UTT
UNIVERSIDAD TECNOLÓGICA DE TIJUANA
GOBIERNO DE BAJA CALIFORNIA

TEMA:

Architecture specification

BY:

Derian Omar Tarango Mendez

GROUP:

10 B

COURSE:

Software development process management

PROFESSOR:

Ray Brunett Parra Galaviz

Tijuana, Baja California, 10 of january 2025

Architecture Specification in Software Engineering

1. Introduction

Architecture specification serves as a blueprint in software engineering, outlining the structure and behavior of a system. It ensures that the system meets functional and non-functional requirements, providing a clear direction for development and deployment.

2. Importance of Architecture Specification

1. **Guides Development**: Acts as a roadmap, ensuring consistency across teams and phases.
2. **Improves Communication**: Aligns stakeholders, developers, and operations teams with a shared understanding.
3. **Risk Mitigation**: Identifies potential issues early, reducing development risks.
4. **Scalability and Maintainability**: Facilitates growth and simplifies future modifications.

3. Components of an Architecture Specification

3.1 System Requirements

- Functional: Describes what the system must do (e.g., user login, data processing).
- Non-functional: Addresses performance, scalability, security, and usability.

3.2 Architectural Design

- High-level diagrams illustrating system structure.
- Interaction between components, such as services, databases, and interfaces.

3.3 Components

- Modules: Defined functionalities or features.
- Data Stores: Relational databases, NoSQL, or cloud storage.
- APIs: Interfaces for communication between components.

3.4 Technology Stack

- Hardware and software technologies used for development and deployment.
- Examples: Frameworks, libraries, cloud platforms.

3.5 Deployment Architecture

- Cloud-based, on-premises, or hybrid models.
- Networking, server configurations, and load balancing.

4. Example: E-Commerce Platform Architecture

4.1 Components

- **Frontend**: React for UI, communicating via REST APIs.
- **Backend**: Node.js and Express for business logic.
- **Database**: MySQL for user data and orders, MongoDB for product catalog.

4.2 Deployment

- AWS with load balancers, RDS, and S3.

4.3 Integration

- Payment gateways like Stripe or PayPal.

5. Best Practices

1. **Document Everything**: Ensure clarity and completeness.
2. **Use Diagrams**: Visualize the architecture for easier understanding.
3. **Prioritize Security**: Define and enforce security measures early.
4. **Iterate and Improve**: Continuously update the architecture as the project evolves.