Research and Application of Data Integration in Aircraft Designing Based on SDO

Yating Yuan
College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, 210016
Email: 15850662469@163.com

Yi Zhuang and Ying Huo
College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, 210016
Email: zhuangyi@nuaa.edu.cn

Abstract—**One of the fundamental problems in complicated system environment is the mechanism of data interaction and sharing between heterogeneous data sources. The known solutions, using middlewares to unify data formats, are not reliable due to the tightly coupled architecture, poor portability and reusability, large update latency, etc. To solve this situation, we herein introduce the concept of Service Data Object (SDO) into a platform of aircraft collaboration designing, to build a data integration mechanism among the multidisciplinary applications that collaborate the designing process. We propose a Global Uniform Model Generation Algorithm (GUMGA) to automatically generate the uniform data format of the applications, using XML as a bridge among heterogeneous data sources. A case study is implemented and tested to prove that GUMGA can shield the isomerism among data sources, and provide an effective and extensible solution to deal with the data integration problem in complicated systems.**

Index Terms—**service-oriented architecture, service data object, heterogeneous data sources, data integration, aircraft design**

I. Introduction

Computer and information technology have been incessantly improved and widely deployed in the field of aviation. Applications of primary disciplines towards aircraft designing, such as dynamic analysis and design, flight mechanics analysis and design, etc., have already achieved great developments. However, aircraft designing is a complex, multi-disciplinary process, while the applications are separated from each other due to their particular design and analysis tools. The separated applications are unable to meet the collaborative designing requirements. As a result, we need to combine the multi-disciplinary applications by using unified data standard and management methods. However, the applications are usually built on different platforms, implemented in various operating conditions, and programmed using different languages, which all result in the "information island" phenomenon. Therefore, we need to establish an effective information sharing and interaction mechanism among applications of the heterogeneous systems, to offer a designing environment that handles heterogeneous data in a consistent and uniform way, so that we can eliminate the information islands fundamentally.

To build such designing environment, a lot of integration platforms for heterogeneous systems have been developed. Most of the previously known platforms perform the information sharing by middleware technology. Messages are converted to a certain format by the middleware using a special adapter, sent to the message bus, and then received by other applications by their adapters. The special adapters determine whether or not to receive these messages and conform them to their own formats[1]. In these kinds of heterogeneous systems, the adapters and message transformation rules in middlewares are tightly coupled and lack of extensibility and generality. Furthermore, they do not have a unified data format. In recent years, with the purpose of loose-coupling and expandability of isomerous systems, Service-Oriented Architecture (SOA) has been exploited[2-8]. Procedural rules are utilized to access such software that is published as a web service. It enables other programs to search and use those published web services [9]. In SOA, many data objects were offered in the form of services[4,10], which are used for information sharing and interaction between isomerous systems.

In different application areas, many scholars have applied Service Data Object (SDO) for data integration of heterogeneous systems. Jiang et al. designed the Distributed Traffic Simulation (DTS) system[11] with the architecture based on SOA and multi-agents. Simulation Road Network is persisted in Oracle database through Object-Relation Mapping (ORM). New modules are encapsulated into Service Component Architecture (SCA), and data transferred through services are packaged with uniform SDO data module to simplify the data programming in heterogeneous environment and improve the interoperability of services. The validity of the whole system is demonstrated by an experimental simulation on a given road network in real scenario. Wang et al.

doi:10.4304/jsw.9.3.689-696

proposed a data integration platform based on SDO, which had been successfully applied to the field of system equipment support [12]. The authors built the topology structure based on a data integration platform of SDO in the field of equipment application, aiming at the heterogeneities of information content, information attributes, operating system, running environment and data storing. Experiments showed that the introduction of the SDO programming framework could effectively loosen the coupling of data access operations and data using business. Zhao achieved data sharing by using data programming technology[13]. The author investigated the railway construction in information sharing platform and analyzed the existing problems. By the research on the related standards and technologies —SDO, he proposed a data exchange model in line with information sharing requirements based on SDO, and designed the data interfaces. Finally, by using SDO technology, he realized a data-exchange example on the basis of the Railway Goods Tickets System and the Railway Goods Transport Marketing and Product Management Information System, which are used to verify their solution.

Inspired by all the above work, we would like to expand and improve this technology specifically in aircraft designing field, because of the absence of a unified data integration model. On the basis of the system architecture of data integration, we adopt the idea of SOA, and encapsulate data of the separated applications in the aircraft collaborative designing system into services according to well-defined interfaces and protocols. We propose a loose-coupling, international-standard-supported, protocol-independent distributed software design patterns. Based on these patterns, we introduce the concept of SDO, propose a Global Unified Model Generation Algorithm (GUMGA), and design a uniform access interface. Our work provides an effective method to settle the heterogeneous data interaction between multidisciplinary applications in the aircraft collaborative designing field.

The rest parts of the manuscript are organized as follows: Section 2 discusses the concept of SDO. In section 3, we design the architecture of data integration of SDO in Aircraft Collaborative Designing, and propose the GUMGA. We also build the aircraft SDO model in this section. Section 4 provides an application example about GUMGA. The conclusions are given in section 5.

II. SERVICE DATA OBJECT

Service Data Object (SDO) was initially developed jointly by IBM and BEA company in a heterogeneous data integration processing specification in JSP - 235[14]. It was a kind of Application Programming Interfaces (APIs), which simplifies the J2EE development mode and pays more attention to business logic.

Instead of the necessity of being familiar with the particular APIs, we just need to access and use the data from heterogeneous data sources through different data access services. SDO can conveniently process the data from multiple data sources, including relational databases of different types (such as Oracle and MySQL), entity EJB components, XML pages, Web services, JCA (Java Connector Architecture), and JSP pages (JavaServer Pages), etc. SDO is a standard, language-neutral data object. Different from any previous data objects, it can realize the decoupling of data code and business code. There are several functional softwares involved in the aircraft collaborative designing platform, such as configuration parameter software and dynamic analysis and design software. Each functional software has its own local relational database. The data types and data presentation formats among these local databases are distinct from each other. SDO can provide a unified and standard description and a uniform access interface for the heterogeneous data sources. The architecture of SDO is shown in Fig.1[9]. The core concepts in SDO are *Data Object*, *Data Graph*, *Data Access Services* and *Change Summary*, which are defined as follows:

(1) *Data Object*. *Data Object* is the fundamental concept of SDO, which is used to be represent the business data. In this paper Data Object is specified as the data in aircraft designing field. *Data Object* describes data as a set of properties, including basic attribute values, the reference to other *Data Objects*, and the relations between *Data Objects*.

(2) *Data Graph*. SDO encapsulated *Data Object* through *Data Graph*, which has a root *Data Object*. Root *Data Object* could be obtained from the *Data Graph* through some related classes. All the *Data Objects* are associated with the root *Data Object* and the *Change Summary*. As the result, we can get the whole structure of *Data Graph* as a tree.

(3) *Data Access Services*. *Data Access Services* provide an abstract service function. Data Service Object accesses the heterogeneous data sources such as relational database, XML database, EJB, web services, etc., by different *Data Access Services* (such as EJB, XML, SQL, etc.)

(4) *Change Summary*. When a client needs to update a *Data Object*, the corresponding update records will be stored in the *Change Summary*. After the modification, the client can update to the data source accordingly by calling interface functions provided by the *Data Access Services*.

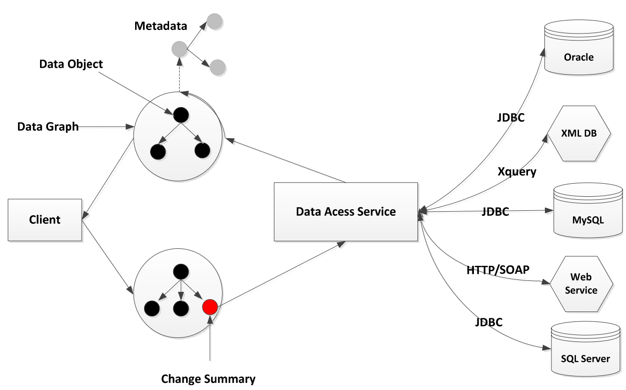

Figure 1 The Architecture of SDO

III. THE ARCHITECTURE OF DATA INTEGRATION BASE ON SDO

Every functional software involved in the aircraft collaborative designing platform handles its information by itself. The "information island" phenomenon, caused by the absence of unified data standard and management method, makes it difficult to access and utilize the local data resources, which restricts the analysis and decision of each functional software. Consequently, it is important to integrate information and provide a uniform query access interface. Aiming at this problem, we design the data integration framework based on SDO and present an SDO data access model in this section. The model can reduce the access complexity of heterogeneous data. We use XML document generated by a global unified model to shield the characteristics of local heterogeneous data sources. We also use enterprise service bus to realize the data integration and management[15]. Users can get access to heterogeneous data sources in the collaborative designing platform eventually through view interface with great convenience and flexibility. The users simply need to specify the data needed, and do not need to focus on data extract nor data synthesis and so on. Data integration system offers the function of global schema generation, query transformation and others. It integrates data from multiple data sources and then provides it to users in a uniform way. Users operates on multiple data sources as simply as a single data source.

A. The System Architecture of Data Integration in Aircraft Collaborative Designing

The aircraft data integration system architecture based on SDO that we propose can be divided into four layers, as shown in Fig.2, including *data resource layer*, *data access layer*, *logic layer* and *user layer*.

(1)*Data Resource Layer*. Data Resource Layer, which provides the data storage we needed, is located at the bottom of the system architecture. The data in this layer is consisting of the data from local databases of the functional softwares that participates in the aircraft collaborative designing, and their shared databases. A local database usually stores the specialty-critical data that is commonly used within the corresponding functional software. We can get access to the local databases through local calls directly. The data that shared by functional softwares is stored in the shared database.

(2)*Data Access Layer*. Data Access Layer provides data supporting for the upper logic layer by handling its accessing and updating requests. Data Access Layer is responsible for getting data from Data Source Layer and performing a corresponding update operation when the data is modified. It is a bridge between Data Resource Layer and Logic Layer. It is consist of modules of global unified pattern generation, data graph, metadata, query transformation, data access services, etc. When a functional software needs to interact with the main functional software or to deposit some data into the shared database, it uses the Global Uniform Model Generation Algorithm (GUMGA) to generate the standard global uniform model. The global uniform mode is described as a XML schema by translating the local model to the public model. Metadata mainly includes the location, type and name of data sources, type and constraints of data, etc. Data Access Service is responsible for connecting local databases of functional softwares and shared databases, by encapsulating the data objects into a data graph, and interacting with Data Resource Layer. Query transformation is used to localize the query/update requests from Logic Layer and to communicate with the data sources through the Data Access Service.

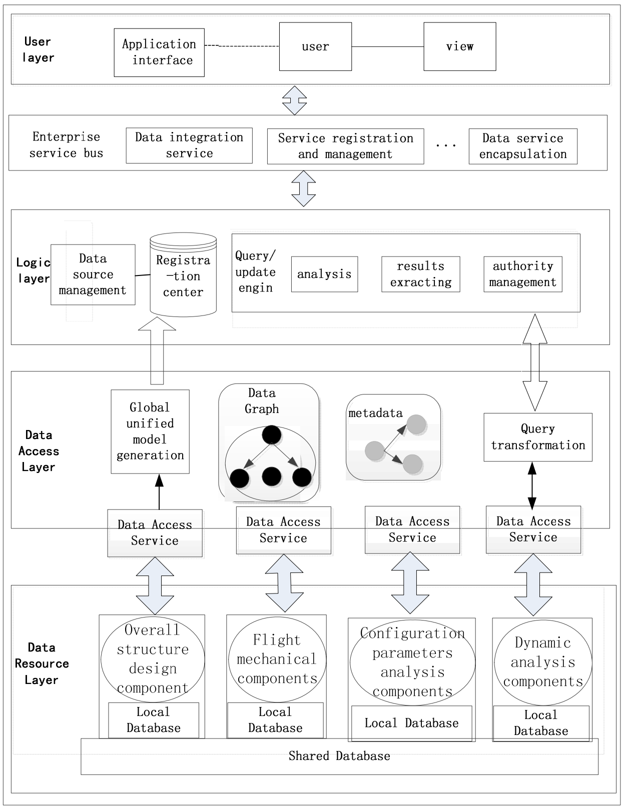

Figure 2The System Architecture of Data Integration Of SDO in Aircraft Collaborative Designing

(3)*Logic Layer*. Logic Layer is consisting of modules of query/update engine, data source management and data registry center. The query/update engine manages the customers' permissions, receives and analyses query/update requests of legal customers, and sends the execution operations to Data Access Layer, keeping the synchronization of data sources through Data Access Service. Data source management mechanism records all the data structure information about the heterogeneous data sources, manages the registry centre, and shields the difference of each data source. Data registry center receives the global uniform model from Data Access Layer.

(4) *User Layer*. User Layer on top of logic layer is the interface of user and applications, providing the access interfaces of uniform data for clients. Besides, User Layer is responsible for displaying the results from Logic Layer in a unified view.

B. The Global Uniform Model Generation Algorithm

The generation of global uniform model is the key to the data access layer, and performs as a bridge between data sources and logic layer. The goal of generating the global uniform model is to translate the heterogeneous data in the underlying data resource layer into a uniform XML schema, in order to eliminate the mismatch among heterogeneous data sources. The generation of global uniform model is the basis of accessing the heterogeneous data transparently. Because the model of local databases involved in our work is defined by different relational data models, the Global Uniform Model Generation Algorithm (GUMGA) designed in this paper is mainly aiming at the translating from the different relational data models to XML schema. GUMGA is applied in the generation of global uniform schema of the components or functional softwares in the aircraft collaborative designing field.

Literature [16] and [17] gave the formal representation of relational data model and XML schema respectively, and defined the transformation rules. However, the existing data models didn't take the safety of data in data source into consideration when describing the relational data model, and were lack of domain constraints of some relative fields. On the basis of the existing data models, We have done some extended research on the data model according to the data characteristics in the aircraft collaborative designing field. To extend the data model in the aspects of safety and domain constraint, we add a field D to control the clients' accessing, and add a turple *ch* in function Ψ to represent the domain constraints in some column type. And then we give the transformation rules between the extended data models.

Definition 1. The relational data model is defined as a six-tuple group

$$R= (D, T, C_R, P, PK, FK)$$

The detailed definitions of the tuples are as follows:

(1)*D* represents a function mapping a database *db* to detailed information *d* about *db* such that $D(db)=d$. *d* is a four-tuple group (*DBN*, *DBA*,*VN*, *VC*), whereas *DBN* is the name of data source, *DBA* is the address of data source, *VN* represents the name of access operation and *VC* represents the access password;

(2)*T* is an ordered set of table names, such that table A is before table B, if the foreign key of table B quotes from the primer key of table A;

(3)C_R is a function that maps the name of a table $t \in T$ to a column set $c \in C$, whereas *C* represents the whole column set;

(4)*P* represents a function that maps the column name c to the column type Ψ, whereas the column type Ψ is expressed as (w, k, ζ ,n, μ ,ch). Its properties are shown in Table 1.

(5)*PK* represents the prime key constrains between tables, namely the mapping from table $t \in T$ to column c $\in C$, $PK(t) \subseteq CR(t)$.

(6)*FK* represents the foreign key constrains between tables, expressed as $\{c_i \subseteq c_j\}$.

Based on the advantages of XML schema definitions in literature [16], we introduce the ideas in literature[17] and describe the XML schema as a tree structure, use *Ro* to describe the root, Le_i to represent the children nodes, *Br* to represent the tree trunks. We propose a tuple called *type*, and merge the tuple *Es* and tuple *As*, in order to support the mapping rules and complete the uniform model generated better.

TABLE 1
THE ATTRIBUTE OF COLUMN TYPE Ψ

element	Meaning	Value
w	Whether or not foreign key	~W:no; W:yes
k	Whether or not primary key referred	~K:no; K:yes
ζ	The data type of the field	Such as:char
n	Whether or not the field could be empty	~N:no, N:yes
μ	Is the field unique	~U:no, U:yes
ch	Domain Constraints	~C:no, C:yes, "check=".

Definition 2. XML schema is defined as a six-tuple group

$$S= (NS, Ro, Le, Br, type, KR),$$

The detailed definitions of the tuples are as follows:

(1) *NS* represents the namespace of the document;

(2) *Ro* is the root node of the XML Schema;

(3) *Le* represents the gathering of the children nodes in XML schema. We denote a child node by Le_i, to describe the name of the child node, the type of relevant properties, and user-defined constraint information;

(4) $Br = \{Le_1,\dots , Le_i,\dots , Le_n\}$ is the trunk consist of a set of children nodes, which are nested as a sequence, whereas Le_1 nestes Le_2, Le_2 nestes Le_3,... , $Le_{(n-1)}$ nestes Le_n;

(5) *Type* represents a function mapping the node N to the type of node denoted by $\{Em, Ar\}$, such that $type(N)=\{Em,Ar\}$, whereas *Em* represents the set of element nodes, and *Ar* represents the set of the attribute nodes. A node is expressed as $\{Le_i \cup Ro\}$.

(6) *KR* is the set of (Le_i, Le_j), whereas Le_i and Le_j represents the relationship between keyword attribute nodes and keyword reference attribute nodes.

The mapping rules from *R* to *S* are defined as follows:

Mapping rule 1. Determine the namespace of XML schema and target schema, present it as the head information of XML schema and store in the tuple *NS*.

Mapping rule 2. Map the name of database *DBN* to the root node of S, and define the node as a complextype. *DBA*, *VN* and *VC* are mapped to the attributes of the root node, and the tables in the database are mapped to the children of the root node.

Mapping rule 3. Map the ordered set *T* of table names in *R* to the *Br* tuple of *S*. Table B will be the child of table A when A is before B in *T*, namely, A nestes B in *Br*.

Mapping rule 4. Map the column information in column set *C* of *R* to the child node *Le* in *S*.

Mapping rule 5. When the value of foreign key *w* is W, it is represented by an attribution node and is deposited in *Le*, namely type (*w*) = *attr*. When the value of prime key *k* is K, it is represented by an element node and is stored in *Le*, namely type (*k*) = *elem*. A node corresponding to a primary key of the main table is expressed with the extension element attributes fref, for instance, "fref = B.id" says that the foreign key is corresponding to the primary key id of table B. The uniqueness *u* of the fields in relational data model is represented in the way of extension element attribute such as "unique = yes/no". When the value of tuple *n* is N, we represent it with “minOccurs=0”. The tuple *ch* is represented by attribute “check= ‘condition’ ”.

Mapping rule 6. Map the relationship between primary key and foreign key in the tuple *FK* of *R* to the tuple *KR* in *S*.

Based on the models and the mapping rules above, we propose the Global Uniform Model Generation Algorithm (GUMGA), which is divided into three stages shown in Fig.3. Phase one realizes the extraction of relational schema. Phase two realizes the conversion of data types. Phase 3 realizes the mapping from relational data model to XML schema. The algorithm flow chart is shown in Fig.4.

Input: the relational data model R;
Output: XML schema S.
Phase one: the extraction of relational schema
Step1: Obtain the metadata from the heterogeneous database in data resource layer through JDBC interface,
Step2: Get the data source type, source address, access name, password, etc. of the relational data model from metadata,
Step3: Read the table name T, constraint information C between tables and Columns data from the local database,
Step4: Query the database system tables for constraint definition, data column type P, Including the tuple n, uniqueness constraints μ, primary key constraints PK and foreign key constraints FK;
Phase two: the conversion between data types
Step5: If the type of datasource was Oracle, we translated the type of relational data to the type of XML schema data according to the transformation rules between Oracle relational data types and XML schema data types, else turned to Step 6;
Step6: If the type of datasource was MySQL, we translated the type of relational data to the type of XML schema data according to the transformation rules between MySQL relational data types and XML schema data types, else turned to Step 7;
Step7: If the type of datasource was SQLServer, we translated the type of relational data to the type of XML schema data according to the transformation rules between MySQL relational data types and XML schema data types;
phase three: schema mapping
Step8: the head description of XML schema;
Step9: Create the root node of the XML document corresponding to the relational schema according to the rule two,
Step10: Set up the children nodes of the root node to generate the XML schema tree structure according to the mapping rules three,
Step11: Add the attributions of every child node to enrich the tree of XML schema according to the mapping rule four and five;
Step12: Generate the global uniform standard XML schema.

Figure 3 the Global Uniform Model Generation Algorithm (GUMGA)

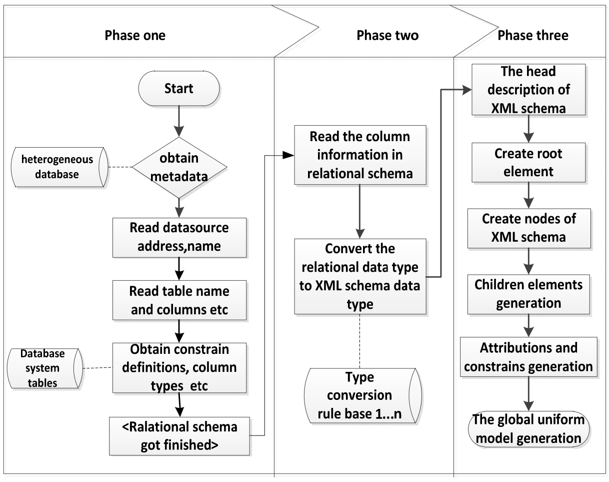

Figure 4 The Flow Chart of GUMGA

C. Service Data Object Model Towards Aircraft Designing

Introducing the concept of SDO into aircraft designing can realize the data calling from heterogeneous sources effectively through the data access services. When SDO is used in the transformation and interaction of an integrated system, it is necessary to model its data object according to the specific domain. Modeling of the data object is the first step of database design and analysis. We need to build a bridge between the real world objects and the database models.

Because of its characteristics of cross-platform and language-independent, XML can abstract the data content from the specific application environment[18]. We define the model of data object by adopting XML Schema. The model ensures high reusability of the data, which is transmitted in a uniform way between various applications of aircraft designing. In this paper, the various heterogeneous data sources in aircraft collaborative designing platform are acting as a whole on the logic level. Users do not need to know the specific location of the data and the type of the applications. The whole system is working as a virtual XML database, and the heterogeneous data source is a part of the database. On the basis of the tree structure of XML language and according to reference [19], we propose the Aircraft Service Data Object model (ASDO), which is shown in Fig.5.

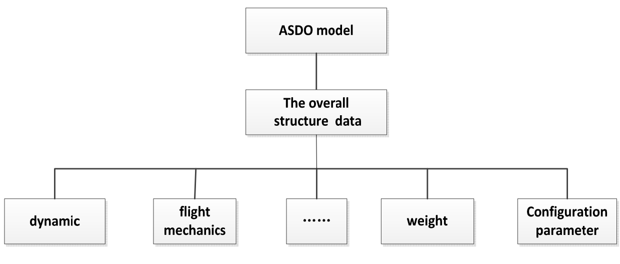

Figure 5 Aircraft Service Data Object Model

The ASDO contains dynamic data object, flight mechanics data object, weight data object, etc. As the data is stored in multiple relational databases of the

Aircraft Collaborative Designing Platform, we can convert the relational data to XML schema by using the GUMGA that is proposed previously.

Due to the limitation of space, we use several relational databases in this section as an example of the converting process: the database of the configuration parameter analysis component is stored in Oracle; the databases of flight mechanics component and weight component are stored in MySQL. We use GUMGA algorithm to shield the isomerism between the data from these data sources, and transform them to a uniform data model.

The type of data source we obtained from the local database of the configuration parameter analysis components is Oracle- RDB, the address of this data source is 192.168.1.167, the access operation name is admin, and the access password is 123456. The instantiated data table name T is shown in (1).

$$\{(General\_Situation,dynamic),(General\_Situation,load\_parameter),(dynamic,engine\ parameter\ldots)\} \quad (1)$$

According to (1), the names in T are: the aircraft general situation table, aircraft dynamic table, aircraft size parameter table, and aircraft engine table. The mapping function C_R which converts table names to data columns are shown in (2)~(4).

$$C_R(General\_Situation)=\{model,Chinese\_name,English\_name,Country\ldots\} \quad (2)$$

$$C_R(dynamic)=\{engine\_model,engine\_name,engine\_type,ngine\_number,engine\_power,model\ldots\} \quad (3)$$

$$C_R(engine\_parameter)=\{engine\_name,engine\_parameter\_model,manufacture,engine\_weight\ldots\} \quad (4)$$

The specific data tables and the main columns are shown in Fig.6. By querying the system tables in the configuration parameter analysis component database, we have the column type Ψ in generation situation table that is shown in (5) ~(6).

$$\Psi(Model)=(\sim W,K,char,\sim N,U,\sim C) \quad (5)$$

$$\Psi(Country)=(\sim W,\sim K,char,N,U,\sim C) \quad (6)$$

The column type Ψ in dynamic table is shown in (7) ~(8).

$$\Psi(engine\_model)=(\sim W,K,char,\sim N,U,\sim C) \quad (7)$$

$$\Psi(engine\_power)=(\sim W,\sim K,FLOAT,\sim N,U,check=\text{"}Cruise\ rated\ power\text{"}) \quad (8)$$

The columns type functions of other tables have similar descriptions, which are skipped here. The situation of primary key and foreign key from the database tables of the configuration parameter analysis component were shown in (9) ~(10).

$$PK=\{General\_Situation \rightarrow General\_Situation.Model, Dynamic \rightarrow dynamic.engine\_model, engine\_parameter \rightarrow engine\_parameter.engine\_name\ldots\ldots\} \quad (9)$$

$$FK=\{dynamic.engine\_model \subseteq General\_Situation.model, engine\_parameter.engine\_parameter\_model \subseteq dynamic.engine\_model\} \quad (10)$$

The type of data source we obtain from the local database of flight mechanics component and weight component is MySQL- RDB, the address of the data source is 192.168.1.172, the access operation name is scott, and the access password is abcd. The instantiated data table name *T (Load_Capacity)* is shown in (11).

$$C_R(Load\_Capacity)=\{Airmodel, maximum\_take\_off\_weight,normal\_take\_off\_weight,empty\_weight,\ldots\} \quad (11)$$

By querying the tables in flight mechanics component databases, we have the column type Ψ that is shown in (12)~(13).

$$\Psi(AirModel)=(\sim W, K, CHAR, \sim N, U, \sim C) \quad (12)$$

$$\Psi(maximum\_take\_off\_weight)=(\sim W, \sim K,FLOAT, \sim N,U,\text{"}C \leqslant 10000.00\text{"}) \quad (13)$$

The primary key is shown in (14).

$$PK=\{ Load\_Capacity \rightarrow AirModel \} \quad (14)$$

The instantiated data table name *T*(*flight_performance*) is shown in (15).

$$C_R(flight\_performance)=\{Airmodel,maxmium\_level\_speed, cruising\_speed, economic\_speed\ldots\} \quad (15)$$

By querying the tables in weight component databases, we have the column type Ψ that is shown in (16) ~(17).

$$\Psi(AirModel)=(\sim W,K,CHAR, \sim N,U, \sim C) \quad (16)$$

$$\Psi(maximum\_level\_speed)=(\sim W, \sim K, FLOAT, \sim N, U, \sim C) \quad (17)$$

The primary key is shown in (18).

$$PK=\{flight\_performance \rightarrow AirModel \} \quad (18)$$

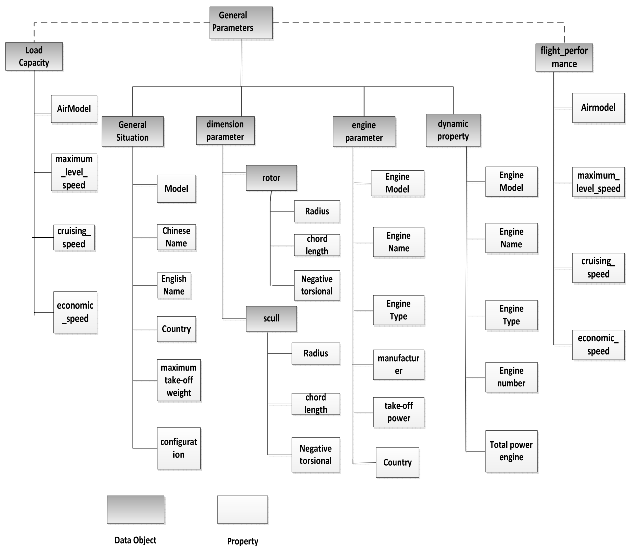

Figure 6 Configuration Parameter Data Object Model

According to the GUMGA, we transform Oracle and MySQL relational databases to a uniform XML schema, and a part of the transforming structure is shown in Fig.7.

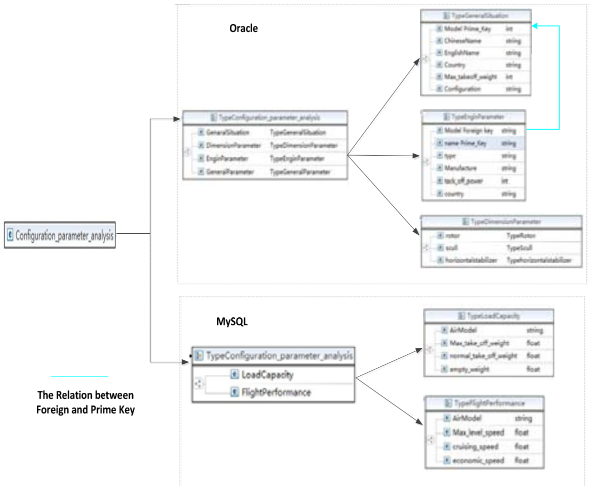

Figure 7 A Partly Architecture of XML Schema of Configuration Parameter Analysis Components and Flight Mechanics Component

XML has become the universal standard of data exchange format. In this paper, we convert the different relational data models into XML schema. Using the open source Apache Tuscany, we are able to create a global data object based on XML schema generated above, providing the functions of adding, deleting, changing, and checking for the data object. Take the XML Schema of aircraft general situation as an example, the creation process of data object is shown in Fig.8.

```
Public class SDOCreator{
  Public static void main(String[] args)throws Exception{
          XSDHelpter.INSTANCE.define(ClassLoader
              .getSystemResourceAsStream
             ("Aircraft_GeneralSituation.xsd"),null);
      DataObject helloSoapRequestDO=DataFactory.INSTANCE
         .create(http://schemas.xmisoap.org/soap/envelope,
         "EnvelopeType");
      DataObject GeneralSituationDO=
      helloSoapRequestDO.createDataObject("GeneralSituation");
      DataObject rotorDO= GeneralSituationDO. createDataObject("rotor");
      rotorDO .set("radius","18");
}
```

Figure 8 the algorithm of creating SDO based on XML

The created data object is shown in Fig.9.

```
<?xml version="1.0" encoding="UTF-8"?>
<envelope:TypeConfiguration_parameter_analysis xmlns:xsi="http://www.w3.org/
2001/XMLSchema-instance"
xmlns:envelope="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:envelope_1="http://schemas.xmlsoap.org/soap/envelope"
xsi:type="envelope_1:TypeConfiguration_parameter_analysis">
<GenaralSituation>
<rotor>
<Radius>18</Radius>
</rotor>
</GenaralSituation>
</envelope:TypeConfiguration_parameter_analysis>
```

Figure 9 the created service data object

We do not need to create the data object beforehand by the method provided by SDO. Based on the XSD document of XML schema, we are able to create the corresponding data object through the APIs of SDO. It can realize the update of data sources simply. Compared with traditional data objects, SDO can realize the decoupling of data code and business code.

CONCLUSION

Data interaction and sharing between heterogeneous data sources has always been a serious problem in data integration. The introduction of the concept of SDO is an effective solution to the data communication problem between heterogeneous data sources. XML schema is used to establish a bridge between the heterogeneous data. SDO has successfully realized the accessing of the relation data model through XML schema. Further work of this paper is to enrich and perfect the GUMGA, to perform the conversion of more complex relationship model. Furthermore, translating other types of data sources into XML schema and interacting with the relational data model will be the subject of future work.

REFERENCES

[1] Zhan P P, "Research and Implementation of a Service-Oriented Message Middleware", Xi An Dian Zi University, 2010.

[2] Zhang Y, Liu X, Wang Z, et al. "A Service-Oriented Method for System-of-Systems Requirements Analysis and Architecture Design". *Journal of software,* 2012, 7(2):pp. 358-365.

[3] Heisey C W, Hendrickson A G, Chludzinski B J, et al. "A Reference Software Architecture to Support Unmanned Aircraft Integration in the National Airspace System". *Journal of Intelligent & Robotic Systems,* 2013, 69(1-4):pp. 41-55.

[4] Deng W, Zhao H, Zou L, et al. "Research on Application Information System Integration Platform in Medicine Manufacturing Enterprise" . *Journal of medical systems,* 2012, 36(4): pp.2289-2295.

[5] Li B. "An approach to build information system based on soa and component oriented; proceedings of the Networking and Digital Society (ICNDS)". 2010 2nd International Conference on, F, 2010 . IEEE.

[6] Xie P, Rui Z. Study on the Integration Framework and Reliable Information Transmission of Manufacturing Inte-grated Services Platform [J]. Journal of Computers, 2013, 8(1): pp.146-154.

[7] Wu M, Xiong X, Ying J, et al. QoS-driven global optimization approach for large-scale web services composition [J]. Journal of Computers, 2011, 6(7): pp.1452-1460.

[8] Li Q, Zhou M. The future-oriented grid-smart grid [J]. Journal of Computers, 2011, 6(1):pp. 98-105.

[9] Wang G, Zhang Z, Xue M. "A Solution for Management Information Systems Integration Based on WebService". Electrical, Information Engineering and Mechatronics 2011. Springer. 2012: pp.1601-1608.

[10] Resende L. "Handling heterogeneous data sources in a SOA environment with service data objects (SDO)". proceedings of the Proceedings of the 2007 ACM SIGMOD international conference on Management of data, F, 2007. ACM.

[11] Jiang B, Zhang H. "Realization of Distributed Traffic Simulation System with SCA and SDO"; proceedings of the Future Information Technology and Management Engineering, 2009 FITME'09 Second International Conference on, F, 2009. IEEE.

[12] Wang P Q,Liu Z L,Tao Y.”Application of SDO in System Integration of Equipment Maintenance Field”. *Computer Engineering*, 2010, 36(14).
[13] Zhao Y S.”Reserch on data exchangeing interface based on SDO of railway information sharing platform”. *Beijing: Beijing jiaotong university*, 2008.
[14] Huang G P. Guo S Z. Chen H Y.”A Survery of Service Data Object”. 2009 International information technology and applicationcollectedpapers, 2009.
[15] Liang A H. “The detail of SOA thoughts,technology and system integration and application”. *Publishing House of Electronics Industry*, 2007.
[16] SHI Ye-ling, LI Jian-hui. “Universal Model of Mapping from Relational Database to XML Schema”. *Computer Engineering*, 2009, 35(7).
[17] HOU Ying, LI Fengqi, NIU Jizhen,et al.”Algorithm to convert from relational schema into modularized XML schema”.*Computer Engineering and Applications*,2011,47(12):pp.122-125.
[18] Joshi P, Imadabathuni M, He D, et al. “Application of the condition based maintenance checking system for aircrafts”. *Journal of Intelligent Manufacturing*, 2012, 23(2): pp.277-88.
[19] Zhang C L, Guo C G. “The general design of helicopter”. *National Defendence Industry Press*, 2006

Yating Yuan was born in 1990, received the B.S. degree in Computer Science and Technology. Now she is a Master candidate of Nanjing University of Aeronautics and Astronautics, Jiangsu, China. Her research interests include information security and network and Distributed Computing.

Yi Zhuang was born in 1957. She is a professor and doctoral supervisor of Nanjing University of Aeronautics and Astronautics. Her research interests include network and Distributed Computation and Information Security.
E-mail: zhuangyi@nuaa.edu.cn

Ying Huo was born in 1988. She has gotten a B.S. in information security. Now she is a Ph. D. candidate of Nanjing University of Aeronautics and Astronautics, Jiangsu, China. Her research interests include information security and trustworthy service.