



# Design and Implementation of Data Adapter in SWIM

Yangfei Sun<sup>(✉)</sup> and Yuanchun Jiang

Civil Aviation University of China, Tianjin 300300, China  
2019022090@cauc.edu.cn

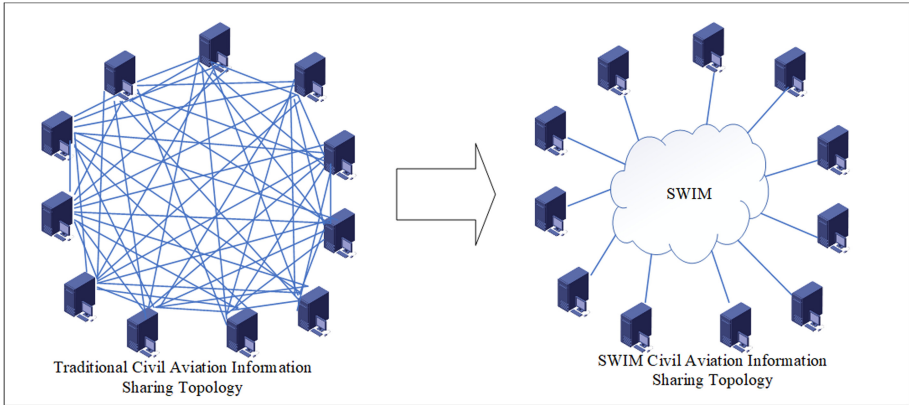
**Abstract.** In the background of the globalized interaction needs of civil aviation information services, the definition of System Wide Information Management (SWIM) was proposed by International Civil Aviation Organization (ICAO). However, the issue of interaction between traditional business systems and heterogeneous systems still exists on the SWIM platform. The data adapter is an important functional component to solve this problem, which can not only solve the problems of high coupling among systems, poor compatibility, low sensitivity, non-standard syntax and poor semantics, but also realize the high sharing of information between systems. Based on the SWIM platform, this article redefines SWIM. First, this article analyzes the concept and architecture of SWIM as well as the role and logical architecture of adapters. Then, according to the requirements of the adapter, a design method of the adapter structure is proposed, and the process design methods of the two main functional modules, data transformation and service encapsulation, are proposed respectively, including data standard and service standard. Extensible Markup Language (XML) technology is a platform independent language, 90% of systems support XML, it is self-descriptive and extensible, suitable for storage and transport on the network, therefore it is the preferred language for software development and is important for the design of SWIM data adapter. Finally, the data adapter is designed and implemented.

**Keywords:** SWIM · Adapter · Data conversion · Service encapsulation

## 1 Introduction

With the continuous development of informatization in the civil aviation industry, aircraft operators, airports, air traffic control (ATC) units, etc. have designed, developed and maintained specific information business systems, as a result, more and more appear inconsistent data interface among systems, which increase the coupling in traditional point-to-point communication systems, the paralysis of one system may affect the normal operation of many systems. The differences in data structures and definitions between different business systems will make it necessary to develop different interface standards for different system interfaces due to the increasing in system access, which improves the cost and obstacles of system interaction, it will adversely affect the normal operation of aircraft especially during cross-country flight. As an information exchange platform for global air traffic management, SWIM changes the traditional data sharing architecture

from a traditional point-to-point model to a SWIM sharing architecture centered on data transmission, exchange and management, as shown in Fig. 1. For example, for  $N$  different business systems,  $N(N-1)$  specific interfaces are needed to connect the systems. The number of development interfaces will increase geometrically with the increase of business systems, which not only increases development costs, but also leads to unnecessary waste of resources. For the model with SWIM as the shared architecture, you can access  $N$  different business systems only by developing common interfaces for new services or applications, which greatly improves the shielding ability of internal details and transparent services between different software and hardware systems.



**Fig. 1.** SWIM can effectively reduce the interface among systems.

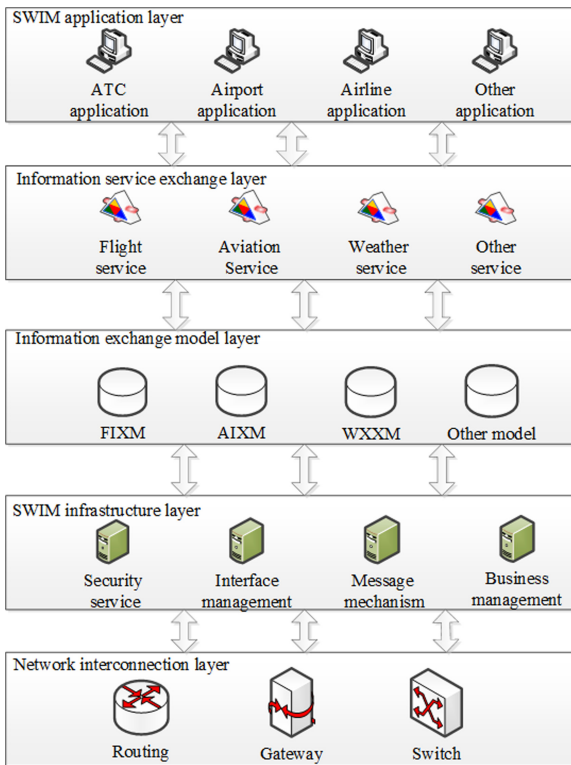
SWIM is a new, system-wide aeronautical information management method that integrates the air traffic management (ATM) network from the information level [1]. It is a platform based on this method that can provide airlines, airports, and ATC units access and interaction capabilities of different systems. According to ICAO's conceptual introduction to SWIM [2], the SWIM system adopts a service oriented architecture (SOA). Therefore, SWIM can improve the traditional point-to-point aeronautical information transmission mode into a centralized platform-based interaction method based on service management. It is based on this architecture that it can reduce the cost of system interaction, minimize the degree of tight coupling among different systems, and conveniently provide comprehensive management capabilities for aeronautical shared information, making information interaction more sensitive and economical. The SWIM interaction architecture is mainly divided into five layers, as shown in Fig. 2, which are SWIM application layer, information service exchange layer, information exchange model layer, SWIM infrastructure layer, and network interconnection layer.

As shown in Fig. 2, the SWIM coverage includes the middle three layers. The information service exchange layer defines appropriate services that the stakeholders reach a consensus for the corresponding information domain. The SWIM application layer will directly use the services provided by this layer for information exchange. The information exchange model layer provides the upper layer with a standard definition of the data model covering ATM shared information, including data content, structure and format.

The SWIM infrastructure layer provides infrastructure for sharing information, including interface management, message routing, security services, and enterprise service management.

An illustration is that SWIM is a “system of systems” [3], that is to say, SWIM is actually a platform for accessing the system, and actual services and data are distributed and provided by each access system, so ensuring the access capability of existing systems on the platform has become an issue that the SWIM must consider. To enable such systems to directly access the infrastructure layer of the SWIM core, the adapter architecture should include the second and third layer functions in the SWIM architecture. The adapter is an important component that guarantees the interaction of legacy services and data on the SWIM platform, which protects the existing cost of the civil aviation system and contributes to the wider promotion of the SWIM platform. Therefore, it is of great value to study the adapter of SWIM.

In this paper, specific research, design and implementation of adapter components are carried out for the concept and architecture of SWIM. Firstly, it introduces the characteristics and composition of the SWIM architecture, then a design method of adapter is proposed.

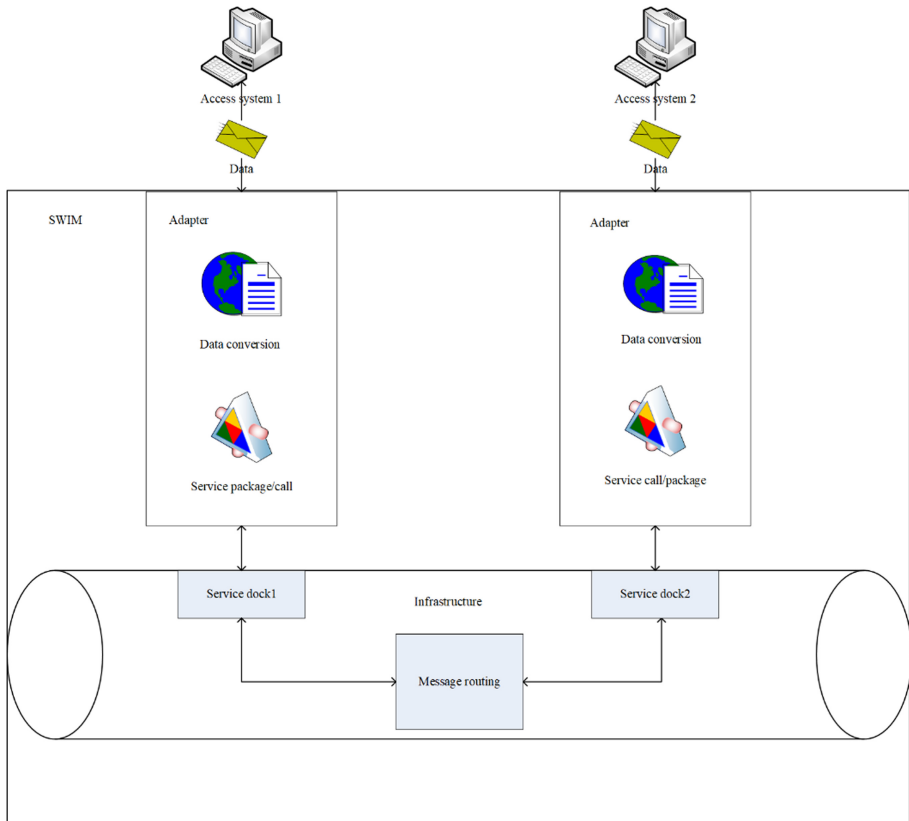


**Fig. 2.** The interactive architecture of SWIM.

## 2 The Analysis of Adapter Functional Requirement

Based on the analysis of SWIM's interactive architecture and the role played by the adapter, taking two access systems as examples, the logic architecture of the adapter in SWIM is designed, as shown in Fig. 3.

The logical architecture of the adapter describes its logical position and logical function on the SWIM platform. The two access systems in Fig. 3 are both legacy systems that need to access the SWIM platform infrastructure to complete the interaction process. Therefore, the data first undergoes data conversion and service encapsulation through the adapter before it can be successfully connected to the SWIM infrastructure. In addition, the infrastructure in the logical architecture needs to provide basic interface access functions, namely service dock and message routing functions. The service dock is a public interface provided by the infrastructure to access the service, while the message routing provides the ability to connect to each other in the case of multiparty interaction. The SWIM-based adapter needs to have the following two functions.



**Fig. 3.** The organization structure of adapter.

## 2.1 Data Conversion

Due to the differences in the format of the data sources of each access system, there are certain differences in data structure and semantics, which directly affect the accuracy and effectiveness of information interaction, resulting in obstacles in data exchange and management on the SWIM platform. Therefore, when each system is connected to SWIM respectively, the adapter should complete the data format conversion function, and the infrastructure should use a unified data format for information management. According to the recommendations of the ICAO and European and American countries in studying SWIM [4], XML, as a language that has nothing to do with the system platform environment, and is standardized, versatile, and extensible, is suitable to be a general-purpose language data format. Therefore, the adapter should support the data conversion function that converts the data format to the XML format.

In order to form a standardized XML data format, it is necessary to establish an appropriate data model to restrict the format and structure of the data source. The SWIM data model is a model formed by using certain modeling standards and technologies to classify different aeronautical business data and express them uniformly. Various data sources can realize the conversion and unification of data formats that meet the standard on the basis of the data model. Therefore, the adapter should use a well-defined data model in order to meet the requirements of the data conversion function.

## 2.2 Service Encapsulation

SWIM uses the SOA, which means that a service is the basic information unit for interaction among different systems. The SWIM infrastructure layer, in essence, is the core SWIM architecture layer that provides service interface management capabilities and completes the actual service interaction functions. In other words, the SWIM core platform does not actually run business systems, but interacts by accessing normalized distributed businesses as services. Such an architecture guarantees the neutrality, loose coupling and scalability of the platform technology. Therefore, the adapter needs to encapsulate the data as a service to publish or expose to achieve the callability of the service.

A service is a business concept determined by an application or a user. It is a technology-independent module that is deployed on a standard middleware platform and can be invoked on the network. The adapter needs to define these service concepts and implement specific service instances so that the SWIM platform can manage the services.

## 3 The Design of Adapter Structure

Since a large part of the civil aviation business remains on the reporting system, the data source uses the message as the center to design the adapter. According to the above-mentioned adapter functional requirement analysis, with corresponding technical support, the design of the specific structure of the adapter is shown in Fig. 4.

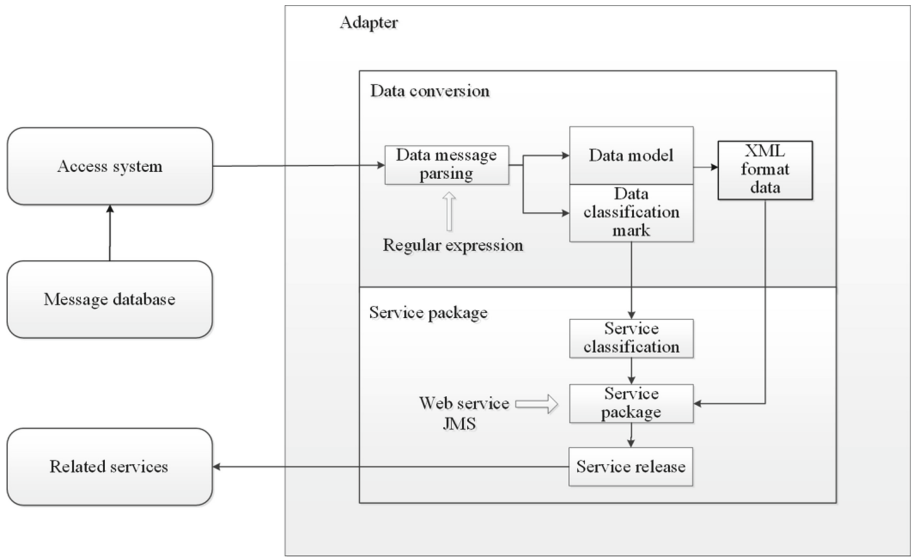


Fig. 4. The design of adapter structure.

### 3.1 The Design of Data Conversion Module

According to the requirement analysis of the adapter, the data passing through the data conversion module should be in a standard XML format. Therefore, the data conversion module needs to parse the message data source and use the standard data model to unify the data format. The data conversion module is composed of several sub-modules, namely the message parsing sub-module, the data model sub-module and the corresponding data classification and marking sub-module.

The main function of the data conversion module is to process the input message data source through each sub-module, and finally obtain the XML data that conforms to the data model. First, the data source message will be parsed by the message parsing submodule, which uses regular expressions to match the relevant content of the message. Regular expression is a method of extracting information using a specific type of character to match data. The analysis content includes judging the message type and mapping the aeronautical data related to the message. Then, the parsed message type will be handed over to the message type marking module for marking, and help the service encapsulation module to select different services. At the same time, the relevant aeronautical information data will be transferred into the data model corresponding to the mark for the conversion and unification of the data format, and finally the data in the standardized XML format will be obtained. The specific design method is shown in Fig. 5.

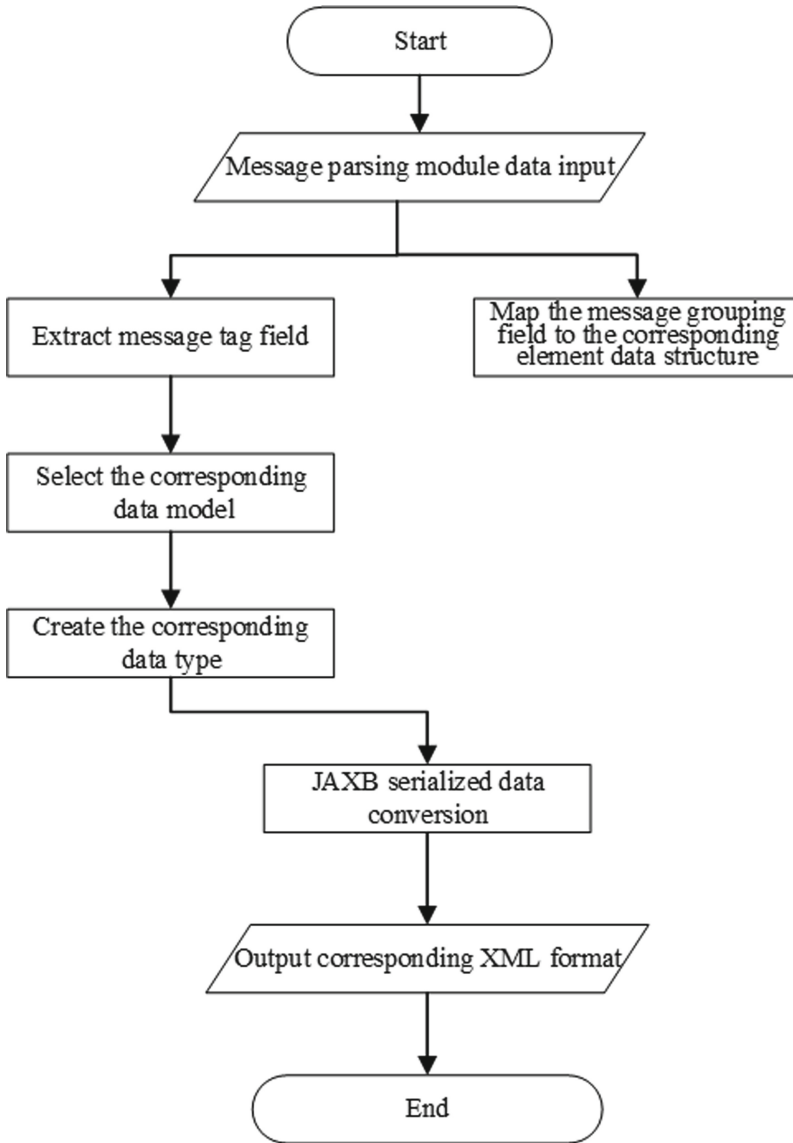
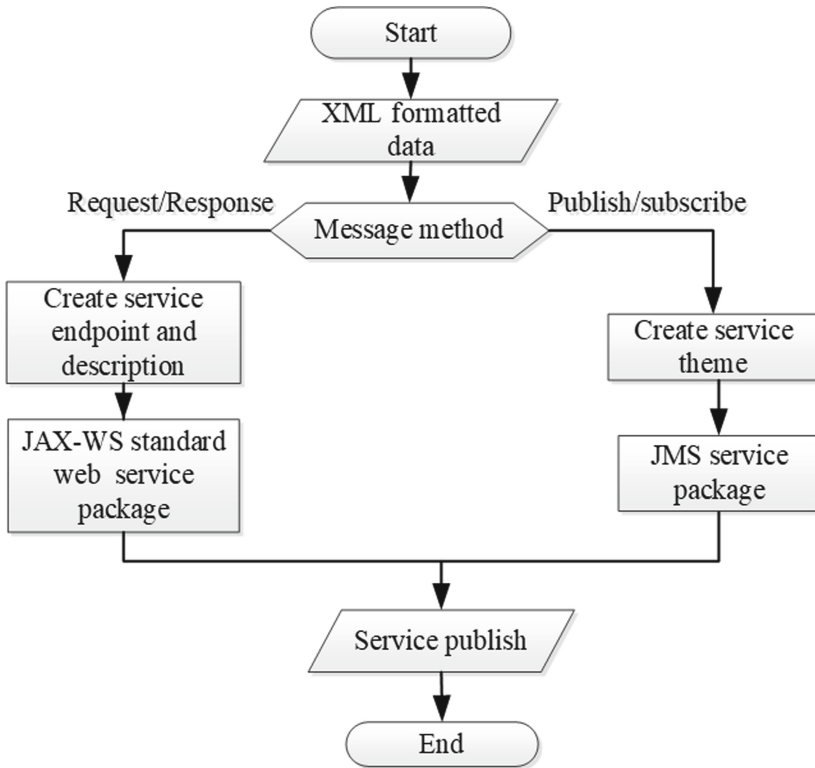


Fig. 5. The flowchart of data conversion.

### 3.2 The Design of Service Encapsulation Module

The service package module further encapsulates the data in the standard format into general services, in order to shield the heterogeneous interaction problems, and can more efficiently manage the interactive business on the SWIM platform. This module mainly provides several functional sub-modules of service classification, service encapsulation and service publishing. The design among modules is as follows.



**Fig. 6.** Data conversion flowchart.

Firstly, the service classification module selects the corresponding service to be encapsulated according to the service mark in the upper data conversion module, and then loads the converted format data into the corresponding business class to encapsulate the service, which can realize business process by using Web Service or Java Message Service (JMS) technology. Web Service is a very commonly used technology in SOA, which is a network-based distributed modular component that can publish callable functions to the Web for application access (applications can use standard Web protocols and data formats to access it). Because Web Service follows certain technical specifications, it can have good compatibility with other components or systems [5]. Web Service provides a standard method for converting the functions of legacy applications into reusable, self-contained, and self-describing services, and a standard way for convenient and flexible application integration. Therefore, it is very suitable as a standard technology for service encapsulation modules. A development method that requires high real-time information services. JMS is an application program interface used to access asynchronous messaging systems, and publish-subscribe model can be used to implement publish-subscribe services. ICAO recommends using JMS as a viable implementation method before the Web Service standards for service subscription are



not yet mature. Finally, the service publishing module publishes the encapsulated service to the corresponding application server for invocation by service users. The specific design method is shown in Fig. 6.

The adapter needs to encapsulate different services into corresponding types of services. At the same time, the service development process will vary according to business needs. For example, the aeronautical business of aeronautical information service requires aeronautical information consumers such as airlines to subscribe for a fee. When the service is updated, the aeronautical information will be automatically pushed to subscribed consumers, and some services require service users to actively apply and call services synchronously. Therefore, regarding the service development process of the adapter, two development methods are adopted for different businesses, one is a publish-subscribe service, and the other is a request-response service.

JMS provides a set of interfaces for service publishers and service subscribers. Both parties need to jointly maintain a conversation and the topic created by it. The service publisher encapsulates information on a specific conversation topic on the JMS server. After the two types of service development are completed, the service can be published for the service caller to call the service remotely.

#### 4 The Results of Service Test

Service testing uses JMS technology, which is the encapsulation of the service used in the subscription publishing model, using airport and weather data as test cases to test the service during the service interaction. Start the tomcat server and conduct the subscription operation of the subscriber.

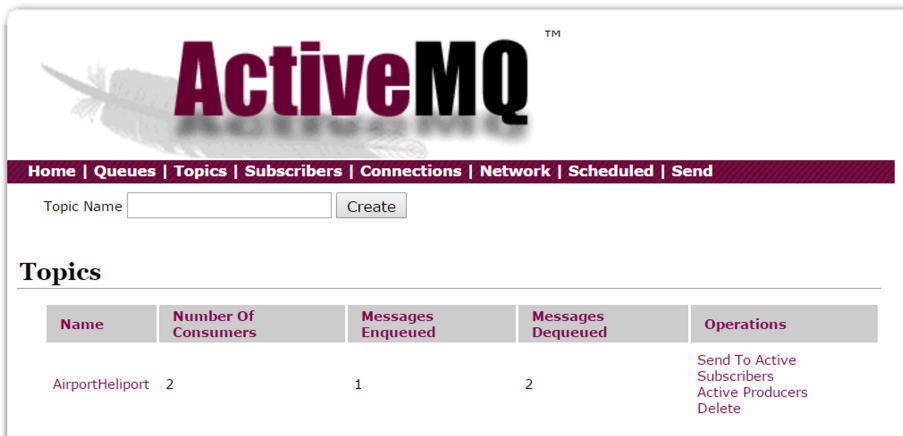
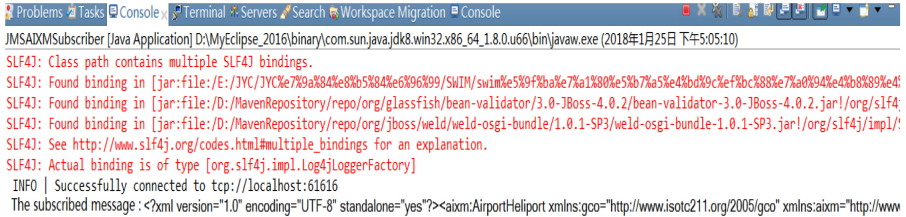


Fig. 7. Test results that airport heliport information service JMS releases.

The test results are displayed through the console and web port. In the test case, after two subscribers subscribe to the service whose subject is AiportHeliport, the service provider publishes the service, the two subscribers can obtain service information and

log in to the server using a browser, namely can display test results and the displayed test results are shown in Fig. 7.

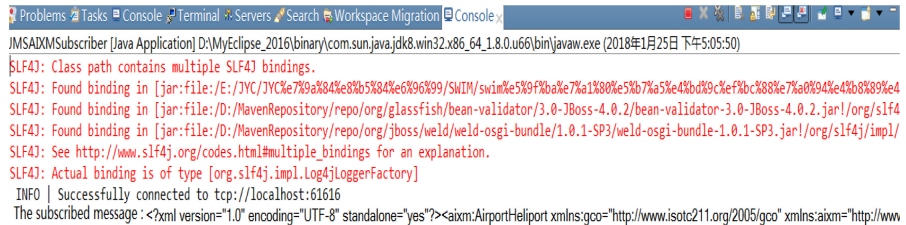
In Fig. 7, the JMS server shows the interactive test results of the subscriber and the publisher under the message topic about the airport service. There are two subscribers under the AirportHeliport topic. When the service message enters the topic, the two subscribers will get the message respectively. The console output of the two subscriber clients is shown in Fig. 8.



```

JMSADMSsubscriber [Java Application] D:\MyEclipse_2016\binary\com.sun.java.jdk8.win32.x86_64_1.8.0.u66\bin\javaw.exe (2018年1月25日 下午5:05:10)
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/E:/JYC/7YCYe7%9a%84%e8%b5%84%e6%96%89/SWIM/swim%5%9f%ba%7%a1%80%e5%b7%a5%e4%bd%9c%ef%bc%88%e7%a0%94%e4%b8%89%e4%
SLF4J: Found binding in [jar:file:/D:/MavenRepository/repo/org/glassfish/bean-validator/3.0-JBoss-4.0.2/bean-validator-3.0-JBoss-4.0.2.jar!/org/slf4j/impl/
SLF4J: Found binding in [jar:file:/D:/MavenRepository/repo/org/jboss/weld/weld-osgi-bundle/1.0.1-SP3/weld-osgi-bundle-1.0.1-SP3.jar!/org/slf4j/impl/
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
INFO | Successfully connected to tcp://localhost:61616
The subscribed message : <?xml version="1.0" encoding="UTF-8" standalone="yes"?><axim:AirportHeliport xmlns:gco="http://www.isotc211.org/2005/gco" xmlns:axim="http://www
  
```

(a) Subscriber 1 output result.



```

JMSADMSsubscriber [Java Application] D:\MyEclipse_2016\binary\com.sun.java.jdk8.win32.x86_64_1.8.0.u66\bin\javaw.exe (2018年1月25日 下午5:05:50)
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/E:/JYC/7YCYe7%9a%84%e8%b5%84%e6%96%89/SWIM/swim%5%9f%ba%7%a1%80%e5%b7%a5%e4%bd%9c%ef%bc%88%e7%a0%94%e4%b8%89%e4%
SLF4J: Found binding in [jar:file:/D:/MavenRepository/repo/org/glassfish/bean-validator/3.0-JBoss-4.0.2/bean-validator-3.0-JBoss-4.0.2.jar!/org/slf4j/impl/
SLF4J: Found binding in [jar:file:/D:/MavenRepository/repo/org/jboss/weld/weld-osgi-bundle/1.0.1-SP3/weld-osgi-bundle-1.0.1-SP3.jar!/org/slf4j/impl/
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
INFO | Successfully connected to tcp://localhost:61616
The subscribed message : <?xml version="1.0" encoding="UTF-8" standalone="yes"?><axim:AirportHeliport xmlns:gco="http://www.isotc211.org/2005/gco" xmlns:axim="http://www
  
```

(b) Subscriber 2 output result.

Fig. 8. The output results of airport heliport information service test subscriber.

Figure 8(a) and Fig. 8(b) respectively show the XML message output of the AirportHeliport service received by two subscribers.

## 5 The Results of Performance Test

Performance testing is to test the content of the adapter's performance requirements and give conclusions including the adapter's data conversion time efficiency, as well as the adapter user interface loading time, resource consumption, and operating conditions in different environments.

The time efficiency performance of the adapter's data conversion refers to the time consumption of its data conversion module. The average time-consuming test for different test case inputs uses four test cases, and the data conversion module is used to perform the time-consuming test of data conversion. After 30 tests, count the test time and calculate the average data conversion time of each test case. The test results are shown in Table 1.

**Table 1.** The test results (ms) of data conversion time efficiency.

Testing frequency	1	2	3	4	5	6	7	8	9	...	30	Average time
Test case 1	365	386	362	370	369	386	345	365	336	...	362	369.2
Test case 2	326	325	305	315	316	320	333	295	327	...	330	316.4
Test case 3	302	286	288	310	293	297	303	305	293	...	295	294.3

The test results of other performance indicators of the adapter, computing resource consumption, including page loading time, and operating environment feasibility are shown in Table 2.

**Table 2.** The test results of other performance.

Computing resource consumption	Program space occupation	19.4 M
	Server running memory usage	92.3 M
Page loading time	Login page load time	243 ms
	Main page loading time	295 ms
Operating environment feasibility	Windows	Feasible
	Linux	Feasible

In the performance test results in Table 2, the adapter’s computing resource consumption, including the space occupied by the program and the memory occupied by the server, are all at the M level, and the page load time is at the ms level, and it can run on both Windows and Linux at the same time, conforming the adapter performance requirements.

Based on the introduction of the SWIM platform, this article designs and implements the adapter. As an important component to ensure the smooth interaction of legacy systems on SWIM, adapters need to implement data conversion and service encapsulation functions. At the same time, the design and implementation should meet the data model specifications and data format standards provided by ICAO. Realize with JAXB technology, and finally get data conforming to the data model and XML format. The service encapsulation module encapsulates and publishes services based on the service business. With the further deepening of civil aviation informatization, future adapters can expand the adaptation of more aeronautical systems on the basis of considering more services and processes.

## References

1. Zhigang, L., Yanying, G.: Research on the survivability of system wide information management (SWIM). In: 2019 IEEE 5th International Conference on Computer and Communications (ICCC), Chengdu, China, pp. 1388–1392 (2019)

2. ICAO, Manual on System Wide Information Management (SWIM) Concept, Doc. 10039 (2014). <https://www.icao.int/airnavigation/IMP/Documents/SWIM%20Concept%20V2%20Draft%20with%20DISCLAIMER.pdf>
3. Morioka, K., et al.: Field taxing experiments of aircraft access to SWIM over AeroMACS. In: 2018 IEEE Conference on Antenna Measurements & Applications (CAMA), Vasteras, pp. 1–4 (2018)
4. SESAR Joint Undertaking, SWIM-TI Yellow Profile Technical Specification 2.1, Edition 00.10.00 (2014). [http://www.sesarju.eu/sites/default/files/solutions/07\\_TS\\_Solution\\_20\\_14.01.04.D44-004-SWIM-TI\\_Yellow\\_Profile\\_Technical\\_Specification.pdf](http://www.sesarju.eu/sites/default/files/solutions/07_TS_Solution_20_14.01.04.D44-004-SWIM-TI_Yellow_Profile_Technical_Specification.pdf)
5. Papazoglou, M.P.: Principle and Technology of Web Services. Machinery Industry Press (2010)