Architectural Tactics and Knowledge in AXIS2 Project

Group 4: Adarsh Choudhary, Mohit Jain, Surabhi Lone

December 2, 2024

1 Introduction

The goal of this report is to explore architectural design decisions in the AXIS2 project, focusing on the transition of issues from the mailing list to the issue tracker and the use of the Archie tool. The primary objective is to identify and analyze the architectural tactics used to achieve quality attributes like availability, security, and performance, and to understand the architectural knowledge behind design decisions.

Research Questions:

- What tactics or patterns are used to achieve quality attributes (availability, security, performance)?
- How do software engineers implement these tactics or patterns in the AXIS2 project?
- What architectural knowledge concepts influenced the design decisions made during the project?

2 Research Design

This study uses qualitative research methods, analyzing email threads, issue tracker entries, and source code from the AXIS2 project. The following steps were followed for the analysis:

Data Sources:

- **Issues:** Relevant issues in the AXIS2 issue tracker were analyzed to identify architectural decisions.

- **Emails:** Discussions from the mailing list were reviewed to understand the reasoning behind certain architectural tactics.
- **Source Code:** Code implementations were examined to observe how tactics were applied in practice.

Process: The research follows a two-step approach:

- **Step 1: Search and Analyze** – Identify tactics and patterns related to availability, security, and performance from issues, emails, and commits.
- **Step 2: Architectural Knowledge Exploration** – Analyze the architectural knowledge behind design decisions, focusing on trade-offs, assumptions, and rationale.

3 Analysis (Week 1)

In Week 1, the focus was on identifying architectural tactics for the quality attributes of availability, security, and performance. Below is a summary of the tactics identified:

3.1 Tactics for Availability

Category: Availability

Several tactics were identified to ensure the system remains functional and accessible:

- **Fault Detection (Ping, Monitor):**
  - *Ping:* Used to detect faults early by sending ping messages between system components.
  - *Monitoring:* Heartbeat mechanisms were employed to check the health of system components regularly.
- **Redundancy and Failover (Replication, Passive Redundancy):**
  - *Replication:* Redundant components were implemented to ensure that if one component fails, others can take over.
  - *Passive Redundancy:* Spare components were kept ready to take over in case of failure, ensuring minimal downtime.

  - *Active Redundancy:* Multiple components work in parallel, ensuring high availability.

Summary of Tactics:

- Tactic Identified: Fault Detection (Ping, Monitor), Redundancy (Passive and Active Replication)
- Justification: These tactics help detect faults early and ensure the system remains available through redundancy.

3.2 Tactics for Security

While the primary focus in Week 1 was on availability, a few security-related tactics were mentioned:

- **Message Integrity Verification:**
  - Used to ensure that messages exchanged between components remain intact, employing techniques like checksums and hashing.
- **Authentication and Authorization:**
  - Authentication ensures only authorized entities access the system, while authorization ensures that authenticated users are allowed to access the appropriate resources.

3.3 Tactics for Performance

Performance tactics were not as widely discussed in the reviewed data, but some tactics mentioned included:

- **Event Prioritization:**
  - Events were prioritized based on their importance to ensure that high-priority operations are handled first.
- **Concurrency Management:**
  - Multiple operations are processed simultaneously to reduce delays and optimize throughput.

In Week 2, the focus shifted to analyzing maintainability and interoperability, as well as automating processes to aid the analysis. The following work was completed:

4 Week 2 Updates

4.1 Work Done:

- **Issue Identification from Source Code:** Continued efforts to identify and map issues from the source code to the issue tracker and emails.
- **Script Writing for Archie Tool:** Wrote scripts to automate the process of using the Archie tool to generate tactics files, aiding in the identification of architectural tactics.
- **Script for Commit Message Analysis:** Developed a script to automate the process of extracting issue IDs from commit messages to streamline issue tracking and association.

5 Challenges Encountered

During Week 1, several challenges were faced during the analysis:

- **Issue ID Not Specified in Emails:** Many emails discussed architectural decisions but did not reference specific issue IDs, making it difficult to correlate discussions to issues in the tracker.
- **Multiple Email Threads Opened on the Same Issue:** Some issues were discussed across multiple email threads, complicating the process of following the flow of the discussion and understanding the full context of the decision-making.
- **Mapping Issues to Emails:** Despite efforts, a significant challenge was the inability to consistently map issues from the issue tracker to their respective discussions in the mailing list emails.

Solutions:

- We matched keywords from the emails with issue descriptions, but this approach was not always accurate.
- We grouped emails based on recurring topics and keywords to form a more cohesive understanding of the discussions.

6 Results

The analysis identified several architectural patterns used to achieve quality attributes in the AXIS2 project. These patterns were centered around availability, security, and performance. Below is a summary of the key findings:

Quality Attribute	Tactic	Description
Availability	Fault Detection (Ping, Monitor)	Detect faults early using ping and monitoring mechanisms.
	Redundancy (Passive and Active)	Ensure system availability through replication and redundancy.
Security	Message Integrity Verification	Ensure message integrity using checksums and hashing techniques.
	Authentication	Control access by verifying identities and rights.
Performance	Event Prioritization	Prioritize events to handle high-priority tasks first.
	Concurrency Management	Handle multiple tasks simultaneously to optimize performance.

Table 1: Summary of Architectural Tactics Identified

7 Hours Spent

Below is a breakdown of the hours spent by each team member during Week 1:

Name	Hours	Remark
Adarsh Choudhary	7	worked on identifying availability issues
Mohit Jain	7	worked on identifying performance issues
Surabhi Lone	7	worked on identifying security issues

Table 2: Hours Spent by Team Members Week 1

Name	Hours	Remark
Adarsh Choudhary	7	Wrote scripts to automate generating tactics files and extracting issue IDs from commit messages.
Mohit Jain		
Surabhi Lone	7	Manually mapped issues from source code to issue tracker and emails.

Table 3: Hours Spent by Team Members Week 2

8 Next Steps

The next steps in the analysis will focus on further exploring maintainability and interoperability tactics, as well as improving the linkage between emails and issues:

- **Enhanced Tracking of Issue-Email Linkages:** We will make efforts to better connect emails to issues, especially when issue IDs are not provided.

- **Security and Performance Focus:** In Week 3, more detailed analysis on performance and security tactics will be conducted.

9 Source Code

The source code for this assignment is available in the following GitHub repository:
https://github.com/iamAdarshh/SADR-Assignment-2