Advances in Engineering Software 118 (2018) 27–34

Contents lists available at ScienceDirect

Advances in Engineering Software

journal homepage: www.elsevier.com/locate/advengsoft

Efficient approach to database integration for an aerospace vehicle design and certification framework

Aye Aye Maw, Tun Lwin, Maxim Tyan, Jae-Woo Lee*, Sangho Kim

Department of Aerospace Information Engineering, Konkuk University, Seoul, South Korea

ARTICLE INFO

Keywords:
Data integration
Database integration
Data schema wrapper program
Integration of databases and legacy program
GUI
Aerospace design framework

ABSTRACT

The integration of distributed data sources is one of the main problems of engineering software. The data integration process for a heterogeneous legacy system is a key aspect of the development of a computerized system and in the integration of a design framework. In this research, our approach to data integration focuses on developing system-building techniques for efficient data integration queries. Keyword-based data searching is investigated and applied within a database for a design framework. A database table connector (DTC) wrapper program is implemented based on the use of data integration processes and keyword-based searching. The DTC provides data integration for various data resources from legacy programs and database management systems using SQL querying. The DTC enables designers and developers to rapidly and efficiently develop integration frameworks for different data resources. This paper also describes the implementation and deployment of the Certification and Aircraft Design Integration System framework, which integrates various analysis and optimization codes, computer-aided design software and database management systems. Multiple data types are used within the framework, including databases, spreadsheets, flat files, XML files and personal data management. Several aircraft design and optimization problems are successfully solved using the developed framework.

1. Introduction

Data integration is the process of combining data residing in different sources and of providing the user with a unified view of these data [1]. Data integration is crucial within a large enterprise with numerous data sources. For large-scale scientific projects in which data sets are produced independently by multiple researchers, more progress and better cooperation among government agencies is essential, since each agency has its own data sources. Data integration aims to satisfy the needs of various clients or users and to determine all data pool patterns accurately by designing a middleware product or methodology. One of the main motivations for data integration is that data often reside in different, heterogeneous and distributed data sources, which are typically processed by legacy systems. Data integration systems have the ability to present an integrated, mediated schema for query by the user, translation or reformulation of the query to combine information from the various data sources according to their relationships with the mediated schema, and execution of the query over various local and remote data sources. Data integration systems [2] provide transparent access to a collection of data stored in multiple, autonomous and heterogeneous data sources. Sources may range from database systems and legacy systems and windows application form, web, web services and flat files.

In the process of coordinating multiple database systems, critical problems can occur, since the databases may have been designed using different models and objects [3]. Furthermore, there may be identity, representation and scope conflicts, for example. Prior research has introduced a new approach to interoperable database systems which relies on the concept of context interchange; this framework focuses primarily on resolving the schematic and semantic conflicts [17]. Another approach resolves structural conflicts in the context of multi-database schema integration [15]. Many researchers have studied the conflicts and integration of heterogeneous database systems; however, there is still no common methodology for resolving conflicts in and integrating multiple-database problems. In fact, few of these studies have emphasized the integration of databases and legacy systems. In legacy systems, the semantics are hidden and difficult to decode; data is also stored in flat files, which is a very different scheme to those used in database management systems. Various research communities (databases, artificial intelligence, semantic web) have been developing and addressing issues related to data integration from different perspectives. Many approaches have been proposed which provide classifications based on comprehensive and shared criteria.

One of the most widely studied architectures for data integration is Mediator [4]. Mediator builds a mediated schema as a synthesis of the

* Corresponding author.
E-mail address: jwlee@konkuk.ac.kr (J.-W. Lee).

https://doi.org/10.1016/j.advengsoft.2018.01.001
Received 20 July 2017; Received in revised form 1 November 2017; Accepted 7 January 2018
0965-9978/

source schemas to be integrated. A mediated schema allows the user to establish a query based on a global perception of the handled information. Two basic approaches for specifying mappings in a data integration system have been proposed in the literature; these are known as global-as-view and local-as-view. In the global-as-view approach, the contents of the mediated schema are expressed as queries of the sources; in the local-as-view approach, the data sources can be described as views of the mediated schema. The main problems of the global-as-view approach are related to the update of the mediated schema [5]; if sources change, the mediated schema may also change, and may exhibit side effects for the applications that refer to it. However, several issues related to query rewriting arise in building a query manager for local-as-view systems. Several systems have been developed using these approaches. The TSIMMIS system [6,7] was one of the first data integration systems to follow a "structural" approach for building a global-as-view virtual view of structured and semi-structured data sources. Garlic [8] builds a wrapper-based architecture to describe the local source data using an object-oriented language. The main component of Garlic is a query processor which optimizes and executes queries for diverse data sources established in an object-extended SQL.

Matching is a basic problem in many database application domains such as heterogeneous database integration, e-commerce, data warehousing, and semantic query processing. Typically, schema matching is performed manually and is supported by a graphical user interface (GUI). Mapping is defined as a set of mapping elements, which indicate that certain elements of a schema are mapped to another element. Each mapping element may have a mapping expression that specifies how those elements are related [9]. Integrated information systems provide users with a united view of multiple heterogeneous data sources; the actions of querying the underlying data sources, combining the results and presenting the results to the user are performed by the integration system. Entity resolution is related to identifying the same object in different databases.

The Database Table Connector (DTC) is proposed here in an attempt to overcome the abovementioned problems. It was developed by focusing on providing a solution for data integration for different data resources from legacy programs with Database Management Systems (DBMS) using SQL querying [10]. It can enable developers or designers to rapidly and efficiently develop various data resource integration frameworks. Furthermore, this approach resolves not only schema and semantic heterogeneities but also conflicts between different query languages. This paper also shows that the Dassault CATIA-API wrapper program effectively performs the data integration process in an analysis of design programs with CAD modeling.

2. System architecture of the aerospace vehicle design and certification framework

The Certification and Aircraft Design Integration System for Small Aircraft (CADIS–SA) is an integrated multi-user framework for a conceptual design and certification compliance check of a small aircraft. The framework has a user account management system, which provides efficient data management for each user at each stage of the aircraft's design. Fig. 1 shows the structure of the CADIS-SA framework. The design process of an aircraft starts with a requirements analysis. Quantitative design and certification requirements are converted into qualitative design requirements. Preliminary sizing and multi-disciplinary aircraft analysis codes are implemented using the MATLAB software package. The basic parameters of an aircraft are calculated at the preliminary sizing stage, using a set of design requirements. An initial configuration is then generated using the Aircraft Configuration Designer (ADC), a program for the automated generation of an aircraft configuration implemented in Dassault CATIA CAD software. The CAD file of the initial aircraft configuration generated by this process is then analyzed, and detailed configuration data are supplied to the Aircraft Design and Synthesis Program for Small Aircraft (ADSP-SA). The main program of ADSP-SA is implemented in MATLAB, with additional modules written in FORTRAN and Java. ADSP-SA is composed of strongly coupled analysis and optimization modules. The final optimum aircraft configuration is then generated using ACD. Configuration data for the updated aircraft are then transferred to the load analysis stage. The calculated loads based on certification regulations are then used at the detailed design stage.

The CADIS-SA framework is composed of several analysis and optimization programs, automation scripts, databases and file storage. An efficient approach for the integration of databases with legacy programs and dynamic data processing is required for an advanced aerospace vehicle design certification framework.

2.1. Program architecture of the framework

The framework comprises five main program components: a requirements analysis, preliminary sizing, ACD [13], ADSP-SA [11], load analysis, and the certification database. The GUI and framework integration is implemented using Microsoft .Net Framework 2008 and VB language. Fig. 2 illustrates the program hierarchy of the framework .The framework enables the saving and loading of the analysis data, load default parameters, and the execution of different analysis modules either separately or in a sequential way. Default data for two- and four-seater aircraft are provided by the framework; additional aircraft data can be also added. Design and certification requirements, competitive aircraft data and the mission profile are analyzed at the requirements analysis stage in order to generate a set of new requirements for a future aircraft. The basic geometry parameters of the wings, tail and fuselage are calculated at the preliminary sizing stage using this set of requirements.

The initial configuration is then generated in the CATIA CAD program using the Aircraft Configuration Designer (ACD) tool [11]. ACD is written in the Visual Basic for Applications language, using the standard CATIA API. ACD runs both in an unattended mode, in which the software automatically reads the result of initial sizing or Multi-disciplinary Design Optimization (MDO), and generates the CAD and text files with additional configuration data, and a GUI mode, when the designer provides all input manually using the user-friendly interface. A high level of uncertainty exists at the initial sizing stage; thus, ACD is performed in GUI mode in order to generate the preliminary configuration. The main wing and tail platform shapes are generated to match the results of the initial sizing. A detailed fuselage geometry is generated manually to match the desired length and diameter. The CAD data is then analyzed using ACD and transferred to a next stage. Detailed configuration data, such as the fuselage frontal and wetted area, the wing wetted area and so on, are required as input for the more sophisticated design stage. ADSP-SA is composed of multiple analysis modules integrated in a MATLAB environment. ADSP-SA covers geometry, mass and balance, aerodynamics, piston engine, propeller, stability and control, performance and mission analysis. ADSP-SA uses the set of design requirements and the detailed configuration data of the preliminary aircraft configuration to run the optimization in MDO mode [12]. Each analysis can be performed separately using the GUI provided by CADIS-SA, giving an output of all the required data in text-based and graphical forms. The data integration in different analysis module is managed by the CADIS–SA framework. Fig. 3 shows the program structure of the framework. The user can evaluate the results of a previously executed analysis or can manually input the required values. For example, performance analysis requires the input of mass data, which can be calculated using the mass and balance module for a given aircraft configuration, or can be provided directly by the user. The load analysis program is used to calculate the structural loads, based on certification regulations for this class of aircraft known as the FAR-23 airworthiness regulations. The load analysis of an optimum aircraft is performed for various conditions, including flight loads, critical loads and ground loads. The FAR-23 and Korean Airworthiness