Cross-Platform Civil Aviation Data Exchange Based on XML

Jia Liu, Xiling Luo, and Feng Zeng

***Abstract*—By analyzing the current Civil Aviation systems, we find the data exchange is difficultly implemented owe to the high complexity. In this paper, we propose a method of cross-platform data exchange based on XML. We highlight the data exchange functions of this method, including data format and transfer protocol conversion. Using the Web Services architecture, the message transmission can treat more easily of the different systems and programming languages. It is demonstrated that this is an efficient way to overcome the shortcoming of the current situation of the Civil Aviation which is primitive, single and with the fixed data exchange mode.**

***Index Terms*—Data exchange, SOAP, XML**

I. Introduction

With the development of the China Civil Aviation, there has been a rapid growth in the airlines passengers and freight volume. As the result of the increasing demand, China Civil Aviation is ushering in the high developing peak.

Since Civil Aviation is a technology intensive and high security industry, all the business systems play significant roles in the operation. At present, each application system gives the different information descriptions so that the data exchange becomes more difficult [1]. This situation is so called the information isolated island [5]. The problem will become more prominent in the Civil Aviation in next several years. Therefore, how to construct the functional module of the data exchange and transmission becomes the most important and urgent issue. The next generation of China Civil Aviation Air Traffic Management system has begun the research and construction work with the System Wide Information Management (SWIM) as the key platform and technology [2]. The FAA NextGen and the Europe Sesar programs all chose the SWIM to enhance the data sharing ability of the current Air Traffic Management information systems; eliminate the information isolated island; integrate the isolated information systems and realize the full value of resource data.

The SWIM is a series of plans and programs to provide the data sharing and exchange platform among different systems. By this way, the related information can be interacted among the airport, ATM and Airline Company safer, faster and more efficiently [1], [2]. For the development and operation part, the SWIM separates the Civil Aviation application systems from the IT infrastructure in order to keeping loose coupling and the flexibility. Functionally speaking, the SWIM integrates all kinds of basic data, including flight information, AIM and weather information. Then it publishes the data through the ESB wrapped as Web Services.

Service Oriented Architecture (SOA) is the popular distributed system architecture, which focuses on the business integration and fast change, as well as the architecture flexibility and the resource reusability. Since XML is a universal language standardization with strong extension and frame work [5]. After converting data to into XML file, the heterogeneous system carries out the exchange with Web Services mode [3]. Therefore, many research centers, such as IBM, Microsoft, Sybase, have conducted the research on data exchanges, mainly for the star exchange pattern.

This paper will analyze the Civil Aviation business data exchange module design and realization based on the XML. The whole paper is divided into five parts. First, we give the introduction of the SWIM program and the main technology used in the design part. Then we analyze the domestic and foreign related work in Section II. In Section III, we propose the design of Civil Aviation data exchange model. After that, we present the implementation work in the Section IV. Then, we demonstrate the realization results in the Section V. In the end, we draw a conclusion of the cross-platform data exchange design and realization in the Section VI.

II. Cross-Platform Data Exchange Technology

A. Adapter Pattern

The adapter pattern is one of the design patterns reflecting the idea of software reuse. And it puts forwards the common used and efficient solution during the object oriented development. The final goal of adapter is to make the current system compatible with the new function so that the interface should be changed into another wanted interface.

There are two methods for the software reuse-inheritance and combination. According to the different reuse methods, the adapter pattern is divided into class adapter and object adapter. The class adapter realizes the client class required interface through inheriting the adaptee class. When the client calls the method of the adapter, the adapter will internally call the method of the inherited adaptee class. For the object adapter, it reuses the adaptee class via combination including a reference. As the client calls the method of the object adapter, it will call the proper method of the included

Manuscript received July 26, 2012; revised September 4, 2012. This work was supported in part by the Research on Service-oriented and Efficient Air Traffic Management Information Sharing Mechanism project, CAAC (No.MHRD201047). The title is Cross-Platform Civil Aviation Data Exchange Based on XML.

Jia Liu and Xiling Luo are with the School of Electronics and Information Engineering, Beihang University, Beijing, China (e-mail:yolanda8748_lj@sina.com, luoxiling@buaa.edu.cn).

Feng Zeng is with the Aviation Data Communication Corporation, Beijing, China (e-mail: zengf@adcc.com.cn).

adaptee class instances. However, the class adapter adopts the multi-inheritance which brings the tight coupling which is not recommended. The object adapter uses the combination way in line with the loosely coupling.

As for the data transmission direction, there are unidirectional adapter and bidirectional adapter. The unidirectional adapter only realizes the target interface and quotes the interface of the adaptee. On the other hand, the bidirectional adapter can realize the interfaces of target and adaptee, which quotes the interface of the two interfaces.

Therefore, when we come to the situation that the two irrelevant classes need put together, we can choose to change the interface. If we could not have the source code, adapter is used to build a compound interface.

B. XML

In order to interchange between the different systems, we should define a data format primarily for analyzing by both systems. Meanwhile, the defined data format should be independent of the platform and programming language in order to the flexibility and complexity.

The W3C put forwards the XML (Extensible Markup Language) to define a data exchange standard [6]. Using this standard, XML can allow the client to define the new mark for identifying the document organization. Since XML provides a structured consistency method, it can be used in data description and transmission. XML has many advantages, such as extensibility, content independency and platform independency. Therefore, XML can overcome the HTML disadvantages and shield the heterogeneity of the source data structure and application environment [3], [8]. Although XML takes up more space than the binary data, it is extremely simple to master and use, which makes it more proper for web data exchange.

C. SOAP

SOAP (Simple Object Access Protocol) is a protocol specification for exchanging structured information in the implementation of Web Servicess in computer networks. It relies on XML for its message format to provide the simple information mechanism, and usually relies on other application layer protocols, most notably HTTP (Hypertext Transfer Protocol) and Simple Mail Transfer Protocol (SMTP), for message negotiation and transmission [7], [10].

SOAP itself doesn't define any application program semantic, neither the programming model nor the specific semantic realization [9]. In fact, it provides the package model with standard component as well as coding data in the module. Therefore, SOAP is commonly used in data transmission to all kinds of RPC (Remote Procedure Call) systems. SOAP is divided into 4 parts, they are:

- SOAP encapsulation: it defines a framework that describes the information content, the handler and whether it is optional or not.
- SOAP encoding rules: it provides a serialization mechanism for exchange the instance of the application data format.
- SOAP RPC representation: it well defines the agreements of the RPC and response.
- SOAP binding: it designs an agreement for the exchange SOAP encapsulation between the nodes using underlying transfer protocols.

III. Civil Aviation Data Exchange Model Design

Through the construction of the SOA, all the application systems can be connected logically by the service bus. The adapter is the interface between the application systems and the service bus in order to bidirectional information interaction and data conversion. When accessing, the new business system will publish the new business function to the ESB through the adapter. Therefore, the other SWIM systems can access to these business functions and data instead of the new connection with the business system. The following picture shows the logical architecture of the adapter, ESB and the business systems:

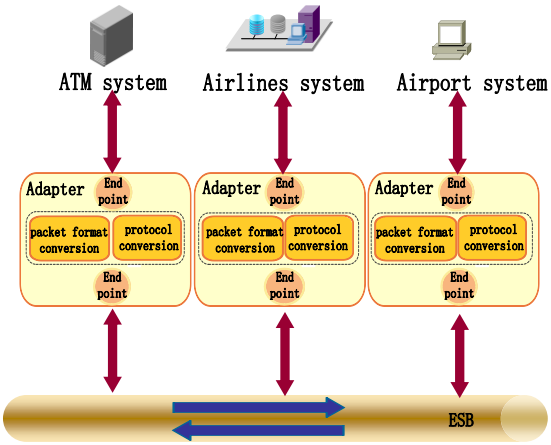

Fig. 1. The architecture of the SWIM system.

A. Data Exchange Endpoint

According to the Civil Aviation system and industry requirements, we design the data exchange end point as the external interface provided by adapter. Through data processing end point, conversion platform can implement the data and message transfer. The bidirectional processing endpoints can ensure the business systems not only publish but also receive services. In this way, the information transmission rate can be increased and the data flow can be enhanced. Since the adapter is designed to support multiple information exchange modes, so how to guarantee the security of the data transmission should be the key issue that we have to consider about. The SSL (security socket layer) protocol is the most widely used security protocol all around the internet. It uses the public key encryption for the identity authentication.

B. Data Adapter

The main functions of the data adapter are package format conversion and protocol conversion. First, we should analyze the Civil Aviation about the data formats and values.

After the modeling, we can define the XML file as the unified standard. The current business systems publish the data using different protocol including HTTP, JMS, MQ or FTP. So we have to convert the transfer protocol to the wanted certain one to ensure the receiver get the service from the provider correctly.

C. ESB

In the ESB module, the consumer and receiver don't

interact directly. Instead, there is a virtual and manageable bus to realize and expand the core of SOA [4]. The main functions of the ESB are the receiving concurrent routing information, service registration, service search, system monitoring and so on. The data processing concludes the processing of the router, format conversion, database reading and writing. The unified service registration and storage management refers to the service registration, publish and query. Using ESB to transmit the service can achieve the integration without limitation. It can support all kinds of the data models with the most extensive data format analytical ability. Due to the best performance, it can guarantee the increase of business.

IV. Implementation

Cross-Platform Civil Aviation Data Exchange is involved in two core processes: Data format transformation process and user protocol conversion process. The whole system design is as the Fig. 2 represents.

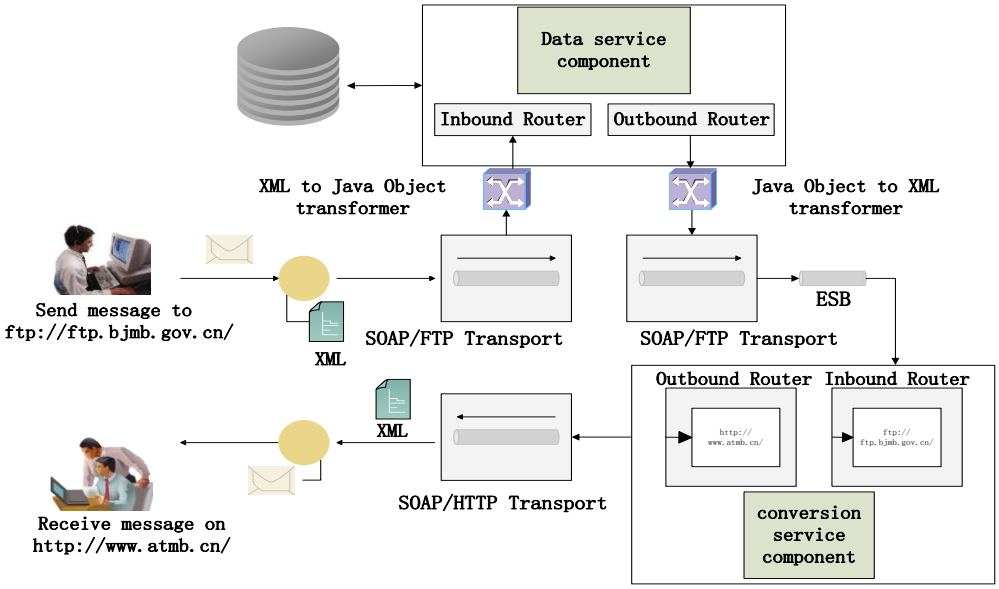

Fig. 2. The design of the data exchange system.

A. Data format Transformation Process

Data format transformation process involves the data exchanging node which submits local Civil Aviation resources and realizes data exchange service.

1) Step 1

Exchange node requests format transformation module at first. If the local data is invalid, the exchange node would send back the data with an invalid information notice. If it is valid, the exchange node will start the encryption to the local data.

2) Step 2

Encrypted data will be sent to the format transformation module. Since the local data are presented in multiple format so that the exchange platform needs to define a unified data format. Here, we design the XML file as the template. At the receiver point, the format transformation module should also convert the message back into the local data format. The mapped standard XML file is designed as Fig. 3.

B. User Protocol Conversion Process

The exchange endpoint identifies the transfer protocol of the service provider. Once the data exchange endpoint is received data successfully, the provider can use the protocol conversion service of the exchange platform.

1) Step 1

The protocol conversion module needs to recognize the protocol which the data provider chooses to use at first. For example, the interface of dynamic flight data in the ATM system always adopts the FTP protocol. The following is an analysis of FTP set up process flow in JAVA:

```
<?xml version="1.0" encoding="gb2312"?>
<x:DynaFlightDataList xmln:x="urn:DynaFlightData">
  <DynaFlightData>
    <BasicData>
      <routeInformation>
            <name>AR20120508</name>
            <date>2012-05-08</date>
            <dataRecord>
                ...
            </dataRecord>
      </routeInformation>
      ...
    </BasicData>
    <FixedPlanData>
      <name>FIXED20120508</name>
      <date>2012-05-08</date>
      <dataRecord>
          ...
      </dataRecord>
    </FixedPlanData>
    <DynaData>
      <DY>
          <name>DY1308</name>
          <date>08-13</date>
          <dataRecord>
              ...
          </dataRecord>
      </DY>
      <TO>
          <name>T020100509</name>
          <date>2012-05-09</date>
          <dataRecord>
              ...
          </dataRecord>
      </TO>
    </DynaData>
  </DynaFlightData>
</x:DynaFlightDataList>
```

Fig. 3. The example of the standard XML file.

```
public FTPClient getLoginedFtpClient() throws IOException,
FTPException
{if (this.ftpClient == null){ init();}
    if (this.ftpClient.connected() == false)
            { this.ftpClient.setRemoteHost(this.ftpHost);
        this.ftpClient.connect();
          ftpClient.uploadFile(RWFileDir,FileName);  }
        return ftpClient;}
```

2) Step 2

Then, we need to bind the SOAP protocol over the FTP protocol. The example FTP request with SOAP envelope as the Fig. 4 shows. After that, the enveloped data should be sent to the XML-to-Java Object transformer in order to call the data service connected with the database.

```
PASV/ Path/DFD.XML PROFTPD/1.3.1
IPAddress: 192.168.1.2
PORT:21
Content-Length:113
SOAPAction: " urn: DFD: transfer
< SOAP: Envelope…
```

Fig. 4. The example of the SOAP binding FTP.

3) Step 3

When the data service is finished, the verified data will be sent back and taken to the ESB to transmit continuously. As for the message receiver, the protocol conversion service will convert the transfer protocol into the HTTP. The SOAP binds the HTTP as the Fig. 5 shows.

```
POST / path/ VDFD. XML HTTP/ 1. 1
Content- Type: text / XML
SOAPAction: "urn: VDFD: transfer"
Content- Length: 124
< SOAP: Envelope…
```

```
HTT P/ 1. 1 200 OK
Content- Type: text / XML
Content- Length: 136
< SOAP: Envelope…
```

Fig. 5. The example of the SOAP binding HTTP.

Besides, through the design of the standard data format, the users can access to the service they want fast and efficiently. And the XML file can be expanded including more data like weather and traffic information in the future. Combining the different data to one XML file, the users would subscribe more information one time.

V. Realization Results

The research of this program is applied preliminarily in the SWIM system construction. The following user interface is designed to upload the related data files of the business systems in the FTP server and the port number is 80. And we design the file upload function as either single upload or batch upload. Then the user interface will access to the format transformation part which is illustrated by Fig. 6. Through the function of protocol convention module, we can get the information of the uploaded file using browser as the Fig. 7 shows.

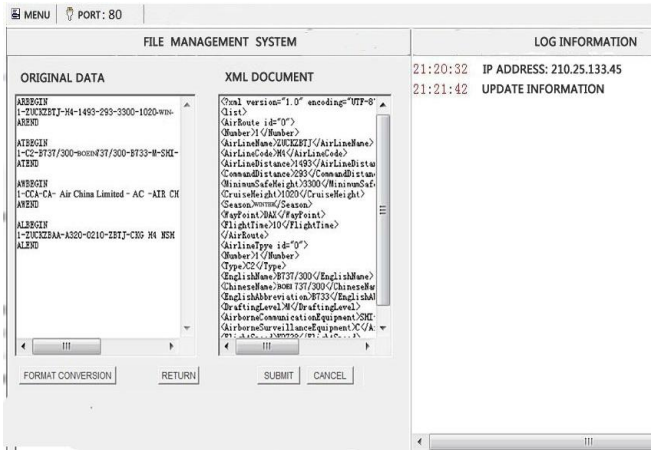

Fig. 6. The example format transformation on the FTP server.

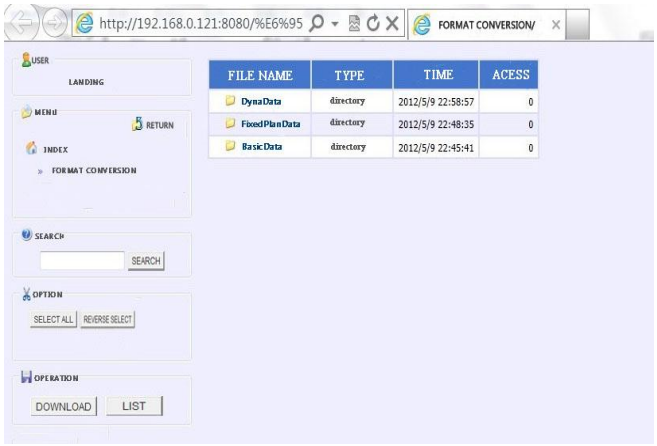

Fig. 7. The example of browser download user interface.

VI. Conclusion

In this paper, with brief introduction of the data exchange research, we approach the Civil Aviation data exchange problem with XML technology. As for the cross platform problem, we propose Web Services architecture. Focused on the protocol and format conversion, we design the detailed operation process and specific implementation. By giving the standard transmission data format, we could bring more flexible service composition function.

Acknowledgment

The authors would like to acknowledge the Research on Service-oriented and Efficient Air Traffic Management Information Sharing Mechanism project, CAAC (No.MHRD201047) for supporting this work.

References

[1] D. D. Crescenzo, A. Strano, and G. Trausmuth, "System Wide Information Management: The SWIM-SUIT prototype," in *Proc. IEEE ICNSURV*, May 2010, pp. C2-1 - C2-13.

[2] B. Stephens, "System-Wide Information Management (SWIM) Demonstration Security Architecture," in *Proc. IEEE DASC*, Oct. 2006, pp. 1 - 12.

[3] J. Xu, G. Xiao, J. W. Lu, Q. H. Liang, and J. Shen, "Customizable Data Exchange Based on Web Services," in *Proc. IEEE ICEBE*, Oct. 2009, pp. 522 - 525.
[4] Y. L. Wang, P. S. Sun, W. B. Zhang, and Z. F.Cao, "Research on Data Exchange Platform for Power Enterprise Based on ESB," in *Proc. IEEE CISE* , Dec. 2010, pp. 1 - 4.
[5] Y. Cheng and H. B. Shen, "Design and realization of data interchange platform based on XML," in *Proc. IEEE ICFCC*, May 2010, pp. V1-122 - V1-126.
[6] M. S. Yang and Y. Zhang, "Quality Information System Data Integration Technology Based on CORBA and XML," in *Proc. IEEE ICMSS*, Sept. 2009, pp. 1 - 4.
[7] M.-C. Rosu, "A-SOAP: Adaptive SOAP Message Processing and Compression ," in *Proc. IEEE ICWS*, July 2007, pp. 200 - 207.
[8] A. Vakali, B. Catania, and A. Maddalena, "XML data stores: emerging practices," *IEEE Trans. Internet Computings*, vol. 9, pp. 62-69, Apr. 2005.
[9] P. Louridas, "SOAP and web services," *IEEE Trans. Software*, vol. 23, pp. 62-67, Dec. 2006.
[10] T. M. Chester, "Cross-platform integration with XML and SOAP," *IEEE Trans. IT Professional*, vol. 3, pp. 26-34, Sep/Oct 2001.

Jia Liu is now a postgraduate student at the School of Electronics and Information Engineering of Beihang University, China. She got the Bachelor's degree from the International School of BUPT. Her current major is air traffic control and mainly carry on the research in the National Key Laboratory of CNS/ATM. She joined the System Wide Information Management program in 2010. Her R&D activities focus on the air traffic management and middleware design and realization.

Xiling Luo received the PhD degree from the School of Electronics and Information Engineering of Beihang University, China. He is now an associate professor at the School of Electronics and Information Engineering of Beihang University. He has participated in several projects activities of National Natural Science Foundation of China, National 863 Plans and Aerospace Support Foundation. His current research interests include computer vision, senior surface movements monitoring, air traffic management and ad hoc networks.

Feng Zeng received the master degree from the School of Electronics and Information Engineering of Beihang University, China. He is now an engineer of Aviation Data Communication Corporation. He was the major researcher of the System Wide Information Management program since 2010. His current research interests focus on the air traffic management, the informatization of civil aviation, the SWIM research and the coordination communication and network.