Discussion-3 The key principles of software design

Discussion Topic:

Please choose one of the following questions to discuss in your initial post:

What are the key principles of software design?
How can software metrics be used to improve the quality of software design?
What are the benefits of using use cases to design and document software systems?

My Post:

Hello Class,

For this discussion, I chose to discuss “What are the key principles of software design?”

The principles of software design can be defined as high-level guidelines “that help developers create cohesive and maintainable software systems” (Institute of Data, 2023). In other words, the principles of software design serve as developers' “architect's compass,” guiding their decisions on how to build software applications that are robust, adaptable, scalable, manageable, and maintainable.
The key principles of software, in the context of this course's final project, can be defined as follows:

- Be a broad view of the system.
- Be linked to specifications.
- Not reinvent the wheel.
- Be efficient and functional.
- Be uniform and integrated. Modular elements should show a consistent theme.
- Be structured to accommodate change.
- Be resilient and responsive to aberrant data inputs, events, or operating conditions.
- Be reviewed to minimize conceptual errors. In this course, use the instructor's feedback.

(CSU Global, 2025)

These principles should not be applied as isolated, standalone guidelines -> rules, meaning that they are connected and should be applied as an interconnected, flexible set of principles that influence each other and depend on one another.
Other, more general principles that are related to the previous set of principles are:

- Don't Repeat Yourself (DRY, reduce repetition in your code by using modularization.
- Keep It Simple, Stupid (KISS), systems work best if they are kept simple rather than complicated (Rau. 2024)
- You Aren't Gonna Need It (YAGNI), implement features when they are actually needed or do not add functionality based on speculation about future needs that are not fully defined.
- Principle of Least Astonishment (POLA), code should behave in a way that least surprises the users (Saafan, 2024).

In addition to these principles, your solution should be modular and consider the system modularity cohesion (the degree to which a module performs one and only one function) and coupling (the degree to which a module is connected to other modules in a system) (CSU Global, 2025). Furthermore, your solution should incorporate information hiding (e.g., functionality - algorithms, data structures, resource allocations), and a software architecture, such as horizontal partitioning or vertical partitioning, which

are high-level structures that aim to comprise software elements, relations among them, and properties of elements and relations that will make the software easier to test and maintain.

-Alex

References:

CSU Global. (2025). Module 3: Project Planning and Analysis [Interactive Lecture]. Canvas. https://csuglobal.instructure.com/courses/110425/pages/module-3-overview?module_item_id=5733223

Institute of Data (2023, June 19). *Software design principles: Creating improved system designs*. Institute of Data https://www.institutedata.com/us/blog/software-design-principles-creating-improved-system-designs/

Rau, R. (2024, December 17). *Clean Code Essentials: YAGNI, KISS, DRY.* DEV. https://dev.to/juniourrau/clean-code-essentials-yagni-kiss-and-dry-in-software-engineering-4i3j

Saafan, A. (2024, August). *Top 5 software design principles for building robust applications*. Nilebits. https://www.nilebits.com/blog/2024/08/software-design-principles-building-applications/