

Redvers Consulting Ltd

White Paper

**Give your COBOL applications
fast efficient access to the
world of XML and web
services with the Redvers
COBOL XML Interface**





Executive Summary

Due to its phenomenal success, XML enablement is no longer an option for today's computer installations.

XML adoption has been driven largely by the need to standardize on an open, self-documenting data format, understood by all. Already, the majority of data entering and leaving computer installations is in XML form.

If the trend towards Service-oriented architecture (SOA) continues, even applications within the corporate structure will communicate in XML. According to analyst firm Gartner, SOA will be used in more than half of new mission-critical applications and business processes designed this year, and in more than 80% by 2010.

Rather inconveniently, the majority of the world's business data is still held in COBOL format. To further aggravate the situation, COBOL is one of the least XML enabled data processing languages in use today.

Converting data currently held in databases and files from COBOL to XML format isn't a viable option. Rewriting vast amounts of business logic held in the COBOL application programs that build and maintain this data would be an expensive undertaking, providing little business benefit. XML is also more verbose than COBOL, typically requiring 3 to 4 times the storage space.

The preferred solution therefore, is a simple, efficient and scalable process to convert COBOL to and from XML as required.

Introduction

This Document

This white paper looks at the issues relating to the integration of computer data held in COBOL and XML formats.

Detailed knowledge of COBOL and XML is not required but a basic understanding of the formats would be an advantage.

Different customers have very different processing requirements and restrictions when considering COBOL XML integration. For this reason a summary of the various options for COBOL XML integration are outlined before examining the Redvers COBOL XML Interface in detail.

Finally, package pricing and ordering procedures are discussed.

COBOL and XML

Developed in 1959, COBOL (Common Business Oriented Language) has evolved from a very different computing environment to that of today. There is a misconception that code written a long time ago must be tired, worn out and slow. In fact it is the very constraints of 1950's technology that lead to a language obsessed with efficient use of computer resources.

In contrast, XML (eXtensible Markup Language) was developed by the W3C (World Wide Web Consortium) in 1998. This was the era of the dot com boom, with powerful computers, huge bandwidth and seemingly unlimited IT budgets. XML isn't so much a language but more of a universal format to describe structured data and documents.

COBOL and XML Formats

Figure 1 shows how COBOL data is organized into records containing a consistent sequence of fixed length fields.

The name, position, length and data type of each COBOL field cannot be derived from the data itself. This information is held separately in COBOL record definitions written in the application programs. Numeric data is usually held in packed or binary form.

If multiple sets of details exist for a given key, these details are placed on subsequent records carrying the same key. The last field in the data structure (**AccountHolder**) appears on the *right* of the first record.

Figure 2 shows how the same information would be represented in an XML document. Here, the data is held in variable length elements, in variable file positions.

Each XML data element is delimited by start and end tags. These tags, not only name the element but also show its position in the data structure. XML data is held in readable character form.

If multiple details exist, these are placed in the element below. If no details exist, the element can be suppressed entirely. The last field in the data structure (**AccountHolder**) appears at the *bottom* of the document.

Figure 1: COBOL Data Format

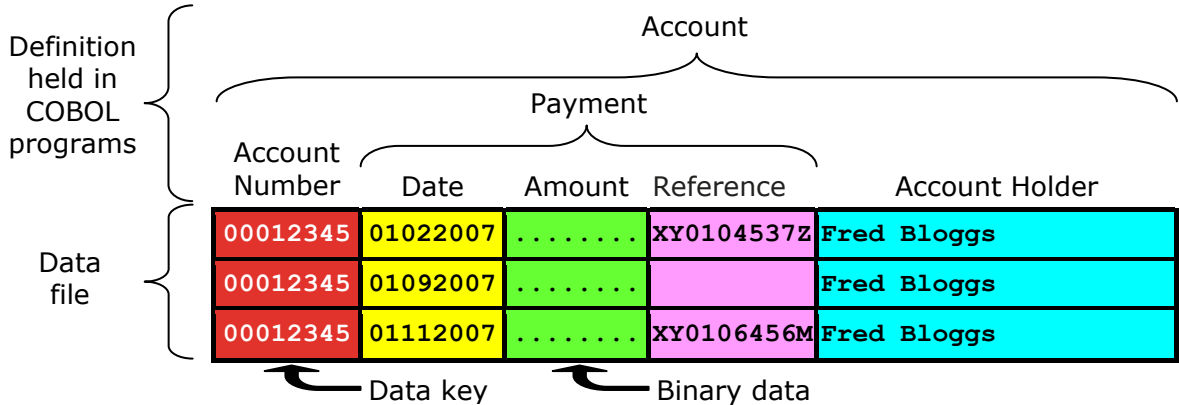
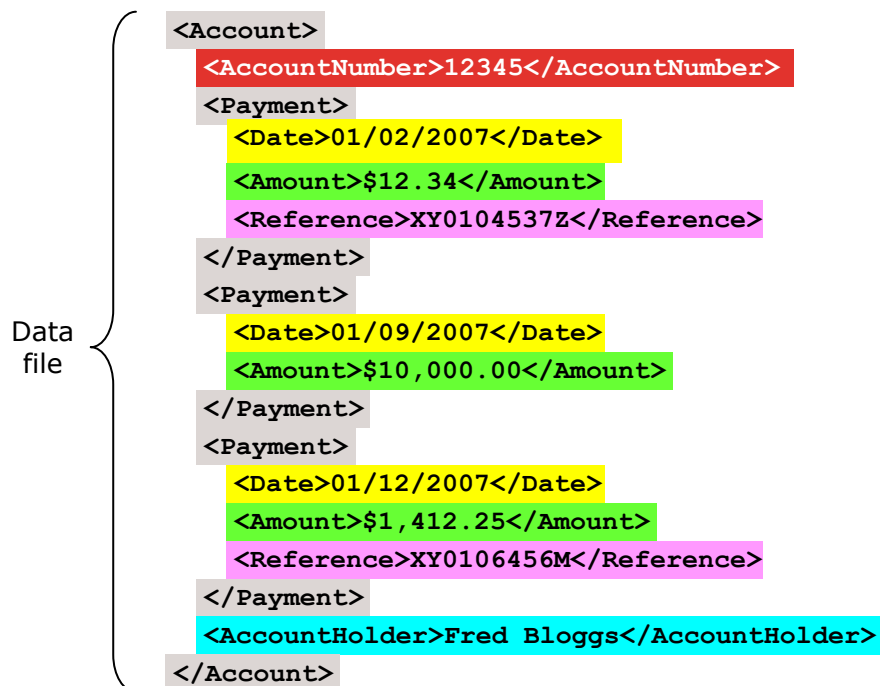


Figure 2 : XML Data Format



The Options

Writing Bespoke Programs

This is probably the most expensive option for COBOL XML conversion. As shown in the previous section, it's not easy to convert COBOL to or from XML and bespoke COBOL programs that do this are usually very complex and resource hungry.

The COBOL language is designed to process record structures from top to bottom using fixed length fields held in fixed positions in storage. However, XML conversion requires a left to right view of data structures as well as character manipulation in variable storage locations.

Companies that take the bespoke route, typically require a dedicated team of programmers assigned to writing and maintaining an ever growing suite of application specific XML conversion programs. This results in inconsistent methods of XML generation and parsing across the business.

Databases

Databases often provide some very good tools for generating and parsing XML. But what if your data doesn't reside on a single database? Loading your XML enabled database with COBOL data from other files and databases, just to unload them again in XML, wastes time as well as disk space. Databases are also expensive to license and maintain.

Language Extensions

Some vendors in the COBOL market offer XML enabled compilers with extensions to the base COBOL language. These extensions may be in the form of special file definitions or use GENERATE and PARSE statements. The problem with this approach is that these extensions don't usually support every aspect of the XML language. You may be able to design your own internal applications in such a way as to avoid this missing functionality but how do you tell customers and partners not to send you perfectly valid XML documents because you don't have the software to process them.

Server Based Interfaces

COBOL XML interfaces that run on server platforms have the advantage of relieving the host machine of the task of performing the actual COBOL XML conversion. For some installations this can be a critical factor in selecting a COBOL XML interface.

Disadvantages of server based interfaces include the installation and maintenance of new software as well as the investment in skilled personnel required to support it. There will also be the task of downloading/uploading the COBOL format data to/from the server, with all the data integrity, scheduling and security issues that arise from such a procedure.

Host Based Interfaces

Host based interfaces need no downloading or uploading of data and require minimal new skills. Most use COBOL definitions to map the data to be translated and some even use COBOL programs to perform the actual conversion.

Using COBOL definitions and programs has many advantages over the other options. It ensures that COBOL data is consistently defined once across the application and interface functions. It also integrates the application and interface under the same set of performance rules and limits. Perhaps the most important advantage is that installations requiring a COBOL XML interface would normally be running COBOL applications already and will therefore have staff familiar with COBOL programs and definitions already on site.

However, a potential problem exists in host based COBOL XML interfaces. Many of these products attempt to store all the data for an XML document in a single COBOL definition. Not only can this lead to very large storage overheads but COBOL always places a specific limit on the number of repetitions of a given data structure. Conversely, XML designers are accustomed to the freedom of unlimited element repetitions.

What's Special about the Redvers COBOL XML Interface?

A Host Based Interface

The Redvers COBOL XML Interface is a host based interface. This means:

- it doesn't require a team of programmers to maintain it;
- it's less expensive than a database;
- it supports the full W3C definition of the XML language;
- it doesn't involve transferring data to external platforms.

Like other host based interfaces, it uses a COBOL definition to define the data to be translated. This provides a single common view of how the data is defined across the interface and application programs.

Unlike most other host based interfaces, the Redvers COBOL XML Interface operates at record level, intelligently building all necessary XML element structures when generating and correctly interpreting XML structures when parsing. This approach replaces complex element level application logic with a single CALL statement that passes the next logical record to/from the application.

Source Code

Unlike all other host based interfaces, the Redvers COBOL XML Interface is delivered in COBOL source code form. It is installed simply by copying the code into the customer's source code library and running the standard compile/link job. This produces a client application with a perfectly integrated interface, due to the fact that application and interface programs are all compiled under the same standard COBOL compiler.

Another advantage to delivering software in source code form is that it makes our interface available to any computer running a COBOL compiler. The Redvers interface is currently running on **iSeries/AS400, UNIX, HP, CA-Realia, Fujitsu Siemens BS2000, Micro Focus and IBM mainframe** platforms.

Intellectual property held in the interface source code is protected using an in-house written COBOL obfuscator known as the *Redvers Cloaking Device*.

Repeating Data

The Redvers COBOL XML Interface solves the problem of handling elements that repeat a variable number of times by allowing the application to pass data to/from the interface in multiple CALL statements. This technique is similar to the way COBOL applications usually read and write data to/from files and it enables the interface to build or receive XML documents of almost infinite length, using only a small storage area.

The Redvers interface not only takes care of all the character manipulation but also the task of re-orienting the data to a left-right perspective, rather than top-down - see *discussion on COBOL and XML Formats on page 3*.

Technical Details

The Redvers COBOL XML Interface is an off-the-shelf product consisting of a generator subroutine and a parser subroutine. Application specific XML documents are generated or parsed using a COBOL copybook which consists of definitions for each COBOL field. The cross-referencing of COBOL names to XML tag names is provided by comment lines in the copybook, containing the XML tag names.

The interface is available in the following levels: "**Batch**", "**Message**", "**CICS**" or "**Standalone**". An additional "**Superfast**" level is also available for high transaction volume users.

The generated well-formed XML standalone document conforms to the World Wide Web Consortium (W3C) [Extensible Mark-up Language \(XML\) 1.0 definition](#).

XML generation rate is **7.8 megabytes per second**; parsing rate is **4.3 megabytes per second** (timings were performed on an IBM zSeries mainframe running z/OS 1.10). Maximum document size is **99 megabytes** (except "**Batch**" level which has no limit).

Further information can be found at:

http://www.redversconsulting.com/cobol_xml_interface.php

Sample XML Document

The sample XML document below, has been generated by the Redvers COBOL XML Interface. It includes examples of a document declaration (A), a comment (B), SOAP (Simple Object Access Protocol) envelope (C), namespace declarations (D), attribute (E), COBOL data formatting (F), a mixed content element (G) and an entity reference value (H).

The two `<CARRIAGE>` elements were generated by passing a single carriage data structure to the interface twice. If there had

been 100 carriages, this would require 100 passes, without any additional memory overhead.

This sample also uses XML element names that are reserved in the COBOL language ("Date" and "Length"), as well as names containing characters not permitted in COBOL words (colon and underscore). It also uses the same element names in different data structures ("Name", "Length" and "Weight"). None of this would be possible using COBOL language extensions.

```

<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<!-- This XML document was generated by RCCOBXML -->
- <TRAINDOC>
- <SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap"
  xmlns:uk="http://www.greenwichmeantime.co.uk"
  xmlns:us="http://www.easternstandardtime.com">
- <SOAP-ENV:Body>
- <TRAIN Date="18/03/2003" Time="13:00">
- <LOCOMOTIVE>
  <Name>Thomas</Name>
  <Length>12,500.00</Length>
  <Weight>3,400</Weight>
</LOCOMOTIVE>
- <CARRIAGE>
  <Name>Annie</Name>
  <Length>10,000.00</Length>
  <Weight>2,000</Weight>
  - <Other_Information>
    Room for
    <count>100</count>
    standing & seated
  </Other_Information>
</CARRIAGE>
- <CARRIAGE>
  <Name>Clarabel</Name>
  <Length>12,500.00</Length>
  <Weight />
</CARRIAGE>
</TRAIN>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
</TRAINDOC>

```

The diagram illustrates the XML document structure with callouts A through H pointing to specific elements:

- A**: Document declaration (`<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>`)
- B**: Comment (`<!-- This XML document was generated by RCCOBXML -->`)
- C**: SOAP envelope (`<SOAP-ENV:Envelope`)
- D**: Namespace declarations (`xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap"`, `xmlns:uk="http://www.greenwichmeantime.co.uk"`, `xmlns:us="http://www.easternstandardtime.com"`)
- E**: Attribute (`Date="18/03/2003"` and `Time="13:00"`)
- F**: COBOL data formatting (commas and periods in numbers like `12,500.00`)
- G**: Mixed content element (`<Other_Information>` containing text and `<count>`)
- H**: Entity reference value (`<Name>Clarabel</Name>`)

Conclusion

Clearly, the Redvers COBOL XML Interface represents a far lower total cost of ownership than a team of programmers writing bespoke programs or running database loads and unloads every day.

The missing functionality in COBOL language extensions makes XML extensions to COBOL a non-starter for most XML projects.

Server based interfaces carry the costs of additional systems skills and creating multiple definitions of the same data. In addition, are the costs and risks in moving corporate information outside the host machine.

Finally, the specific design of the Redvers COBOL XML Interface not only makes it functionally superior to its rivals but the pure COBOL approach, results in better integration with existing applications and a far smaller system footprint.

To order a free 30 day trial of the Redvers COBOL XML Interface, please go to:
http://www.redversconsulting.com/cobol_xml_free_trial.php

If you have any questions or wish to request an individual quotation, please contact us at:
<http://www.redversconsulting.com/contact.php>

Users of the Redvers COBOL XML Interface

Affiliated Computer Services (USA)

Bank One / JP Morgan (USA)

Canada Life Assurance (UK)

CUNA Mutual Life Insurance (USA)

Edulinx (Canada)

FirstBank (USA)

Fegro Selgros (DE)

John Deere (USA)

Oppenheimer (USA)

Sasktel (Canada)

Standard Life Assurance (UK)

Suncorp (AUS)

WorldCom / MCI (USA)

Zurich Insurance (UK & SUI)

The Product Package

A perpetual license for a generator-parser pair of modules costs **18,000 US dollars** or equivalent currency. Alternatively, the interface can be leased on an annual basis for just **3,600 US dollars** per year.

Your purchase includes:

- Program source code (*cloaked*)
- Sample COBOL calling programs
- User Guides
- Corporate level software license
- Two year warranty
- Product upgrades and support via email*

Additional options:

- 24 x 7 telephone hotline support
- Software escrow agreement with the NCC Group

Software and documents are shipped in the form of email attachments unless otherwise requested. Installation is performed by copying the source code text into your COBOL source code library and running your standard site compiler.

We also provide a consultancy service for the building of client application programs that call the Redvers COBOL XML Interface. Charges for this service are based on our standard hourly consultancy rate.

Further information can be found at:
http://www.redversconsulting.com/cobol_xml_interface

* Free for the first two years followed by an annual fee of 2,400 US dollars.

About Redvers Consulting

Redvers Consulting have been providing top quality products and services for COBOL applications since 1988. Our clients are primarily large financial institutions in Europe and North America, although we also have customers in many other business and geographical areas.

Our ability to deliver software in COBOL source code form, gives customers reliable, efficient and perfectly integrated solutions to business needs. Source code distribution also means our software will run on all hardware platforms and operating systems: *EBCDIC, ASCII, big endian or little endian*.

We are business partners with **IBM, HP, WRQ** and **Fujitsu Siemens**, and our development team are members of the **Professional Contractors Group**. In 2009 we won the Thames Gateway **Best use of Technology Award**.

Our client list includes:

Affiliated Computer Services (USA)
Bank One / JP Morgan (USA)
Barclays Life Assurance (UK)
Canada Life Assurance (UK)
Citibank NA (USA & UK)
CUNA Mutual Life Insurance (USA)
Deutsche Bank (USA)
Deutsche Rentenversicherung Bund (DE)
Dun & Bradstreet (UK)
FirstBank (USA)
Fegro Selgros (DE)
GMAC Insurance (USA)
John Deere (USA)
Lehman Brothers (USA & UK)
Oppenheimer (USA)
Railtrack / Network Rail (UK)
Sasktel (Canada)
Standard Life Assurance (UK)
Suncorp (AUS)
WorldCom / MCI (USA)
Zurich Insurance (UK & SUI)

Contact Information

www.redversconsulting.com/contact.php

Development Office:

Redvers Consulting Limited
Channelsea House,
Canning Road,
London E15 3ND, UK

Tel: +44 (0)208 503 1211

Fax: +44 (0)700 603 8655

Head Office:

Redvers Consulting Limited
1st Floor, 48 Dangan Road,
London E11 2RF,
UK

Tel: +44 (0)870 922 0633

Fax: +44 (0)700 603 8655

German Office:

Redvers Consulting Limited
Postfach 30 03 26,
50773 Köln,
Deutschland

Tel: +49 (0)221 1704 9000

Fax: +49 (0)221 271 1016