Tools Validation

Daniel R. Cender

Grand Canyon University: CST-235-O500

July 5, 2020

**N-Layer Architecture**

Layered Architecture (also referred to as “N-Layer”) is commonly known by technologists as one of the most widely practiced software patterns by industry teams (Richards, n.d.). It’s been highly mistaken with the N-tier architecture pattern, but they are not entirely similar. While N-tier encompasses an approach to determining which conceptual pieces of an application will run on which machines or threads, N-layer is only concerned with the “logical” separation of application components into same-purpose horizontal levels (Baxi, 2015). Figure 1 shows a diagram of how layered architectures fit together and what technologies can be used to implement that layer in JavaEE.

A close up of a sign

Description automatically generated

*Figure 1.* Representation of a layered software.

Traditionally, there are four layers in the N-Layer architecture: Database, Persistence (“Data Access Layer”), Business Logic, and Presentation. Each layer only needs to know how to retrieve data from the previous layer and pass it forward. The presentation layer only needs to know how to receive data from the business logic layer and format it for presenting to the browser.

This architecture pattern gives the clear benefit of isolation between components. Changes to components in one layer do not warrant much or any changes to any other layers, maximizing the application’s scalability and ease of maintenance. Although four layers is the usual standard for small-to-mid sized applications, many large applications can contain many more layers. Teams may implement a services layer containing utilities that many layers may need to access. In order to maintain the layered architecture standard, certain layers can be defined as “open” and allow requests to pass over them instead of *through* them. This is a handy way to leverage this pattern to avoid an excess of client requests that pass through all layers, yet do not require any business logic or data manipulation from data access to presentation.

Layered software patterning should be considered by large teams, or, teams working on projects which may scale over time. It promotes separation of concerns in applications and divides projects into maintainable, horizontal segments for easier growth and maintenance. As a result of industry trends, the JavaEE stack has grown into a well-suited companion for this pattern.

References

Baxi, S. (2015, July 4). N-Tier / N-Layer Architecture. Retrieved July 05, 2020, from <https://www.linkedin.com/pulse/n-tier-n-layer-architecture-swapnil-baxi/>

Richards, M. (n.d.). Software Architecture Patterns. Retrieved July 05, 2020, from https://www.oreilly.com/library/view/software-architecture-patterns/9781491971437/ch01.html

What is N-Tier Architecture? Examples, Tutorials & More. (2017, May 19). Retrieved July 05, 2020, from <https://stackify.com/n-tier-architecture/>