Journal of Engineering Technology and Sciences
Volume 02, Issue 05, May 2025 *ISSN: 3048-913X*
www.jets.org.in *DOI:10.63590/jets.2025.v02.i05.pp19-21*

Finding Anti-patterns in the Development of Service-Oriented Systems

Mohammad Rafi Khan
Department of Communications Engineering, King Abdullah University of Science & Technology, Saudi
Corresponding Author: mohammadraficse@gmail.com

To Cite this Article
Mohammad Rafi Khan, "Finding Anti-patterns in the Development of Service-Oriented Systems", Journal of Engineering Technology and Sciences, Vol. 02, Issue 05, May 2025, pp: 19-21, DOI: http://doi.org/10.63590/jets.2025.v02.i05.pp19-21
Submitted: 12-02-2025 *Accepted: 14-04-2025* *Published: 05-05-2025*

***Abstract**: One of the most widely utilized software architectural patterns for creating many contemporary systems is service-oriented architecture. It has, nevertheless, been a part of numerous failures. Anti-patterns are solutions that appear to be good but are actually flawed and lead to system failure. Numerous anti-patterns for SOA exist, and new ones are discovered daily. Anti-patterns are developed for different reasons and arise in specific areas of the problem. It will be challenging for architects to identify anti-patterns because the human mind is limited in its ability to process several states (pieces of information). In order to find SOA anti-patterns, we present a systematic approach in this paper that is based on a library of anti-patterns and a check list. This technique will help architects identify and steer clear of anti-patterns in the development process, hence avoiding anti-pattern-related hazards. Additionally, we offer a collection of 45 general anti-patterns in SOA in this study. By reviewing these anti-patterns, developers can prevent numerous potential issues and work with a clear grasp of patterns throughout software development phases. Additionally, our approach is tested in practice.*
.

***Keywords**: Checklist, Repository, Service-Oriented Architecture, Anti-patterns*

I. Introduction

In the process of service-oriented development, there are appealing and well-thought-out solutions that, in reality, are incorrect approaches and cause project failure. We refer to these remedies as anti-patterns. To prevent incorrect solutions (antipatterns) and effectively complete their projects, architects should be well-versed in all anti-patterns and keep up-to-date on fresh information by studying new anti-pattern resources. In the worst situation, an anti-pattern can lead to project failure and expense increases. It will be incredibly challenging for architects to find anti-patterns because there are so many of them and new ones are being discovered every day. Systems with service-oriented architectures are rarely threatened by these anti-patterns[7]. Regretfully, a lot of developers ignore these anti-patterns and instead focus on their prior knowledge, experiences, and accessible tools from previous projects. Managers, on the other hand, are project stakeholders who lack technical know-how in creating service-oriented systems. Consequently, a methodical approach to identifying anti-patterns is required. This paper presents a methodical approach to identifying anti-patterns in service-oriented development.

Our approach uses anti-pattern databases and check lists to find anti-patterns. This is how the remainder of the paper is organized. Section 2 will cover related topics, and Section 3 will present a repository of anti-patterns in service-oriented architecture. Additionally, in this section, we outline a few prominent anti-patterns that lead to additional anti-patterns. Section 4 will describe the methodical process for identifying anti-patterns in service-oriented architecture. Section 5 will include a case study that demonstrates the operation of our approach, followed by a conclusion.

II. In Service-Oriented Architecture and Anti-Patterns

An anti-pattern is a well-known and appealing way to solve an issue that isn't useful or practical. Stated differently, anti-patterns are solutions that fail rather than succeed [1]. The following components are part of a language or structure for expressing anti-patterns [2, 3]. It is evident from the aforementioned classification that anti-patterns are present throughout the whole software development process. Anti-patterns typically result in expensive and complex architecture, which makes implementation expensive and challenging. Even the application and use of anti-patterns in a system might cause a project to fail [7]. As a result, identifying and controlling anti-patterns is a top priority during the development process. Furthermore, this strategy is one of the most important ways for architects to ensure that their designs are appropriate and valid. On the other hand, there is a good chance that a private or customized solution used by architects will turn out to be incorrect and ultimately anti-pattern. Additionally, there is a greater chance that architects who employ new technology or a different strategy and lack service-oriented experience will fail.

It is evident from the aforementioned classification that anti-patterns are present throughout the whole software development process. Anti-patterns typically result in expensive and complex architecture, which makes implementation expensive and challenging. Even the application and use of anti-patterns in a system might cause a project to fail. As a result, identifying and controlling anti-patterns is a top priority during the development process. Furthermore, this strategy is one of the most important ways for architects to ensure that their designs are appropriate and valid. On the other hand, there is a good chance that a private or customized solution used by architects will turn out to be incorrect and ultimately anti-pattern. Additionally, there is a greater chance that architects who employ new technology or a different strategy and lack service-oriented experience will fail.

III. Assessing Anti-Pattern Risks

Since antipatterns can lead to project failure and increased costs, assessing antipattern risk and employing risk management strategies are important in SOA-based initiatives. The anti-patterns of SOA have different chances of occurring and can lead to different deficits [7]. These two parameters were used in [5] to quantify the risk of numerous anti-patterns. Depending on the developer's experiment, these two elements may be worth more or less for each anti-pattern. We shall have three types of anti-patterns with less, high, and very great risk probabilities, respectively, based on these two parameters. Anti-pattern risk management is accomplished by first providing a list of anti-patterns and then calculating their risk. They are arranged according to their risk values and contacted accordingly. Critical anti-patterns are anti-patterns that carry a high-risk value and require risk management. For anti-patterns with high-risk value, risk management is frequently required as well.

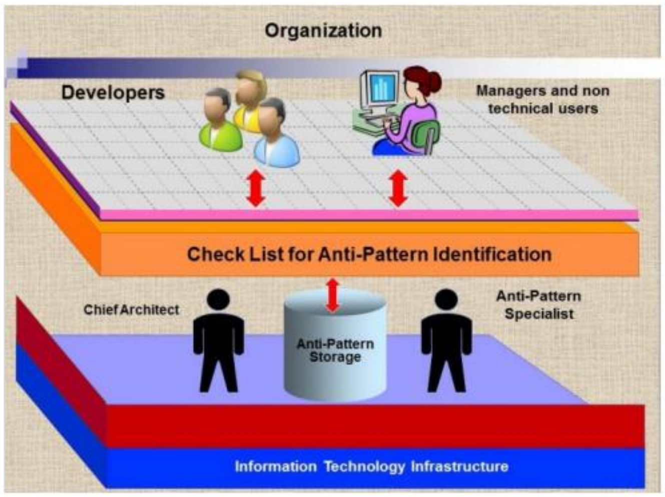
Organization
Developers
Managers and non technical users
Check List for Anti-Pattern Identification
Chief Architect
Anti-Pattern Storage
Anti-Pattern Specialist
Information Technology Infrastructure

Fig 1: Anti Pattern Identification

IV. The Suggested Model In Use

The Iranian Telecommunication Manufacturing Company (ITMC) is a business that works in the fields of ICT and electrical engineering. ITMC offers a wide range of web-based software, such as Finance and Human Resources, for both itself and other central telecoms companies, in addition to a few items in the electrical and communication sectors. Currently, staff members in the Training and Research unit are working with the MIS unit to create R&D websites using Visual C# and SQL Server. The two primary areas of the website are Training and Research. Users can access seminars and training courses in the training section. In addition, users can apply for openings and sign up for new courses. Additionally, those who wish to work as interns at ITMC should register online without physically being there. Additionally, ITMC course participants should receive their diplomas via websites. It should be possible for managers to access the website and obtain various reports via the local network. The services created by the website should be utilized by shareholders who live in different cities. Comprehensive details on the work done at ITMC, such as articles, research programs, projects, seminars, and presentations, should be included in the research section. A place for registering new ideas and research plans must also be included on the website. A person should register their idea if they would like ITMC to support it.

V. Conclusion

In the process of service-oriented development, there are appealing and well-thought-out solutions that, in reality, are incorrect approaches and lead to project failure. We refer to these remedies as antipatterns. Finding anti-patterns will be extremely challenging for architects because there are many anti-patterns and new ones appear every day. Systems with service-oriented architectures are rarely threatened by these anti-patterns. Project managers are stakeholders who lack technical know-how in creating service-oriented systems [7]. Consequently, a methodical approach to identifying anti-patterns is required. In order to help developers identify anti-patterns in service-oriented architecture and manage risk, we presented a novel approach in our article that is based on check lists and anti-pattern repositories. Organizations can identify anti-patterns with the help of a check list, which serves as guidelines for anti-pattern storage. Because human minds are limited in their ability to digest information, chick lists are necessary even when an organization has an up-to-date anti-pattern library and sufficient experience with service-oriented architecture.

References

[1] Noori M and Riazi M. “Anti-Pattern in Service oriented Architecture”. First Information Technology Conference, Iran, 2010, http://www.ibm.com/developerworks/webservices/library/ws-antipatterns/, 2010-11-14

[2] Carrato T and Srinivasan H. “Learning from Mistakes: A guide to SOA anti-patterns - how to benefit from known unworkable solutions”. 2007,http://soa.sys-con.com/node/318437, 2010-11-14.

[3] Brown WJ, Malveau RC, McCormick HW and Mowbray TJ. “AntiPatterns - Refactoring Software, Architectures and Projects in Crisis”. John Wiley & Sons, 1998.

[4] Kr´al J and Zemliˇcka M. “The Most Important Service-Oriented Antipatterns”. IEEE. 2007.

[5] Kr´al J and Zemliˇcka M. “Enterprise Architecture Antipatt”. IEEE. 2009.

[6] Kavis M. “Top 10 reasons why people are making SOA fail”. July 2008. http://www.cio.com/article/438413, 2010- 11-14.

[7] K, Nagendra Reddy. “Thermographic Assessment of the Quality of Electro-Mechanical in Railroad Automation.” Journal of Science Engineering Technology and Management Sciences, vol. 2, no. 5, Apr. 2025, pp. 7–11. Crossref, https://doi.org/10.63590/jsetms.2025.v02.i05.pp07-11.

[8] Jones S. “SOA anti-patterns”. http://www.infoq.com/articles/SOA-anti-patterns, Jun 19, 2006.

[9] Biswas P. “Principles of Service Orientation: What is SOA as illustrated by what it is not”. 2010, www.percepsys.com/blog/WhatIsNotSOA.pdf