SOFTWARE REQUIREMENT SPECIFICATION (VERSION 1.0)

GPT-Assisted Coding Trainer

Software System for Training Coding with GPT Assistance

May 2023

Due Dates

	Scope of AD	Due Date	Length Limit
Interim Report	**Chapters 1, 2, 3**	**6/07(Wed), 9pm**	**20± Pages**
Pre-Final Report	Chapters 4, 5	6/20(Tue), 9pm	50± Pages
Final Report	Chapters 6, 7	7/03(Mon), 9pm	70± Pages

CEP Instructor

Soo Dong Kim, Ph.D.
Professor, School of Software
Soongsil University, Seoul, Korea
sdkim777@gmail.com, https://soodongkim.net

삼성전자 첨단기술아카데미

Name CEP Reports To: **AA_A3차_인증과제_Interim_홍길동**

Submit CEP Reports To: **s777kim@naver.com**

TABLE OF CONTENTS

1. Purpose of the Document

The purpose of this document is to specify the requirement for developing the target system in this CEP. The requirement will become the basis for designing the software architecture of a target system, which is required as a fulfillment to acquire the Samsung Associate Architect (AA) certification.

2. Comprehensive Evaluation Project (CEP)

2.1. About Associate Architect (AA) Program

The Associate Architect Program of Samsung Electronics is to provide participants with two sets of software architecture design proficiency.

- **Body of Knowledge (BoK) on Software Architecture**

  This set includes the fundamental theories and methods for designing SW architecture.

- **Skillset for Designing SW Architecture**

  This set includes the practical skill for applying architecture design methods to a given SRS.

CEP is designed to fulfill the second set of AA program through an individual design project.

2.2. How is the CEP problem prepared?

The CEP problem is prepared by the instructor, based on the following principles.

- **Principle 1. Utilizing the Whole BOK of Software Architecture Design**
  - Utilizing Architecture Styles
  - Designing Architecture for multiple Views
  - Designing Architecture for Non-Functional Requirements
- **Principle 2. Handling the Complexity of Industrial Systems**

  The target system to design in CEP is an industry-level complex software system, i.e., not an academic problem appeared in books or literature.
- **Principle 3. Solution Not Available in Public**

  CEP problem is not a reproduction of already existing exercise problem in books, and hence the architecture design solution for the CEP problem is not available in public.

2.3. Architecture Design Reports in CEP

Each participant designs and submits the design of software architecture for the target system in an incremental manner.

- Interim Report
  This report includes the context analysis model and the skeleton architecture design of the target system.
- Prefinal Report
  This report includes the architecture design for multiple views of the system: functional view, information view, behavior view, and deployment view.
- Final Report
  This report includes the architecture design for non-functional requirements and the validation of the architecture design.

❑ Weight Distribution of CEP Reports

Interim Report	Prefinal Report	Final Report	TOTAL
25 points	35 points	40 points	**100 points**

2.4. Template for Architecture Design

❑ Template for CEP Reports

A template in Microsoft Word is available for designing the software architecture in CEP, allowing participants to effectively specify their architecture design.

❑ Alignment with Architecture Design Methodology

The template has been carefully developed to align with the architecture design methodology recommended by the instructor, ensuring consistency and coherence throughout the process.

By using this template, participants can streamline their architectural decisions and create robust and well-structured CEP reports.

3. Overview of the Target System

3.1. What is GPT?

- **GPT (Generative Pre-trained Transformer)**

  GPT refers to a type of artificial intelligence model developed by OpenAI. GPT models are based on the Transformer architecture, which is a deep learning model architecture known for its ability to handle sequential data efficiently.

  GPT models are renowned for their ability to generate coherent and contextually relevant text. They can be used for various natural language processing tasks, including language translation, text summarization, chatbot interactions, and creative writing assistance.

  - Reinforcement Learning from Human Feedback
    Human reviews are used to fine-tune the system in Reinforcement Learning.

- **GPT 4 (Released on March 14, 2023)**

  - Multimodal Model
    GPT-4 is a large multimodal model, which can accept both text and image inputs, called GPT prompts, and output human-like text. For example, it can describe the humor in unusual images, summarize text from screenshots, and answer exam questions that contain UML diagrams.

  - Complexity of GPT-4 Models
    GPT-4 is more reliable, creative, and able to handle much more nuanced instructions than GPT-3.5.

    Two versions of GPT-4 have context windows of 8,192 and 32,768 tokens, where GPT-3.5 is limited to 4,096 and GPT-3 limited to 2,049 tokens.

    Tokens are the basic unit that OpenAI GPT models use to compute the length of a text. They are groups of characters, which sometimes align with words, but not always. In particular, it depends on the number of characters and includes punctuation signs or emojis.

  - System Message
    A directive in natural language given to GPT-4 in order to specify its tone of voice and task.

3.2. What is GPT-Assisted Coding Trainer?

Coding Trainer is a software system designed to facilitate the teaching of programming languages without the need for human instructors.

GPT-Assisted Coding Trainer is an advanced coding trainer that harnesses the power of GPT-4 to deliver its core functionality. The system leverages the capabilities of the GPT-4 model through its API to perform essential tasks and provide a comprehensive learning experience.

The GPT-Assisted Coding Trainer takes advantage of the latest advancements in GPT language modeling and artificial intelligence to create an immersive and effective learning environment for programming languages.

3.3. Process of Training Coding Skill

Once a target programming language is chosen, the training is conducted in the following order:

- **Step 1: Learning the Foundation of current Unit.**

  The system begins by teaching the basics, i.e., the syntax and usage of the chosen programming language. This step focuses on providing learners with a solid understanding of the current unit.

- **Step 2: Solving Coding Exercises.**

  After teaching the foundation, the system generates coding exercises specifically tailored to the given topics in the unit. These exercises are designed to provide practical application opportunities for learners to reinforce their understanding and improve their coding skills.

- **Step 3: Evaluating Program Codes.**

  Learners submit their program codes for evaluation. The system assesses the submitted code, checking for correctness, adherence to coding conventions, efficiency, and other relevant criteria.

  The evaluation is given in both quantitative and qualitative ways.

- **Step 4: Repeat Steps 1 through 3:**

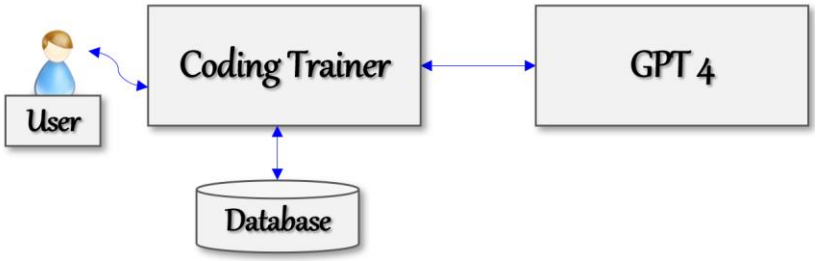
  The process of (1) Learning the Foundation of current, (2) Solving Coding Exercises, and (3) Evaluating Program Codes is repeated for each topic and unit of the language until the learner completes all the learning units.

  This iterative approach allows learners to gradually progress through the language's concepts, continually reinforcing their knowledge and skills.

The overall goal is for learners to successfully complete all learning units of the chosen language by passing the coding exercises, demonstrating their understanding and proficiency in the programming language.

3.4. Deployment of the System

The system is deployed as a web server, which subscribes to GPT 4, as shown in the figure.



A user interacts only with the Coding Trainer, not directly with GPT 4. The system maintains a database for managing the essential datasets such as user profiles, programming language profile, sessions for using the system including teaching contents, exercise problems, users' program codes, and their evaluations.

4. Functional Requirements

The functionality of the system is organized as a set of functional categories.

4.1. Registering Learner

This functionality is designed to manage the profiles of individuals who are interested in learning programming through this system. The *Learner Profile* includes the following attributes: name, official identification, address, phone, email, and affiliation. Learners also register their login ID and password.

4.2. Registering Director

This functionality is designed to manage the profiles of course directors who are responsible for managing the course contents and monitoring learners' activities. The *Director Profile* includes the following attributes: name, official identification, director identification, address, phone, email, department, specialty, and courses managed. Directors also register their login ID and password.

4.3. Registering Language Profile

This functionality is designed to manage the profiles of programming languages offered by the system for learners, including popular languages such as C, C++, Java, and Python. A *Language Profile* serves as a meta-description of each programming language and is defined by the course director.

A language profile includes the following attributes:

- Profile Name
- List of Units in the programming language
  The content of a programming language is organized into units, with each unit representing a key topic or concept of the language. These units cover various aspects,

including control structures, inheritance, and other fundamental elements of the language.

- List of Topics for each Unit

  The content of a unit is further organized into topics, with each topic representing a specific language construct related to the unit's main theme. For example, within the unit on Control Structure, topics could include the for loop, while loop, and repeat-until loop. These topics delve into the details of each language construct, providing learners with a comprehensive understanding of the unit's content.

Below is an example of a language profile:

- Language Profile ID: Java_Level_1
- Programming Language: Java
- Units and Topics (Organized in a table)

Unit	Title of Unit	Topics in Unit
1	Introduction to Java	○ Overview of Java programming language ○ Installing Java Development Kit (JDK) ○ Installing Integrated Development Environment (IDE) ○ Writing and running a simple Java program
2	Variables and Data Types	○ Primitive data types (e.g., int, double, boolean) ○ Variable declaration and initialization ○ Type casting and conversion ○ String manipulation
3	Control Structures	○ Conditional statements (if, else if, else) ○ Loops (for, while, do-while) ○ Switch statements
4	Array	○ Declaration, initialization, and manipulation of arrays ○ Array traversal and manipulation ○ Multidimensional arrays
5	Encapsulation, Information Hiding, Object, and Class	○ Encapsulation ○ Information Hiding ○ Object ○ Class ○ Constructor and Destructor ○ Static ○ Method Overloading
6	Inheritance and Polymorphism	○ Inheritance and Subclassing ○ Polymorphism ○ Method Overriding ○ Object Substitutability ○ Dynamic Binding

7	Exception Handling	○ Exception as Objects ○ Handling and throwing Exceptions ○ Try-Catch Block ○ Exception hierarchy ○ Checked vs. Unchecked Exceptions
8	File Handling	○ Reading from and writing to files ○ File input/output operations
9	Collections	○ ArrayList ○ LinkedList ○ Other Collection Classes ○ Iteration and manipulation of Collections ○ Map and Set interfaces
10	Java GUI	○ Basics of GUI programming using Swing or JavaFX ○ Event-driven programming ○ Creating and handling GUI components

- **Importance of 'Language Profile'**

  The comprehensiveness and accuracy of language profiles are of utmost importance as they serve as the foundation for the training provided by the system and the learning process for the learners.

  - Coverage of Teaching

    Language profiles are meticulously designed to ensure that all essential aspects of a programming language are covered in a logical order. By following the units as outlined in the language profile, learners can acquire knowledge and skills in a structured manner, building upon previously learned concepts and progressing towards more advanced topics.

  - Sequence of Teaching

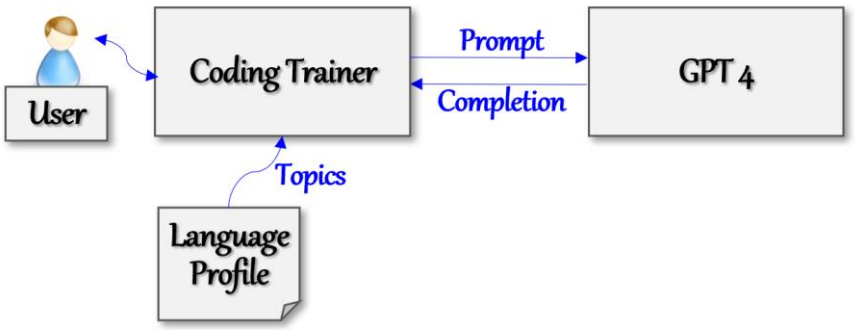
    The system follows the sequence of units outlined in the language profile to teach coding effectively. Hence, the sequence of units should be logically defined for effective learning.

  - Basis for Constructing GPT Prompts

    Furthermore, the language profile forms the basis for generating the "GPT Prompts," which are specifically designed questions used to interact with the GPT model. These prompts are constructed by aligning them with the topics and concepts outlined in the language profile.

    By leveraging the language profile, the system creates GPT prompts that are relevant to the specific programming language and the corresponding learning objectives. These prompts serve as input to the GPT model, enabling the system to generate accurate and contextually appropriate responses.

  The interaction between the Coding Trainer and the GPT is shown in the following figure.



The Coding Trainer reads the current language profile, retrieves the topics of the on-going unit, generates GPT prompts based on the topics, send it to the GPT for a textual answer, called *Completion*. Then, the completion is reformatted and aligned for the current learner.

4.4. Teaching the Foundation

This functionality is to deliver comprehensive instructions for teaching the fundamental concepts of the current unit. Each unit has predefined topics outlined in the language profile, and the system generates instructions for learners in the specified order of topics.

- **Contents of Instruction for each Topic**

  The system generates instructions for learners in the following order.

  - Introduction

    An initial overview or explanation that introduces the topic and its significance within the programming language. Especially, the usage of the language constructs is clearly explained. This helps learners understand the practical application and purpose of these constructs.

    Associated Libraries: If applicable, information regarding relevant class libraries or modules related to the topic. This highlights additional resources and functionalities available to learners.

  - Syntax

    Detailed explanations of the syntax and structure of the language constructs. This ensures learners grasp the correct syntax and usage guidelines.

  - Examples

    Illustrative code snippets and examples that demonstrate the usage of the language constructs. Learners can refer to these examples to better understand how to implement and utilize the constructs.

  By following this structured approach, learners receive comprehensive instruction, covering all essential aspects of the language constructs within each topic.

- **Consulting GPT**

  The system generates these instructions by leveraging the knowledge and capabilities of

GPT (Generative Pre-trained Transformer). GPT is consulted to assist in generating clear and concise instructions for each topic within the unit.

- **Interactive Learning**

  The system provides an interactive learning experience that emulates in-class instruction. Learners have the opportunity to engage with the system by typing questions or queries, and the system responds with appropriate and accurate answers.

4.5. Generating Coding Exercises

This functionality is to dynamically generate a personalized set of coding exercise problems for the current topic and learner. By leveraging the power of GPT, the system can dynamically generate exercise problems that cater to the specific context and individual needs of the learner.

Using the GPT model, the system analyzes the current topic and learner's profile to generate exercise problems that are relevant, engaging, and appropriately challenging. The GPT model's natural language processing capabilities allow the system to understand the learner's level of proficiency, preferred learning style, and specific areas that require focus.

By consulting GPT, the system generates exercise problems that align with the learning objectives, ensuring the learner has opportunities to practice and apply the concepts covered in the current topic.

4.6. Submitting Exercise Solutions

This functionality enables learners to submit their solutions to the exercise programs, specifically in the form of their program codes. Upon completion of the exercise, learners can submit their program codes through the system's interface. This allows them to showcase their understanding of the concepts taught and demonstrate their coding skills.

4.7. Evaluating Program Codes

This functionality is to evaluate program codes submitted by learners and provide them with comprehensive feedback on their submissions. The evaluation is based on the criteria of correctness, efficiency, adherence to coding standards, and other relevant criteria.

The system assesses the submitted code by consulting the GPT. The system generates both quantitative and qualitative evaluations.

- **Quantitative Evaluation**

  It offers a quantitative evaluation of user submissions, utilizing a rating scale ranging from 1 to 10. This evaluation provides a measurable assessment of the quality or performance of the submitted code.

❑ **Qualitative Evaluation**

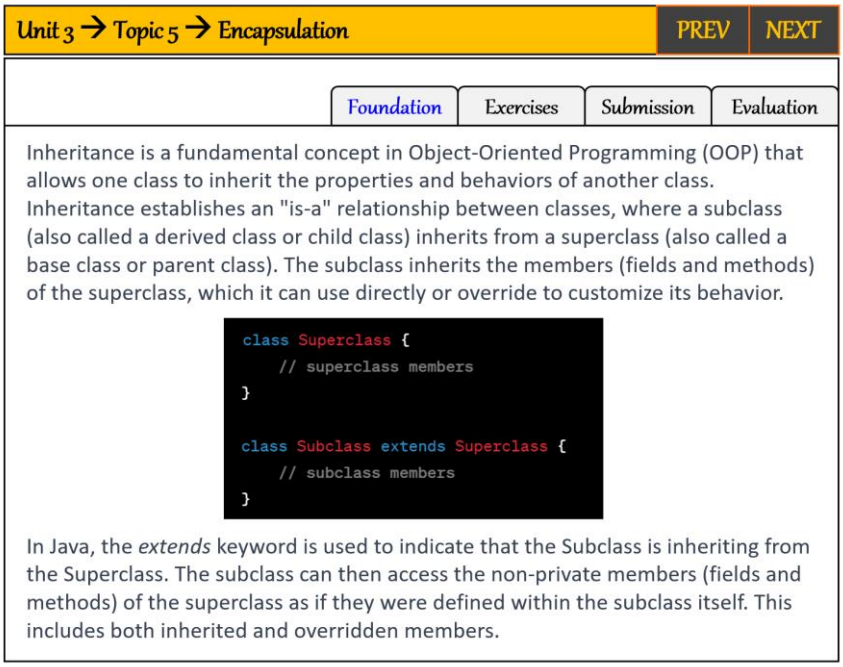
The system goes beyond quantitative evaluation and provides comprehensive feedback on user submissions. It offers a detailed analysis in text format that highlights the strengths and weaknesses of the submitted code. The feedback includes explanations for necessary corrections, suggestions for improvement, and alternative approaches to writing the code. This thorough feedback helps users understand their mistakes, learn from them, and enhance their coding skills effectively.

4.8. Conducting the Instruction Process

This functionality is to execute the main control flow, which triggers the appropriate functions to perform the steps 1 through 3 as described in section 3.3. It utilizes a loop to execute all the learning steps iteratively until a learner successfully completes the entire training session.

Conversely, the system provides learners with the option to select and learn a specific unit from the menu without requiring them to complete its preceding units beforehand.

The user interface for conducting the learning should look like the following.



4.9. Generating Training Reports

This functionality is to generate comprehensive reports on the learning progress of learners and certificates of completion.

❑ **Generating Progress Report**

This progress report provides comprehensive details of the training sessions conducted. It encompasses the complete history of training sessions conducted for the specific programming language, highlighting the exercises undertaken and the evaluation results

of code submissions.

- **Generating Certificate of Completion**

  This certificate confirms that the recipient has successfully completed all the required training units and fulfilled the necessary criteria for certification. It is a concise, one-page document that bears the official seal of the training institute.

5. Non-Functional Requirements

In the target system, there are multiple non-functional requirements that are crucial. However, for the purpose of CEP, we focus specifically on two essential NFR items, which are outlined below.

5.1. NFR-1. High Reliability of the System

The target system provides services potentially to a large number of users, and hence the system should be designed to provide high-level reliability.

By emphasizing reliability in the design process, the system can effectively mitigate potential failures, minimize downtime, handles various faults, recovering efficiently from failures, and enhance the overall user experience.

Reliability in ISO 9126 is defined with three sub-quality attributes, and they should be satisfied by the system.

- **Maturity**
- **Fault Tolerant**
- **Recoverability**

5.2. NFR-2. High Personalization of Instruction and Coding Exercises

Learning can be maximally effective when training is delivered in a personalized manner. To ensure this, the system should be designed to offer a high level of personalization across three key aspects:

- **Personalizing the Instruction**

  The instructions for explaining the language constructs should be customized for the level of details, the level of compactness, and types of examples for each learner.

- **Personalizing the Coding Exercise**

  The coding exercise problems should be designed with consideration for the proficiency level of each learner, moving away from a 'one-size-fits-all' approach. Instead, the problems should be generalized to accommodate learners at different skill levels.

  By considering the proficiency level of learners, the system can generate exercise problems that align with their current abilities and knowledge. This ensures that learners are appropriately challenged without feeling overwhelmed or bored.

6. Guidelines for Conducting CEP

6.1. Guidelines for Designing the Architecture

Apply the following guidelines for writing CEP Reports.

- **Conformance to the given SRS**

  The submitted AD should conform to the given SRS.

- **Conformance to UML Standards**

  The submitted AD should conform to the notational and usage standards of UML.

- **Consistency among various Artifacts in AD**

  There should be a high consistency among various artifacts (such as diagrams) in the submitted AD.

- **Comprehensibility of Textual Description**

  The textual elaboration of the architecture design should be written in accurate, precise, and condensed way. Hence, the understandability of the AD becomes high. The textual description can be written in English, Korean, or their mixture.

- **Readability of Figures and Tables**

  The figures and tables should be easily readable by applying good formats, right font size and special effects on them. For example, a use case diagram with 100 use cases should be well structured and enlarged if needed.

- **Reasonable Details of Machine Learning design**

  The submitted AD would include a design for managing machine learning models. The description of the machine learning model generation should be written in reasonably details. The description typically includes machine learning algorithms utilized, training sets used, the details for designing the model generation components.

- **Originality of the AD**

  The submitted AD should be an individual work. Any same or highly similar solutions would get a score penalty.

6.2. Guidelines for Submitting Reports

Apply the following guidelines for writing CEP Reports.

- **Due Dates for Submission**

  The due dates and times for each CEP report are specified. The CEP reports should be submitted by the due date and time. Late submissions of CEP reports are not accepted.

- **Format of the CEP Report**

  Use the word processor, MS Word, for formatting your CEP reports. Submit the word files, not the PDF files.

- **Submission**

  Submit your CEP reports as instructed by the course manager.

6.3. Evaluation Form for CEP Reports

The following form is used to evaluate the CEP report.

Evaluation of CEP Report

Name: 홍길동

Criteria	Max	Earned	
Ch.1, Introduction	2		
Ch.2, (A1) Architectural Requirement Refinement	3		
Ch.3, (A2) System Context Analysis			
System Boundary Context	5		
Functional Context	5		
Information Context	5		
Behavior Context	5		25
Ch.4, (A3) Skeleton Architecture Design			
Justification of Candidate Architecture Styles	6		
Integrating Selected Styles into Architecture	4		
Ch.5, (A4) Architecture with Views			
Applying Functional Viewpoint	7		
Applying Information Viewpoint	7		
Applying Behavior Viewpoint	7		
Applying Deployment Viewpoint	4		35
Ch.6, (A5) Architecture with Quality-driven Design			
Design for NFR #1, Applying Process	5		
Design for NFR #1, Quality Delivered	10		
Design for NFR #2, Applying Process	5		
Design for NFR #2, Quality Delivered	10		
Ch.7, (A6) Architecture Evaluation			
Intermediate Steps of applying Evaluation	5		
Correctness of the Evaluation	5		40
CEP Score	100		