DATA EXTRACTION AND TRANSFORMATION WITH FLAT FILE FOR BUSINESS INTEGRATION

Sheng YE, Wei SUN, Zhong TIAN
IBM China Research Lab

Keywords: Flat file, XML, data extraction, data transformation, application integration

Abstract: With the development of e-business, the capability of exchanging data in different formats is necessary for integrating heterogeneous enterprise applications. Though XML is becoming the standard communication protocol over the Internet, most enterprise applications today can only process a specific format text data, mostly in a flat file. This paper introduces a round trip transformation technology between flat file and XML, Flat File Adapter. This technology employs a systematic data extraction and formatting method to support the processing of complex format flat file. By using Flat File Adapter, developer can quickly design templates containing the data transformation rules. It can be easily updated when requirements change later. In this paper, we introduce the system architecture, detailed components, and particular data extraction and transformation method. Finally, a sample application in B2B e-procurement solution is also described.

1 INTRODUCTION

Data exchange is the most widely applied business-to-business integration (B2Bi) pattern. Data exchange B2Bi is effective because it is simple in concept and has been in use since the days of Electronic Data Interchange (EDI).(Andre, 2001) Different application has its own data management method and in-house data format. Although Extensible Markup Language (XML) can serve as a universal data communication language, most business applications can only produce and process flat files. Flat file is defined as data with fixed format rule and typically encoded as printable characters. We can divide the flat file in three types according to their format characteristics. The first type has strict space position rule, for example, the recognized results by scanning traditional tables or forms. The second type has strict logic rule, for example, inquiry results list of a database, EDI messages, SAP IDOC. The third type aggregates both logic and space rule, for example, the general reports used for reading, transmitting or printing. To enable the flat file based documents to exchange between heterogeneous enterprises' backend applications, data extraction and transformation between different types of flat file and XML is in great demand.

In prior arts, developers design specific adapter from scratch for each possible application to be integrated with, and it's difficult to update when application or data change. Such a data exchange technology is inefficient and wastes time and human resources. Unidex XML Converter could transform flat file into XML, and vice versa by process a same template based on XFlat schema(Unidex, 2003). XFlat schema demands that flat file consists of records with delimiter or with fixed length. XML Converter is mainly designed to process flat file with delimitated format. Additionally, Zamora devised a data transformation method to analyze data files by using the structural, syntactic and semantic knowledge about the data files(Zamora, 1990). This technology is particularly appropriated to extract information from business correspondence documents. Richard and Shin-Ywan have respective patent on data extraction from printed form or image but not flat file(Casey et al., 1992)(Wang et al., 1998). These existing solutions can only handle some types of flat file, especially for data format has strict logic rules. Flat File Adapter (FFA) technology is a template driven dual-way engine for data extraction and transformation between flat file and XML. FFA features flexible data markup, extraction and output formatting mechanism for flat file and XML, it also features data fusion and processing extension mechanism, which make FFA differentiate with the existing techniques.

2 SYSTEM ARCHITECTURE

As depicted in the figure 1, a data extraction and transformation system comprises the data locating component, the format mapping component, the data extracting and the data transforming component.

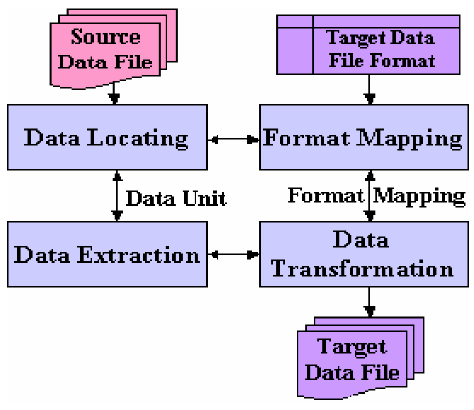

Figure 1: FFA Architecture

The data locating component generates the location descriptions of data units. The format mapping component builds up the correspondence between the data units and the target format. The data extracting component extracts the data which position has been determined. The data transforming component transforms the extracted data into data in the target data file. According to the architecture, FFA is invented to facilitate the transformation from flat file to XML and vice versa. FFA includes three core function components, Flat File Reader(FFR), Flat File Writer(FFW), and Extension Controller. According to the predefined data extraction and transformation rule, the FFR could transform flat file to XML. The FFW could transform XML file to flat file. The responsibility of Extension Controller is to invoke external extension programs during the transformation process to fulfill complex data processing.

Two different types of template, FFR and FFW, define the data extraction and transformation rules. FFR introduces multi-dimension data extraction and transformation technique to support complex format flat file processing. According to the difference of the problem addressed, FFR and FFW don't adopt symmetrical template.

Developer designs a FFR or FFW template mapping source flat file to target XML file or vice verse. FFA loads the corresponding template, reads a source flat file or XML, performs the extraction, and transforms to target XML or flat file according to the loaded template definition.

3 FUNCTION COMPONENTS

3.1 Flat File Reader

FFR engine comprises data extracting unit and data transforming unit. The FFR template design covers data locating and data format mapping. FFR Engine interprets a FFR template and applies to an input flat file, extracts required data and transforms into designated objective format XML.

- **Data Markup and Locating**

According to the FFR template design, the data in a flat file is identified by predefined different data units. The "data units" are located by location description and may include "sub data units".

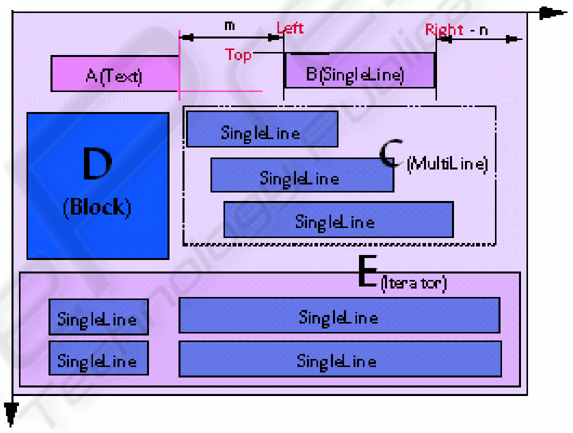

Figure 2: FFR Data Markup and Locating

As shown in figure 2, the "data units" mainly consist of five types, that is, the "Text", "SingleLine", "MultiLine", "Block" and "Iterator". In particular, the "Text" represents the string capable of being located and matched. The "SingleLine" represents the defined data unit is arranged in a line. The "MultiLine" represents the defined data unit is arranged in multiple lines consisting of plurality of single lines. The "Block" represents a rectangular area in the file. The "Iterator" represents the data unit arranged in such a manner that the data unit includes several sub data units with same form feature and iteratively presenting. For example, the data unit E in figure 2 is defined as an "Iterator" data unit, including the sub data units "SingleLine" having the same type and iteratively presenting. The "Iterator" is used to define the list data.

After the data units are defined, the position descriptors of data units need to be determined by the location elements, except that the "Text" data unit is directly defined by string matching. By design, four location elements are used to determine the position of a data unit, that is, the “Top, Bottom, Left, and Right”, representing the uppermost position, lowest position, most left position, and most right position of the data unit respectively.

Each location element has several attributes, they are: “Base” which is another data unit as a location reference; “From”, which is a position in the “Base” data unit for referring; “Skip”, which represents the offset of the location element from the “From” position. The positive offset “+N” represents moving N columns/rows in the down/right direction; the negative offset “-N” represents moving N columns/rows in the up/left direction.

The value of the attribute “Base” may be the ID of a “data unit” having been located, such as the ID of the data unit having the type of “Text”, or “the start of a row (RB)”, “the end of a row (RE)”, “the start of a column (CB)” and “the end of a column (CE)”. Here row or column refers to the row or the column of the “Base” currently located. The “Base” may also be an original point in an absolute coordinate.

For example, as show in figure 2, the data unit A may be defined as a “Text” data unit by string matching. The location element “Left” of the data unit B may be defined by using the attributes: Base=”A”, From=”End”, Skip=”+m”. The data unit B may be described by the basic attributes of the location elements, when using XML language based on the FFR Template XML schema, as followings:

```
<SingleLineSpan>
    <Top BASE=“A”, from=“Start”/>
    <Left BASE=“A”, from=“End”, SKIP=“+m”>
     <Right BASE=“RE”, from=“Start”, SKIP=“-n”>
</SingleLineSpan>
```

- **Data Format Mapping and Transformation**

According to the FFR template XML schema design, each data unit location element has attributes "TargetTag" and "PrintFormat", which would be used to formulate the output document structure. Through this approach, FFR can put extracted data into XML file with predefined format.

Attribute “TargetTag” defines the XML tag name of the data in the objective XML file. Attribute “PrintFormat” determines the format of extracted data which is output into target XML, such as keeping or trimming space and line break.

FFR output simple structured XML. Through XML transformation technology, the simple FFR output can be transformed to standard conformance or any complex structure, such as CXML, RossetaNetXML.

3.2 Flat File Writer

Data exchange is always round trip. Many legacy systems require data conforming to some specific format, such as SAP IDOC, CVS or form style file. XSLT can transform XML text file. But XSLT is very weak in describing space position.

FFW is designed for generating plain text with strict position requirements. Table 1 illustrates different types of transformation rules of FFW template.

Table 1: Flat File Writer Template Elements

Type	**Elements**			
Retrieve	Data	ForEach	Variable	
Output	Text	Row	Cell	
Logic	Switch	Loop		
Structure	Head	PageNum	Body	Tail

“Retrieve” type elements can be used to retrieve data from source XML using XPath expression. As shown in figure 3, “Output” and “Structure” type elements can work together to facilitate output with specified position and generating required page structure. This enables the user experience of design FFW template is just like designing HTML form.

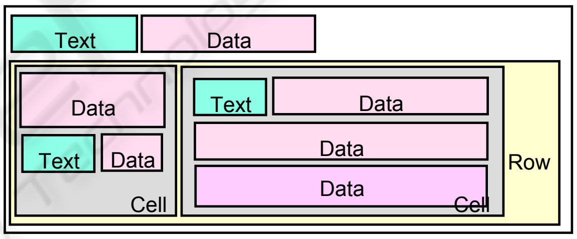

Figure 3: Flat File Writer Template Structure

“Logic” elements can describe complex rule of data retrieving and output, it facilitates the dynamic generation of output according to input instance data.

3.3 Extension Controller

Though FFA template is designed to cover most transformation pattern, FFA does not include semantic rules on data. The extension mechanism of Flat File Adapter can help in this situation.

Extension is a piece of code developed by user. The extension code can perform a specific complex processing on retrieved data and return the result to the FFA to fill into target. The FFA template defines extension type, extension URL, and reference name, so as to let FFA engine reach and invoke the extension code. Extension code can be called anywhere in the template. The input arguments of extension usually are data extracted from source file.

3.4 Advantages of Flat File Adapter

Flat File Adapter can be used to bridge XML with flat file, which is a common requirement of business integration application. FFA technology introduced in this paper provides the following advantages:

- FFA provides a complete solution for round trip transformation between flat file and XML.
- FFA employs pattern matching plus two dimensional position data locating method. This facilitates the data extraction from data sources with diverse layout format.
- FFW template combines the user experience of XSLT, XPath expression and HTML form design together, makes it has low study barrier.
- The transformation logic is separated from engine. This makes the transformation rule easy to design and flexible to change.

To those applications which can only process flat file, the FFA application would be a great choice for fulfilling XML based data exchange without need to be upgraded to XML capable system, which would help to protect the existing IT investment.

4 FLAT FILE ADAPTER APPLICATION

FFA technology has been included in the Data Fusion component of our business integration platform, eDocXchange. It has been successfully deployed in several production systems. One of the sample applications is described as follow.

IBM China Procurement Center (CPC) serves as general and product procurement brokering house for 4 IBM manufacturing joint ventures in China. The e-procurement (e-Proc) solution for CPC is a lightweight solution, supports ordering and invoicing functions for 4 JVs. Each JV has its own ERP/MRP systems. The e-Proc system connects the 4 JVs and suppliers to CPC on the Internet.

To reduce the investment and impact to exiting enterprise backend system, JVs prefer to export their Purchase Orders as flat files from their ERP and upload into e-Proc system. The FFA component on the e-Proc system combines information extracted from the flat files with information from other sources (e.g. Buyer profile, trading partner profile) to formulate a formal PO in XML/EDI format.

Suppliers can connect to the e-Proc system through messaging systems, or directly review and process messages from CPC on the Web with a browser. PO Acknowledgment (PO ACK, either accept or reject) against PO, PO Change, PO Cancel and Invoice against PO, PO Change ban be sent back to CPC.

Each document from the ERP systems has a different format. As most JVs were suffering major business transformations, e-Proc system has to survive a number of major file format changes as the JV's adjust their order printout or switch from one ERP to another as part of overall IT transformation. This actually prompted our design of the FFA. The successful experience demonstrated that FFA is an effective and lightweight approach and solution for rapid and cost effective business integration.

5 SUMMARY

Data exchange is the most effective business integration pattern. XML has been becoming the B2B transaction protocol, however most enterprise legacy systems can only handle specific flat file. To enable the effective data exchange between the heterogeneous enterprise applications of different trading partners or even different divisions within one enterprise, a flexible and easy solution for transformation between flat file and XML is in great demand. Flat file adapter has been designed to tackle this problem. It provides the capability of round trip transformation between XML and flat file. The system architecture and core function components, Flat File Reader, Flat File Writer, Extension, have been introduced in detail. One real application sample in B2B e-procurement solution is presented also to demonstrate how this technology can be used in a business solution. This technology's advantages have been summarized in this paper.

REFERENCES

Andre Yee, 2001. E-Business Integration pattern, Retrieved Oct. 8, 2003, from http://www.informIT.com

Unidex Inc., Overview of XML Convert and XFlat, Retrieved Oct. 8, 2003, from http://www.unidex.com/

Zamora, Elena M., 1990. Computer method for automatic extraction of commonly specified information from business correspondence, US Patent 4,965,763

Casey, Richard G., Ferguson, David R., 1992. Computer-implemented method for automatic extraction of data from printed forms, US5140650

Wang, Shin-Ywan; Yagasaki, Toshiaki, 1998. Feature extraction system for identifying text within a table image, US5848186