Week 6: Examine Architectural Tactics

Nate Bachmeier

TIM-7140:Software Engineering

April 4th, 2021

Northcentral University

# Architectural Tactics

An architectural tactic is a design pattern that seeks that efficiently address a business problem. When software does not correctly implement these tactics, it produces significant risk toward the application’s security, reliability, and maintainability. Mirakhorli (2014) states once the architect becomes a “pile of mud,” any future changes likely come with a high regression risk. For instance, a program might hardcode the credentials or follow a monolithic design. These anti-patterns prevent the software from evolving into future needs. It is therefore critical that engineers understand design decisions and build to specification.

# Literature Review

Kontopoulos et al. (2018) describe the challenges of validating Automatic Identification System (AIS) metadata. Shipping vessels use this insecure protocol to transmit their location and intended route. The initial design requires that only ships exceeding 299 tons send these messages. However, countless smaller boats voluntarily leverage the system for safety purposes. Due to the data’s high volume, performing analysis is overly complex. Fraudulent and malicious transport companies spoof their location to exploit these issues. Along with hiding their location, it is also possible to fabricate congestion, causing legitimate cargo vessels to divert.

Addressing this big data problem is essential for preventing illegal activity, ensuring efficient routing, and protecting sailor’s safety. Exploiting the AIS data enables scenarios, such as illegal fishing and whale hunting. Farming animals into extinction will create irreversible changes to the fragile wildlife. Malicious actors that redirect third-party ships can increase costs and delays, creating unfair competitive advantages. When participants lose confidence in a safety system, that will result in accidents. These risks justify research investments into detecting deceptive activity and preventing it.

Kontopoulos et al. (2018) propose a scalable architecture for processing AIS firehose. Their solution begins with AIS messages flowing into the master node. After decoding the payload, a load balancer routes the traffic to an appropriate worker node. Message routing attempts to reuse the same worker node for repeated vessel updates. This decision improves caching performance, not calculation accuracy. After determining the ship’s delta position, a central consensus node predicts the probability of being spoofed. The prediction uses historical information to learn across many features like speed and routing consistency. The authors base their predictions on a combination of machine learning and statistical tests. These assessments include Support Vector Machines and feature deviations from the mean.

The authors determine the system’s accuracy by using 2.4GB of historical data from May 2016. After training the model, they use varying levels of data fuzzing to simulate malicious locations. These results show their solution has a precision above 85% and recall above 90%. There are specific classes of false positives that require further analysis. For example, flooding the worker nodes with too many wrong messages produces concept drift. Addressing those enhancements was outside the project’s scope. However, querying an external source of truth such as the manufacture design specifications could mitigate these issues. It would also be possible to learn these norms by examining more extended periods.