Introduction

This week's learning centered around software design patterns and information security. I explored the Publisher-Subscriber pattern, along with other behavioral and structural patterns such as Command, State, Decorator, and Proxy. In addition, I learned the foundational principles of cryptography and broader information security techniques. These topics offered valuable insight into writing more maintainable, extensible, and secure software systems. Through readings, quizzes, and discussion activities, I gained both conceptual understanding and practical insights that can be directly applied in real-world software development.

Difficulties Faced

One of the challenges I faced was grasping the nuances between similar design patterns. For instance, understanding when to apply the State pattern versus the Strategy pattern required careful analysis of their intent and usage context. Additionally, cryptography concepts such as symmetric versus asymmetric encryption and hashing functions involved complex mathematical underpinnings that required revisiting foundational materials multiple times. Taking the self-quiz and graded quiz helped reinforce these topics, though it took time to process the full implications of each concept.

Activities Performed

This week, I actively participated in a discussion post where I applied the Publish-Subscribe (Observer) pattern to an online auction scenario from Marsic (2012). I identified ItemInfo as the publisher and BuyerInfo as the subscriber, detailing how these classes could interact using interfaces to promote decoupling. I suggested that ItemInfo manage a list of subscribers and notify them of auction events, such as new bids or auction closures. This exercise helped me internalize the purpose of the Observer pattern, especially how it improves scalability and separation of concerns in software systems (Marsic, 2012). In addition to the discussion post, I completed both the self-quiz and graded quiz, which provided practical scenarios for evaluating design choices and applying cryptographic principles.

Conclusion

Design Patterns Some of this week’s lessons stressed the magnitude of design patterns for having reusable solutions for common software design issues. Now this was also starting to make sense, the Observer pattern demonstrated the value of breaking down components into smaller, easily replaceable parts. The author also learned a lot more about computer security, so often neglected yet increasingly important with security, secure communications and indeed encryption, all featuring frequently in the story. In the world of software-security the scale is increasingly important and hand in hand with that it becomes indispensable. One key idea that keeps coming back to me is the importance of designing systems not only to function, but to function in the face of flexibility and resilience to future changes and threats.

References

Marsic, I. (2012). *Software Engineering*. Rutgers: The State University of New Jersey. https://my.uopeople.edu/pluginfile.php/57436/mod_book/chapter/46513/CS4403MarsicTextbook.pdf

University of the People. (n.d.). *CS4403 Software Engineering – Unit 7 Learning Guide*. https://my.uopeople.edu/

Wordcount: 409