to visually represent and understand system requirements. Various techniques and notations have been developed to facilitate requirements modeling, including use case diagrams, class diagrams, and sequence diagrams. Use case diagrams depict the interactions between actors (users) and the system, illustrating the functional requirements of the software [6]. Class diagrams represent the static structure of the system, showcasing the relationships between classes, attributes, and methods. Sequence diagrams focus on dynamic behavior, displaying the chronological flow of messages between objects for a specific scenario of interactions based on a specific use case. Previous research has explored the application of various tools and technologies to support requirements modeling, including CASE (Computer-Aided Software Engineering) tools, modeling dialects like UML (Unified Modeling Language), and methods for generating code [7]. However, there is still a need to bridge the gap between theoretical concepts and practical implementation in softwareengineering education. The integration of artificial intelligence (AI) and chatbots has shown promising potential in enhancing the learning experience in various domains, including software engineering education. Chatbots, powered by algorithms for natural language processing and machine learning, can engage students in interactive discussions, provide answers to their queries, and offer guidance. Chatbots have been utilized to simulate interactions with domain specialists, assisting in knowledge acquisition and hands-on skill development[8].They can facilitate students in comprehending complex concepts, providing instant feedback, and delivering personalized learning paths. Within the field of software engineering education, chatbots have been tested for tasks such as code reviews, debugging assistance, and project management support. The recent advancements in language models, notably OpenAI's GPT-3.5 architecture, have expanded the potential of chatbots. These models can generate human-like text and simulate interactiveconversations, opening up new possibilities for their application in software engineering education.

- ***Related Work***

Several studies have explored the integration of AI and chatbots in the domain of software engineering education. For example, researchers have developed chatbot-based systems to assist students in understanding software requirements and modeling techniques. These systems employ natural language processing techniques to extract information from students' queries and provide appropriate responses, facilitating their learning process[9]. Other studies have investigated the use of chatbots for code generation and project documentation. Chatbots have been employed to generate code snippets based on user requirements, aiding students in understanding the translation of requirements into implementation. They have also been utilized to extract information from user interactions and automatically generate project documentation, reducing the manual effort required in this process [10]. However, to the best of our knowledge, research on employing chatbots, particularly ChatGPT, in learning the requirements engineering process, including the extraction of user stories and the generation of diagrams, remains limited. This paper aims to bridge this gap by presenting an interactive learning environment that leverages ChatGPT for requirements engineering education, with a focus on user stories, use case diagrams, class diagrams, and sequence diagrams [1,3,5,7]. By incorporating the interactivity and language generation capabilities of ChatGPT, this paper offers a novel approach to engage students in requirements engineering, promoting a deeper understanding of the subject matter and facilitating their transition from theoretical knowledge to the practical application [3,5].

III. METHODOLOGY

The methodology adopted in this study is articulated through a UML activity diagram, which vividly represents the interactive process between a student and ChatGPT. The intention is to generate various software engineering artifacts starting from user stories, escalating to use case diagrams, class diagrams, and culminating with sequence diagrams.

1. Initial Requirement Provision: The process kicks off with the student supplying the necessary software requirements [2,4,5,8].

2. User Stories Generation: Drawing from these requirements, ChatGPT endeavors to generate pertinent user stories. It's noteworthy that the model uses these initial requirements to shape relevant user stories, bridging the gap between vague software needs and structured requirement specifications [6,9].

3. Iterative Feedback Loop for User Stories: The student then evaluates the quality and accuracy of these user stories. If found unsatisfactory, ChatGPT refines and regenerates them. This iterative process, dictated by student feedback, continues until the produced user stories meet the student's satisfaction[8].

4. From Stories to Visualization - Use Case Diagrams: Moving up the abstraction ladder, ChatGPT uses the solidified user stories to draft illustrative use case diagrams, providing a visual representation of different use cases[٦,9,11].

5. Iterative Feedback Loop for Use Case Diagrams: Similar to user stories, the student appraises the use case diagrams. Inconsistencies or inadequacies lead to regeneration by ChatGPT. This cycle perpetuates until the diagrams achieve the requisite standard set by the student[11].

6. Deriving Class Diagrams: Building upon the use cases, ChatGPT processes the information to derive class diagrams, encapsulating the blueprint of the system's structure[10,11].

7. Iterative Feedback Loop for Class Diagrams: The class diagrams undergo scrutiny by the student and any deviations from expectations prompt regeneration by ChatGPT. The loop remains active until the student expresses contentment with the diagrams [11].

8. Sequence Diagram Formation: Leveraging the information from the class diagrams, ChatGPT sketches out sequence

Authorized licensed use limited to: Hochschul- und Kreisbibliothek Bonn-Rhein-Sieg. Downloaded on April 09,2024 at 16:57:12 UTC from IEEE Xplore. Restrictions apply.

diagrams, detailing the interactions among classes in the system over time [10,11].

9. Iterative Feedback Loop for Sequence Diagrams: The final iteration involves the student's assessment of the sequence diagrams. Should they not align with the student's vision, ChatGPT embarks on refining them. This loop persists until the student deems the sequence diagrams satisfactory[11].

In essence, this methodology underlines the collaborative dynamics between the student and ChatGPT. The emphasis is on iterative improvement, ensuring that every artifact, from user stories to sequence diagrams, aligns perfectly with the student's expectations and the stipulated requirements.

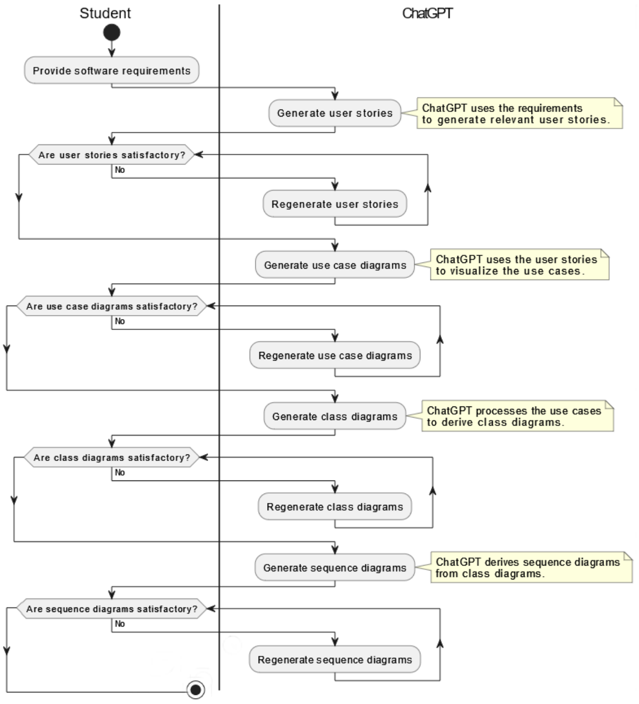
Student
ChatGPT
Provide software requirements
Generate user stories
ChatGPT uses the requirements to generate relevant user stories.
Are user stories satisfactory?
No
Regenerate user stories
Generate use case diagrams
ChatGPT uses the user stories to visualize the use cases.
Are use case diagrams satisfactory?
No
Regenerate use case diagrams
Generate class diagrams
ChatGPT processes the use cases to derive class diagrams.
Are class diagrams satisfactory?
No
Regenerate class diagrams
Generate sequence diagrams
ChatGPT derives sequence diagrams from class diagrams.
Are sequence diagrams satisfactory?
No
Regenerate sequence diagrams

Fig. 1. An Iterative Process for AI-Assisted Generation of Software Engineering Diagrams.

A . REQUIREMENTS MODELING

In the requirements engineering process, the initial step involves capturing the needs and desires of stakeholders in the form of user stories. These user stories serve as concise yet comprehensive representations of requirements, enabling a clear understanding of the software system's functionality. These user stories serve as the basis for defining the use case diagram. In this section, we present a methodology for defining user stories and for generating the use case diagram using ChatGPT [11].

1. DEFINING USER STORIES USING CHATGPT:

To engage with ChatGPT and extract user stories, software engineering students can follow the following step-by-step process:

Step 1: Contextualizing the Requirement:

Begin by providing a brief context or scenario that outlines the software system under consideration. This context helps ChatGPT better understand the specific domain and generate more relevant user stories. For example, if the project revolves around online banking, specify the main functionalities and actors involved.

Step 2: Interaction with ChatGPT:

Interact with ChatGPT by posing questions or prompts related to user requirements. Ask ChatGPT to generate user stories based on the given context. Start with a broad question to capture the overall system functionality and gradually narrow down the focus.

Step 3: Iterative Refinement:

Evaluate the user stories generated by ChatGPT and refine them iteratively. Assess their clarity, completeness, and relevance to the project. If necessary, re-engage with ChatGPT, providing additional context or posing specific questions to improve the quality of the generated user stories.

Step 4: Documentation and Review:

Document the extracted user stories in a structured format, such as using a user story template. Include information such as the actor, action, and desired outcome. Review the documented user stories to ensure they accurately reflect the stakeholders' requirements[12].

Example:

Consider an example where a student is developing an online banking application. The student engages with ChatGPT and provides a contextual description of the application, including its main functionalities and actors involved. The student then interacts with ChatGPT, asking it to generate user stories. A sample conversation with ChatGPT could be as follows[10]:

Student: "Please provide user stories for a basic online banking application."

ChatGPT: "Sure! Here's an example user story: As a customer, I want to be able to view my account balance so that I can keep track of my finances. Another user story could be: As a customer, I want to transfer funds between my accounts for convenient money management. Is there anything specific you would like to focus on?"

Student: "Can you generate a user story related to account creation for new customers?"

ChatGPT: "Certainly! How about this user story: As a new customer, I want to be able to create a bank account online so that I can access banking services conveniently. Please let me know if you need any further details."

By following this process, students can effectively utilize ChatGPT to generate user stories tailored to their software engineering projects. The generated user stories provide a solid foundation for further analysis and modeling steps.

Next, we will discuss the generation of use case diagrams using ChatGPT and the conversion of generated code into diagrams using tools like PlantUML and draw.io.

Formulating Use Case Descriptions:

- Begin by engaging with ChatGPT and providing it with the necessary context, such as the system's purpose and target users.
- Interact with ChatGPT in a conversational manner, describing the potential actions and behaviors of the system's users.
- Prompt ChatGPT for specific use cases and ask it to generate detailed descriptions for each use case.

2. EXTRACTING USE CASE INFORMATION:

- After obtaining the use case descriptions from ChatGPT, extract the relevant information, such as actors, actions,

Authorized licensed use limited to: Hochschul- und Kreisbibliothek Bonn-Rhein-Sieg. Downloaded on April 09,2024 at 16:57:12 UTC from IEEE Xplore. Restrictions apply.

and outcomes.

- Use natural language processing techniques or regular expressions to identify the essential elements within the generated descriptions.

3. Mapping Use Cases to PlantUML Syntax:

- Transform the extracted use case information into the appropriate PlantUML syntax.
- Define actors using the `actor` keyword and use `usecase` to represent the use cases.
- Establish relationships between actors and use cases using the `-->` or `<--` arrows to indicate communication directions.

4. Generating PlantUML Code:

- With the use case information mapped to PlantUML syntax, generate the corresponding PlantUML code using a text editor or a script.
- Ensure the code is structured properly, with proper indentation and clear naming conventions.

5. Converting PlantUML Code into Diagrams:

- Utilize the draw.io site or another PlantUML-compatible tool to convert the generated PlantUML code into use case diagrams.
- Open the draw.io site and select "Create New Diagram."
- Choose the "Arrange" option and select "Insert" -> "Advanced" -> "PlantUML" to access the PlantUML editor.
- Paste the generated PlantUML code into the editor, and the diagram will be rendered automatically.

6. Refining and Iterating:

- Review the generated use case diagrams for accuracy and completeness.
- Make necessary adjustments to the PlantUML code or user story descriptions as needed.
- Repeat the process with ChatGPT if additional use cases or refinements are required. The above steps used to develop use case diagram shown in Figure 1 for the online banking application example.

By leveraging ChatGPT's capabilities and integrating it with the PlantUML syntax, software engineering students can effectively generate use case diagrams. This interactive process not only enhances their understanding of requirements engineering concepts but also provides them with practical experience in translating user stories into visual representations. The resulting use case diagrams serve as valuable artifacts for communicating and refining system requirements throughout the software development lifecycle.

B .ANALYSIS MODELING

Analysis modeling is a crucial step in the requirements engineering process, focusing on capturing the structure and behavior of the system. In this section, we will describe how ChatGPT can be leveraged to develop class diagrams and sequence diagrams, enabling students to gain a deeper understanding of the system's design and interactions.

- *Developing Class Diagrams using ChatGPT:*

To develop class diagrams using ChatGPT, students interact with the chatbot by providing information about the system's entities, attributes, and their relationships. The conversation with ChatGPT can proceed as follows:

1. Student: "Please help me develop a class diagram for the system."
2. ChatGPT: "Sure! Let's start by identifying the main entities in the system. What are the key components?"
3. Student: "The system has three main entities: Customer, Account, and OnlineBanking."
4. ChatGPT: "Great! Now let's define the attributes for each entity. What attributes are associated with the Customer entity?"
5. Student: "The Customer entity has attributes like customer ID, name, email, and password."
6. ChatGPT: "Understood. What about the Account entity?"
7. Student: "The Account entity has attributes such as accountId, accountType, and balance."
8. ChatGPT: "Excellent. Finally, what attributes are associated with the OnlineBanking entity?"
9. Student: "The OnlineBanking entity doesn't have any specific attributes in this context, as it represents the overall system."
10. ChatGPT: "Thanks for details. Let's generate the usecase diagram. Here is the corresponding code..."

```
@startuml

left to right
directionactor
Customer

rectangle OnlineBanking {

 Customer -- > (View Account
 Balance)Customer -- > (Transfer
 Funds) Customer -- > (Create Bank
 Account)

}

@enduml
```

Based on this conversation, ChatGPT can generate a class diagram. The resulting class diagram provides a visual representation of the system's structure and relationships. It is important to note that the complexity of the class diagram can be adjusted based on the specific requirements of the educational context.

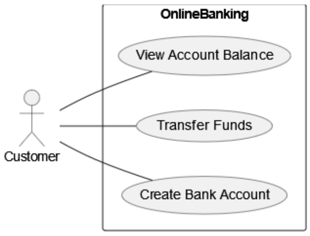

Fig. 2. use case diagram created by ChatGPT for the illustrated example

Authorized licensed use limited to: Hochschul- und Kreisbibliothek Bonn-Rhein-Sieg. Downloaded on April 09,2024 at 16:57:12 UTC from IEEE Xplore. Restrictions apply.

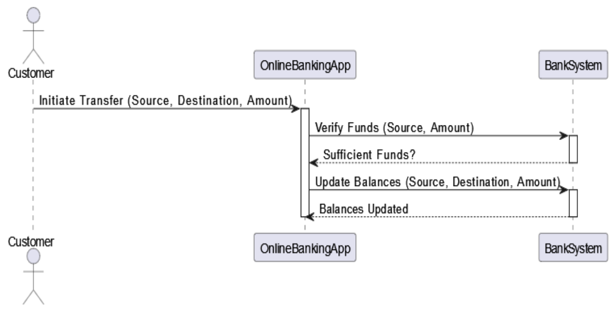
Customer
OnlineBankingApp
BankSystem
Initiate Transfer (Source, Destination, Amount)
Verify Funds (Source, Amount)
Sufficient Funds?
Update Balances (Source, Destination, Amount)
Balances Updated
Customer
OnlineBankingApp
BankSystem

Fig. 3. sequence diagram created by ChatGPT for illustrated example

- *Creating Sequence Diagrams using ChatGPT and PlantUML:*

Sequence diagrams are essential for illustrating the interactions between objects in a system. By leveraging ChatGPT and PlantUML, students can generate sequence diagrams that depict the flow of messages between objects.

1. Student: "I need assistance in creating a sequence diagram for the 'Transfer Funds' functionality."
2. ChatGPT: "Certainly! Could you please describe the steps involved in the 'Transfer Funds' process?"
3. Student: "Sure. First, the customer initiates the transfer by providing the source and destination accounts along with the transfer amount. Then, the system verifies the availability of sufficient funds in the source account. Finally, the system updates account balances and records the transaction."
4. ChatGPT: "Thanks for details. Here is the corresponding code..."

```
@startuml

actor Customer as C

participant OnlineBankingApp as App
participant BankSystem as B

C -> App: Initiate Transfer (Source, Destination,Amount)

activate App

App -> B: Verify Funds (Source, Amount)activate
B

B --> App: Sufficient Funds?
deactivate B

App -> B: Update Balances (Source, Destination,Amount)

activate B

B --> App: Balances Updated
deactivate B

deactivate App@enduml
```

The generated code can then be copied and pasted into the draw.io site or any PlantUML-compatible tool to obtain the visual representation of the sequence diagram. The resulting diagram showcases the specific interactions and the order in which objects communicate during the "Transfer Funds" process.

By involving ChatGPT in the analysis modeling phase, students can gain hands-on experience in developing class diagrams and sequence diagrams. The interactive nature of the learning environment facilitates a deeper understanding of the system's design, allowing students to visualize the relationships and interactions between different components. In conclusion, the integration of ChatGPT into the analysis modeling phase provides an innovative approach to software engineering education. The interactive learning experience empowers students to actively participate in the requirements engineering process, improving their comprehension of class diagrams and sequence diagrams while reinforcing their knowledge of system design principles.

IV. DISCUSSION

The integration of ChatGPT in the software engineering education process creates an interactive learning environment that fosters engagement and enhances students' understanding of requirements engineering. This section highlights the benefits and experiences derived from the interactive learning experience between students and ChatGPT.

A. ENHANCED UNDERSTANDING OF REQUIREMENTS ENGINEERING CONCEPTS:

ChatGPT aids software engineering students in actively participating in the requirements engineering process, offering instant feedback and guidance on various aspects. This interaction promotes a more profound comprehension of essential concepts and methodologies.

B. REAL-WORLD EXAMPLES AND PRACTICAL APPLICATIONS:

ChatGPT offers real-world examples and applications of requirements engineering, bridging the gap between theoretical knowledge and practical implementation. It enables students to understand the relevance of requirements engineering in actual software projects.

C. ITERATIVE LEARNING AND AGILE PROCESS:

ChatGPT encourages iterative learning and an agile approach in students, aligning with industry practices. It helps students understand the evolving nature of requirements in software projects and cultivate adaptability.

D. INSTANT FEEDBACK AND GUIDED LEARNING:

Serving as a virtual mentor, ChatGPT provides immediate feedback and guidance, fostering a supportive learning environment. This interactive feedback loop helps students refine their skills in requirements engineering.

E. STUDENT ENGAGEMENT AND MOTIVATION:

ChatGPT boosts student engagement and motivation by creating a dynamic learning environment. It’s immediate responses and personalized assistance kindle curiosity and exploration in students, motivating them to delve deeper into requirements engineering principles.

F. PEER COLLABORATION AND KNOWLEDGE SHARING:

The interactive learning environment facilitated by ChatGPT enhances peer collaboration and knowledge sharing, refining students' critical thinking, problem-solving skills, and communication abilities.

G. FEEDBACK AND EXPERIENCES:

Feedback is gathered from students to evaluate the effectiveness of the interactive learning experience. The data provides insights into the utility of ChatGPT as a learning tool and its impact on students' understanding of requirements engineering.

In conclusion, ChatGPT revolutionizes software engineering education by enabling an interactive learning environment.

Authorized licensed use limited to: Hochschul- und Kreisbibliothek Bonn-Rhein-Sieg. Downloaded on April 09,2024 at 16:57:12 UTC from IEEE Xplore. Restrictions apply.

This approach results in a comprehensive educational experience, preparing the next generation of software engineers who are proficient in requirements engineering practices.

V. CONCLUSION AND FUTURE WORK

This study introduced an interactive learning platform for software engineering education, using the capabilities of ChatGPT to streamlining the requirements engineering process. We presented a step-by-step approach, showing students how to interact with ChatGPT to create user stories, use case diagrams, class diagrams, and sequence diagrams. The platform's simplicity and agility allow students to comprehend the critical concepts of requirements engineering using a limited number of user stories. The study showcased the practical application of ChatGPT in real-world software engineering contexts, highlighting the importance of these artifacts in capturing software requirements. The interactive learning experience proved to be invaluable, promoting active student engagement, knowledge retention, and an enhanced understanding of the software engineering process. Our study paves the way for further research into AI integration in education, including the potential use of advanced natural language processing techniques and the application of ChatGPT in other areas of software engineering. Overall, this study confirms that integrating AI chatbots like ChatGPT into software engineering education is a practical and efficient method to help students understand and model software requirements, making this technology increasingly relevant in an evolving field.

The creation of an interactive educational platform using ChatGPT for instruction in software engineering is a significant leap in the utilization of AI to bolster the process of requirements engineering. Nevertheless, there exist numerous areas that invite further research and refinement. We'll delve into some promising future paths and enhancements that can build on the groundwork established in this paper.

1. Improved Natural Language Comprehension: Even though ChatGPT has excellent language creation abilities, further refining its understanding of natural language can enhance the user stories it creates. Training ChatGPT with specific software engineering datasets or merging external knowledge bases can improve its understanding of software engineering principles for more accurate results.

2. Verification and Assessment: There's a need to empirically test the effectiveness of the interactive learning platform. This involves collecting data and conducting experiments to measure the impact of ChatGPT in software engineering education. It's crucial to compare student performance with traditional teaching methods and seek feedback from learners and teachers for system improvement.

3. Synchronization with Cooperative Tools: Combining the interactive learning platform with tools that foster teamwork enrich learning experience. This enables learners to work on projects, with ChatGPT helping in requirements engineering discussions, document collaboration, and task organization.

4. Augmentation of Knowledge Base: Expanding ChatGPT's knowledge base to include more software engineering concepts, strategies, and best practices can enhance its utility as an educational tool. This can be achieved by integrating selected domain-specific databases, educational materials, and research papers.

5. Immediate Feedback and Customization: Future developments could include immediate feedback and customization options, tailoring the learning environment to the learners. Teaching ChatGPT based on learner interactions can allow it to tailor its responses and adapt to individual learning styles. Also, using sentiment analysis or user feedback tools can help identify areas needing more assistance or clarification.

6. Synchronization with Professional Tools: Software engineering often requires professional tools. Integrating ChatGPT with popular software development tools and platforms can make it more functional. This can provide a comprehensive learning experience that closely simulates real-world software development processes.

7. Expandability & Implementation: A critical future consideration is the platform's expansion. Creating user-friendly interfaces or plugins for easy implementation of ChatGPT across different educational settings can encourage wider acceptance and use.

In conclusion, the study provides a robust foundation for an engaging learning platform using ChatGPT for software engineering education. Many opportunities exist for further research and refinement, and the goal is to capitalize on AI technologies to transform software engineering education.

REFERENCES

[1] Pressman, R. S. (2014). Software Engineering: A Practitioner's Approach. McGraw-Hill Education.

[2] Sommerville, I. (2016). Software Engineering. Pearson Education

[3] Data Retrieved from ChatGPT. (n.d.). OpenAI.

[4] Leffingwell, D., & Widrig, D. (2003). Managing Software Requirements: A Use Case Approach. Addison-Wesley Professional.

[5] Wiegers, K., & Beatty, J. (2013). Software Requirements, 3rd Edition. Microsoft.

[6] Cockburn, A. (2000). Writing Effective Use Cases. Addison-Wesley Professional.

[7] Data Retrieved from PlantUML. (n.d.). PlantUML.

[8] Wiegers, K. (2003). More About Software Requirements: Thorny Issues and Practical Advice. Microsoft Press.

[9] Alexander, I., & Maiden, N. (2004). Scenarios, stories, use cases: Through the systems development life-cycle. Wiley.

[10] Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1995). Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley Professional.

[11] Jacobson, I., Booch, G., & Rumbaugh, J. (1999). The Unified Modeling Language User Guide. Addison-Wesley Professional.

[12] Data Retrieved from Draw.io. (n.d.). Draw.io.

[13] Sutcliffe, A., & Maiden, N. (2000). Supporting Scenario-Based Requirements Engineering. IEEE Software, 17(6), 48-57.

Authorized licensed use limited to: Hochschul- und Kreisbibliothek Bonn-Rhein-Sieg. Downloaded on April 09,2024 at 16:57:12 UTC from IEEE Xplore. Restrictions apply.