Global Science Research Journals

Available online at
www.globalscienceresearchjournals.org/

Global Journal of Engineering and Achitecture

Perspective | Open Access

ISSN:2449-1886 Vol. 3 (2), pp. 1-2, August, 2022

Authentication of software architecture is essential for pattern driven design

K Zhou*

Department of Architecture, Louisiana Tech University, Ruston, United States

*Corresponding author. E-mail: zhoukaile678@gmail.com

Received: 03-Jun-2022, Manuscript no: GJEA-22-72224, **Editorial assigned:** 06-Jun-2022, Pre QC no: GJEA-22-72224 (PQ), **Reviewed:** 20-Jun-2022, QC no: GJEA-22-72224, **Revised:** 27-Jun-2022, Manuscript no: GJEA-22-72224 (R), **Published:** 04-Jul-2022, DOI:10.15651/2449-1886.22.3.008

DESCRIPTION

Software architecture is critical to the success or failure of any software system since it deals with the underlying structure, subsystems, and interactions between these components. Software architecting may be thought of as decision-making process software architects examine a variety of potential solutions to a system design issue and select the set regarded to be the best. Software architecture decisions are design decisions that meet system requirements, including functional and quality criteria. We discuss the findings of a Systematic Literature Review (SLR) meant to assist architects in their decision making by linking quality criteria to software patterns in this study. Decisions on software architecture design, such as the selection of architectural patterns and software design patterns, are frequently made early in the software development process of life cycle the following sections identify architectural patterns, styles, and tactics.

Architectural patterns are repeatable and universal solutions to typical software architecture difficulties. Each architectural pattern in software architecture design defines high-level structures and behaviours of software systems while addressing a specific recurring problem in a given context. Architectural patterns are created to satisfy a wide range of functional and quality attribute requirements. In the literature, the phrases "architectural patterns" and "architectural styles" are occasionally used interchangeably since they are basically the same concepts that only differ in their descriptive forms. Software design patterns are tried and true ways that developers employ to solve common problems while developing a software system. A software design pattern is not a complete design that can be easily converted into source or machine code in scope; architectural patterns are similar to software design patterns. In this research, we will concentrate on architectural patterns, which we will refer to as patterns for the purpose of convenience.

Software architecture strategies are design decisions that elevate the importance of specific quality attributes. Older design techniques can have a significant impact on system patterns. In other words, strategies are reusable architectural building blocks that provide broad solutions to questions about quality attributes influenced by patterns. Pattern descriptions give information regarding quality characteristics, and software architects rely on that information to make sound judgments. As a result, increasing such information entails increasing the significance of patterns in fulfilling quality criteria. Patterns and quality characteristics do not exist in isolation and interact heavily. These interactions are known as quality attribute trade-offs. To address quality problems, software architects must select and apply the optimal pattern set. According to some studies, the Layers pattern's Reusability is its strength, but its Scalability is its weakness. If an architect wants both characteristics, she has two choices: choose another pattern or use software architectural tactics to boost Scalability. Software architects make design decisions that have long term implications for the quality of a software-intensive system.

Software architects design the system's architecture, ensure its architectural integrity, analyse technical risks, employ risk-mitigation approaches, participate in project planning, consult with design and implementation teams, and assist with product marketing. As a result, software architects make high level design decisions every day. Every day, software architects engage in creation, perfection, and annihilation processes. Setting standards for developers, creating and implementing new system architectural components, building shells around and interfaces to existing systems, monitoring quality attributes, and even creative destruction to create place for major improvements are all part of their work. Pattern selection is a natural process that takes place during the system architecture process.

Architectural decisions may be influenced by previous decisions, address one or more stakeholder concerns, and should be accompanied by some explanation. The outcome of the decision has an impact on the architecture description. Furthermore, the decision may create new issues. Influence the architecture description Furthermore, the ruling may create new issues. Concerns include both functional requirements and qualitative characteristics. A software architecture pattern encapsulates a fundamental structural organising model. We refer to both concepts as "architectural pattern" because neither has a commonly accepted definition in the literature. An architectural pattern differs from software patterns, also known as design patterns, in that a software pattern solves a broad design problem, whereas an architectural pattern explains the organisational structure of a software system.