Beyond Y2K
COBOL in the New Millenium

A white paper for Fujitsu Software Corporation

September 1999

Abstract

COBOL has been a major player in business application development for over forty years with good reason. It has been a constant factor through several major milestones in the computing world. The new millennium is starting with the biggest shift in global business ever seen - the move to e-commerce and Internet-based applications. COBOL is ideally positioned to play a significant role in enabling corporations and individuals to participate in, and benefit from, a radical change in the way we do business.

This paper comments on why COBOL has lasted so long and presents how it is ideally positioned to be a major contributor to the Internet revolution. It reveals how both corporations and individual COBOL programmers can cash in on the immense value of their existing COBOL applications and expertise.

Introduction

By the year 2000, the COBOL programming language will have been in use for over forty years. This versatile language has survived years of usage that have seen computers go from multi-room sized behemoths to small desktops, and even handheld units. As the next millennium approaches, it is estimated that there are as many lines of COBOL[1] code in production usage than all other programming languages combined.[2,3]

Over the years, COBOL has been criticized for supposedly being an outdated language. However, it remains widely in use today. Those who truly understand COBOL know that it has evolved over the years both spearheaded by the COBOL vendors and solidified by its standards committees. While critics feel that COBOL is a dying dinosaur, others recognize that it is instead more like an evolving shark - always on the move! In an article written for *COBOL World Online*, Steve Curd, Senior VP and CIO of Strategic Business Services for United HealthCare said, "COBOL applications will continue to be found powering the hearts of most large companies - a fact that is likely to remain a constant far into the future".[4]

Significant Computing Events

Looking back at the evolution of the computer industry, there have been a few significant events that have vastly changed the computing landscape. For example, the development of mini-computers, and later microcomputers and PC's, brought computing power to departments and then to individuals. Telecommunications and networking advancements over the years have connected global corporations and made instantaneous electronic transactions an every day, or every microsecond, occurrence. Advancements in operating systems to include graphical user interfaces (GUI's) and to encompass multimedia technologies such as audio and video have brought computers within the grasp of even the least technically minded.

We are now facing one of the most significant evolutionary steps in computing history, and many say of business history, the coming-of-age of the Internet. The Internet is changing the way we think about using computer technology. It brings the power of a worldwide global network into the computing arena, just as early telephone networks connected people at the beginning of the current millennium. The growth of the Internet - traffic doubling every 100 days[5] - as well as the potential business opportunity - $1.3 Trillion by 2003[6] - is mind boggling.

COBOL has continued to follow these significant evolutionary steps in the computer industry - providing workhorse, record-crunching power with straightforward maintainability wherever it has been required. Its greatest contributions have been in environments where extensions to the language or complementary products have seamlessly integrated COBOL into that environment.

For example, the CICS and IMS systems on the mainframe took COBOL from batch to online transactions. The addition of the Screen Section and the EXEC SQL interface to databases made COBOL an essential tool for mini-computers and early PC's. It has only lost ground in the PC arena when GUI strategies were unclear - but even that is changing.

Now COBOL is finding a new home on the Internet.

COBOL's Vital Role in the E-Commerce Revolution

The development and growth of the Internet has sparked an electronic commerce (e-commerce) revolution. It is now possible to offer goods and services for sale directly into businesses and homes from almost anywhere on the planet. These new electronic storefronts cost a fraction of what traditional storefronts cost to set up and can reach out to customers anywhere via the Internet, as opposed to customers located in the same area as a traditional storefront. Additionally, they are dynamic and may be modified or even completely redesigned quickly and with much less cost and effort.

However, these electronic storefronts still require traditional business systems support, such as order entry, inventory management, credit card transactions, accounts receivable, shipping, and all the other systems required in a modern distribution operation. These application systems need to:

- interface to database management systems
- produce reports
- control sophisticated user interfaces
- handle multiple transactions at the same time
- provide online reporting
- preserve data integrity

Sound familiar? COBOL programmers have been developing these types of systems for many years. These systems have traditionally been developed on mainframes or on mini-computer and PC networks. While the advent of the Internet brings to bear a new application delivery paradigm, traditional development methodologies fit very nicely with Internet applications. Many COBOL application systems are easily ported to Internet server platforms with minimal changes to tap into this exciting marketplace.

Amazon.com, one of the biggest e-commerce "successes" in terms of sales generated, is having to invest enormously in creating an infrastructure to support its success. Do you want your corporation to have to borrow $1.5 billion[7] to get into this market? Certainly from an information systems' perspective you may find, like many other companies, that your systems can be readily adapted to support new Internet storefronts.

At their website Gartner Group advises executives, “To approach e-business holistically. A key challenge will be to integrate technology into the core of the business, rather than use technology to create ancillary or parallel businesses. Enterprises that fail to do this will waste valuable time and may not survive."[8] When you use your existing COBOL systems, that integration is going to be a given. COBOL will play a vital role in the e-commerce revolution by providing the support systems that are required by all businesses, and by providing it now, not later.

How COBOL Applications can Generate Huge Profits on the Internet

How is the Internet going to generate new profits for your business?

- By offering cheaper products?

  Current models (for example, Amazon.com[9] and eToys[10], where new e-commerce businesses use price as one of their selling points have yet to see a profit) suggest that this may not be the solution.

- By offering better products?

  This may be an element, but making better products is not confined to the Internet.

Where your business can make a difference is in offering either better online service or by saving time and money by offering more services online.

The Internet allows the business model to reach directly into businesses and homes, thus bringing your services to millions of customers and clients. Using a combination of HTML forms (web pages) and supporting server-side COBOL applications, you can take answers to your customers where previously they had to come to you.

COBOL has been a winner in business programming whenever it is well integrated with the environment in which it is to run. One area that demonstrates the necessary integration is in Internet communications. Common Internet communication protocols are CGI (Common Gateway Interface) and ISAPI (Internet Server API). They essentially provide ways of sending HTML pages to and receiving data from users - data-in, data-out functions that are familiar to COBOL programmers.

Fujitsu COBOL, for example, makes it easy to develop and deploy these types of applications using a higher level API[11] - a generalized interface that supports CGI, Oracle's Application Server and, in Fujitsu COBOL Version 5, Microsoft® ISAPI[12] (*See Figure 1*). An interface like this allows you to use much of your existing COBOL skills to develop Internet applications with a minimal time investment required to learn additional technologies such as CGI.

Microsoft's Active Server Pages (ASP)[13] are another way of building business logic into web applications. An enhanced version of HTML (the Active Server Page) contains Visual Basic® scripts and / or Java scripts that can invoke COM objects to perform whatever functions are required. Just like a regular sub-program these objects receive parameters, execute the requested functions, and return data. The COM objects can contain or invoke COBOL programs, which could be existing sub-routines - another way of integrating COBOL into the Internet environment.

The best part about this is that if you already have "information-for-customers" applications in hand, you may re-deploy them on an Internet server with little modification (with the exception of changing the user interface to use HTML forms). Whether your data is stored in SQL databases, indexed files, or BTRIEVE™, you can access it from a Fujitsu COBOL program.[14] Whether you come from IBM, Wang, Unisys, or a myriad of UNIX platforms, check for dialect conversion tools that ensure your COBOL syntax is compatible with the new environment.[15] These tools ensure that you have a smooth transition.

Like the companies in Case Studies D and E (*See pages 11 and 12*), you can increase your business's impact and profitability by tapping the power of the Internet with both new and existing COBOL applications. When you survey the options you are likely to concur with Martin Butler, chairman of the Butler Group, "As far as I am concerned, there is no good reason not to use one of the contemporary versions of COBOL for developing the server end of e-business applications, and in reality many organizations will."[16]

The code below, with explanatory comments inserted, demonstrates how a well-designed COBOL API can bring portable web programming within the reach of all COBOL programmers.

```
* Get the setting "Check 1" of a checkbox
*   on the input HTML form
     MOVE "Check 1" TO COBW3-SEARCH-DATA
     MOVE 0 TO COBW3-SEARCH-LENGTH
     CALL "COBW3_VALUE" USING COBW3
* Is it set?
    IF COBW3-SEARCH-FLAG-EXIST
      PERFORM CHECK-1-PROCESSING
    ELSE
      PERFORM NOT-CHECK-1-PROCESSING
    END-IF
    :
* Setup a prototype HTML variable name and
*   value
*   (Prototype HTML is a Fujitsu API feature
*   that lets you place variable strings in
*   HTML files. The code below inserts the
*   string "<B>John F. Kennedy</B>" in the
*   output HTML.)
    MOVE "GET-NAME" TO COBW3-CNV-NAME.
    MOVE 0 TO COBW3-CNV-NAME-LENGTH.
    MOVE "<B>John F. Kennedy</B>" TO COBW3-
     CNV-VALUE.
    MOVE 0 TO COBW3-CNV-VALUE-LENGTH.
* Register the value for GET-NAME.
    CALL "COBW3_CNV_SET" USING COBW3.
     :
     :
* Output the HTML document
    MOVE "a.htm" TO COBW3-HTML-FILENAME.
    CALL "COBW3_PUT_HTML" USING COBW3.
```

Figure 1. Simplified Internet Communications

The above code executes unchanged on CGI, Oracle Application Server and, in Fujitsu COBOL Version 5, ISAPI. The bold lines show three of the eleven simple function calls in the API.

Note: Moving zero to the length fields tells the system to work out the data length itself.

Other Ways of Leveraging your Existing COBOL Applications

Another way to take advantage of your existing applications is to make use of a corporate Intranet. An *Intranet* is a closed or secured local area network (LAN) whose clients are enabled for the TCP/IP network protocol and have Internet browsers such as Microsoft's Internet Explorer™ installed on them. COBOL applications are deployed on web servers on the Intranet. The difference between the Internet and an Intranet is that the Intranet-based web servers are not accessible to the outside world. Otherwise, you use the same technologies described in the previous section to bring your staff increased access to your corporate data.

You can also use Fujitsu COBOL and PowerCOBOL™ to port existing legacy COBOL applications to the PC environment. These applications may actually perform much better in this environment, and offload your mainframe or mini-computers, freeing them up for other tasks, or allowing them to be replaced completely by PC's and networked PC servers. Fujitsu has had customers report systems running in a fraction of the time of their mainframe counterparts[17,18]. This can be an extremely cost-effective solution and may allow you to balance application usage in a much more efficient manner. COBOL is a highly portable language and moving applications from the mainframe to PC's can be quite a straightforward process. Fujitsu's standard and custom COBOL converters can additionally aid more complex conversions.

Finally, if you have COBOL applications developed on the PC already, having used another vendor's COBOL compiler, converting them to Fujitsu COBOL can increase the performance significantly. Customers report their PC COBOL applications executing in half to a third the time of competitive PC COBOL products[19,20].

COBOL in the Microsoft World

While the COBOL language has shown a remarkable resiliency throughout its history, the Microsoft® Windows® world has probably offered the largest challenge to COBOL compiler vendors to support. This environment has progressed very rapidly and is at the heart of most of the new innovations in the computer industry, encompassing GUI and multi-media technologies at a staggering pace. To survive in the Microsoft world today software products must offer:

- GUI interfaces
- ODBC access
- COM and DCOM support
- Creation or use of ActiveX™ objects
- OLE connectivity
- (*Ideally*) Direct access to Windows system functions

Not all COBOL vendors have been able to offer these features quickly or in a convenient manner. In the 70's and 80's, when it was unclear which platform would dominate - UNIX, Windows, OS/2®, or Apple® - many vendors focused on cross-platform portability. Consequently, some of these COBOL compilers do not produce standard Windows object code that is linkable using the standard Microsoft Linkage Editor. As a result, both performance and integration with outside technologies is sacrificed. This has lead some people to believe that COBOL cannot work well in the Microsoft world.

Fujitsu COBOL, on the other hand, was written from the ground up to create high performance native Windows object code. The native Microsoft Linkage Editor is used with Fujitsu COBOL. This provides the additional benefit of allowing smooth integration with outside technologies. Microsoft native Windows API's may be called as easily from Fujitsu COBOL programs as they are from "C" and "C++" programs, for example.

The solid technical base of Fujitsu COBOL has ensured a good fit with the Microsoft world:

- **GUI's** can be created using PowerCOBOL as the design/execution tool (where you code entirely in COBOL) or by integrating with Visual Basic.
- **ODBC** access is built into the product - accessed through EXEC SQL syntax or PowerCOBOL's DBAccess controls.
- PowerCOBOL is **COM and DCOM** compliant and allows you to use and create **ActiveX** controls.
- Fujitsu COBOL supports invoking Microsoft C++ written objects.

- **OLE** servers, written in other languages, can be instantiated directly from COBOL. For example, Microsoft Excel™ can be invoked with no additional "glue" (*See Figure 2*).

- Anyone with a basic understanding of Windows data types[21] can write the code to call the **Windows system functions** using standard COBOL CALLs.

```
 REPOSITORY.
 CLASS OLE AS "*OLE".
 :
 01 EXCEL OBJECT REFERENCE OLE.
 01 WORKBOOK OBJECT REFERENCE OLE.
 01 SHEETS OBJECT REFERENCE OLE.
 01 WORKSHEET OBJECT REFERENCE OLE.
 01 CELL OBJECT REFERENCE OLE.
 01 APPLICATION PIC X(20) VALUE "EXCEL.APPLICATION".
 01 FILENAME PIC X(20) VALUE "C:\TEST.XLS".
 01 SHEETNAME PIC X(20) VALUE "SHEET1".
 01 OLE-TRUE PIC 1(1) BIT VALUE B"1".
 01 FILLER PIC 1(7) BIT.
 01 ARRAY-ROW PIC S9(9) COMP-5.
 01 ARRAY-COL PIC S9(9) COMP-5.
 01 VAL PIC X(256).
 :
* Start EXCEL
    INVOKE         OLE "CREATE-OBJECT"
         USING     APPLICATION
         RETURNING EXCEL.
* Retrieve WORKBOOK.
    INVOKE         EXCEL "GET-WORKBOOKS"
         RETURNING WORKBOOK.
* Open the file.
    INVOKE         WORKBOOK "OPEN"
         USING     FILENAME
         RETURNING WORKBOOK.
* Retrieve the sheet collection.
    INVOKE         WORKBOOK "GET-WORKSHEETS"
         RETURNING SHEETS.
* Retrieve a worksheet.
    INVOKE         SHEETS "GET-ITEM"
         USING     SHEETNAME
         RETURNING WORKSHEET.
* Retrieve the cell(1,1)object.
    MOVE 1 TO ARRAY-ROW ARRAY-COL.
    INVOKE         WORKSHEET "GET-CELLS"
         USING     ARRAY-ROW ARRAY-COL
         RETURNING CELL.
* Retrieve the cell value.
    INVOKE         CELL "GET-VALUE"
         RETURNING VAL.
    DISPLAY "VALUE = ", VAL.
* Change the cell value.
    MOVE "ABCDEFG" TO VAL .
    INVOKE         CELL "SET-VALUE"
         USING     VAL.
* Close the file and save the change by setting the argument to true.
    INVOKE WORKBOOK "CLOSE"
         USING     OLE-TRUE.
* Quit EXCEL.
    INVOKE EXCEL "QUIT".
```

The functions ("methods") being invoked: "GET-WORKBOOKS", "OPEN", "GET-WORKSHEETS", etc. are defined by the OLE application server (Excel). Notice that they can be accessed directly from COBOL.

Figure 2. Example code illustrating using Microsoft Excel as an OLE server.

For COBOL programmers this means that the Microsoft world is immediately available to you. For IS managers, the considerable investment you have in COBOL code and programmers can be brought to the Microsoft world today.

Which Tools Work Best with COBOL?

Provided that your COBOL system is OLE and COM enabled like the Fujitsu COBOL development environment, you may choose from a wide array of tools to assist you in building applications. Using PowerCOBOL, for example, you can obtain literally thousands of third party controls to use in your COBOL applications. Microsoft, among others, publishes catalogs of such controls. Any application that can be used as an OLE server also can become a valuable add-on to your COBOL application.

Microsoft's Visual Basic development environment is excellent for creating your GUI (*See Case Studies C and E on pages 10 and 12*) and you can easily call COBOL DLL's or you can drop COBOL controls into Visual Basic applications. This allows you, for example, to compile sophisticated relational database access routines written in COBOL using embedded SQL into DLL's and to call these from a newly developed Visual Basic front end.

Additionally, vendors like Fujitsu provide supporting tools such as a data file editor, a data conversion utility (which can also convert EBCDIC data to ASCII), and a source code control and configuration management facility (PowerGEM). Fujitsu's world class PowerBSORT product is available for handling industrial-strength sorting needs, and is also available as an ActiveX control that may be used in other development environments, such as Visual Basic.

Another key tool is Microsoft's ODBC (Open DataBase Connectivity). ODBC allows developers to code to a standard API that is transparent from the actual database management system being used. This has two advantages. The first is that you do not have to be familiar with a particular database's proprietary API. You can develop database applications that are largely uncoupled to a specific vendor's database system.

The second advantage is scalability. You can write an application that uses a standalone PC database system later switching it to use an enterprise-level database system such as Microsoft's SQL Server or Oracle.

Fujitsu COBOL adds another advantage to ODBC: you don't even have to learn the somewhat complex ODBC API. Instead, you code standard embedded SQL, and the Fujitsu COBOL compiler automatically and transparently converts it to the native ODBC API. This means that you can port existing SQL COBOL applications to ODBC and thus to any number of database systems quickly and often without any modification.

Using these types of tools with COBOL development vastly cuts down on the effort required to field modern, world-class applications.

Reusing Code and Components and Object-oriented Technology

If you are going to continue or even start writing applications in COBOL, how much are you going to be able to reuse? After all, industry experts are still buzzing about object reuse and component-based development.

First, we suggest you get through the hype. Even IDC, a proponent of component-based development says, "Despite the existence of a standard, developing applications from objects and components still represents a significant change in culture for many organizations. Component-based development still has a long way to go before it is considered mainstream."[22] The bottom line is: Don't rush away from COBOL because another product claims to be offering a component-based system!

Second, you'll find that your methodology is far more important to your ability to reuse your code than the implementation system. For example, it has been found that information hiding and encapsulation are the real keys to success in code reuse.[23]

As described previously, Fujitsu's ability to use OLE and to create ActiveX components provides a very high level of reusability for COBOL application

development. A COBOL ActiveX component may be used across a wide number of other applications. This allows COBOL developers to segment applications logically into reusable application components and to share them across applications via exposed properties and methods, while encapsulating the actual COBOL logic and data from the outside world. Fujitsu COBOL is fully object-oriented COBOL[24] compliant as well. This means that you can produce object-oriented COBOL applications that define and use classes, properties and methods.

Fujitsu's PowerCOBOL produces ActiveX components and allows developers to create custom methods and properties to be called in an object-oriented fashion from object-oriented languages such as Visual Basic and Visual C++. PowerCOBOL, however, does not force developers to learn Object Orientation in order to produce these types of applications. This can be a real boon for organizations that desperately need to interface their COBOL applications to other object-oriented applications.

Summary

As we enter the next millennium, it is comforting to realize that COBOL still encompasses new technologies while carrying along its longstanding dependability and robustness. The Internet, Object Orientation, Windows GUI, ODBC, and ActiveX are just a few of the technologies that are with us as we enter the next century. COBOL is ready today for these exciting technologies and will continue to evolve to encompass newer and more exciting technologies as they emerge.

Leveraging your existing COBOL applications and skills is critical to your on-going success, and choosing the proper platform to successfully empower you is critical to your future.

Case Studies

The following five case studies demonstrate how many businesses are successfully using Fujitsu COBOL. For brevity some of the case studies are composites of the experiences of more than one customer.

Footnotes

[1] 125 Billion lines (University of Michigan estimate), 180 Billion lines (Gartner Group, 1997), 500 Billion lines (Carol Baroudi, "Mastering COBOL", Sybex 1998)

[2] Gartner Group - Quoted in University of Michigan Overview of Y2K

[3] COBOL and the Business Programming Paradigm, by Jonathan Sayles, Micro Focus, Ltd.

[4] A View from the Top, COBOL World Online, see http://www.objectz.com/cobolworld/vft/view_curd.htm

[5] The Emerging Digital Economy Published by the United States Department of Commerce, April 15, 1998

[6] Forrester Research, press release, December 17, 1998

[7] Amazon.com, Quarterly SEC Report, June 30, 1999

[8] Gartner5.gartnerweb.com, "Is e the only letter?", August, 1999

[9] Amazon.com, Quarterly SEC Report, June 30, 1999

[10] ETOYS Inc, Quarterly SEC Report, August 13, 1999

[11] Interface is fully documented in the "Fujitsu COBOL CGI 2.0 User's Guide", February 1999

[12] COBOL Developer News, Volume 3, 1999, Fujitsu Software Corporation

[13] See http://msdn.microsoft.com/workshop/server/asp/aspfeat.asp for more information

[14] See http://www.adtools.com/products/windows/cobol.htm for more information

[15] See http://www.adtools.com/products/windows/diaconv.htm for more information

[16] Martin Butler, Contra, May 1999

[17] COBOL Developer News, Volume 2, 1999, Fujitsu Software Corporation

[18] COBOL Developer News, Volume 4, 1999, Fujitsu Software Corporation

[19] COBOL Developer News, Volume 4, 1999, Fujitsu Software Corporation

[20] Case Study A, in this paper (page 8)

[21] For an example, see Internal Data Representations in Carol Baroudi's "Mastering COBOL", Sybex 1998, (good coverage of this topic)

[22] IDC#19187 Programmer Development Tools Synopsis: 1999 Worldwide Markets and Trends (June 1999)

[23] Steve McConnell, "Rapid Development", Microsoft Press 1996

[24] See "Object Oriented COBOL Programming in COBOL 2000" by Artur Reimann, April 1999, for more information. http://www.adtools.com/info/whitepaper/oocoby2k.pdf

Customer A - Regains Competitive Edge by Moving from Out-of-date COBOL Vendor to Mainstream Fujitsu COBOL

Starting Point

- Developing PC applications for the medical services industry for several years.
- Had built up suites of applications consisting of **millions of lines of code**, tailored to many different clients.
- Using a **PC COBOL product that had fallen behind today's technology** yet still demanded fairly high maintenance and run-time distribution fees.
- **Future developments** were increasingly **restricted** by the PC COBOL product's built-in limitations.
- Business is based on providing quick updates to customers - alternative solutions must not interrupt this flow.
- Used **many proprietary extensions** to the COBOL language - extensions not supported by other COBOL vendors.

Goals

- **Move to a mainstream COBOL vendor**.
- **Break out of limitations**, such as program size restrictions.
- Save on maintenance and run-time fees.
- Gain performance and functionality improvements in applications delivered to customers.
- **Have a smooth transition** - ideally developing for both old and new COBOLs using the same source files, until all applications are moved to the new vendor.
- **Absolutely no interruptions** to the continuous rollout of **customer application updates**.

Solution

Fujitsu COBOL with Dialect Conversion Services
By contracting with Fujitsu Software Corporations' COBOL Dialect Conversion Team, the major headaches were taken on by Fujitsu and a smooth transition to a major COBOL vendor (Fujitsu) was accomplished.

Key Product Features

Fujitsu COBOL

- **Reliable platform** providing a bug-free transition (actually we did discover a bug in an area that relied on the values and handling of un-itialized fields, which we felt was unsafe territory, but our conversion team still provided a fix).
- **Super-fast code** providing application execution in half the previous times - with room for further improvements when code is tuned.
- Future development paths include creating GUI's with PowerCOBOL.

Dialect Conversion

- **Reliable conversions** built on proven transformation technology.
- **Tailored conversion** developed in association with customer A, ensuring **solutions that worked**.
- Creative solutions devised and delivered for coping with highly-specialized extensions to the COBOL language.
- **Dialect processing options** that allow developers to code in the old COBOL dialect yet execute under Fujitsu COBOL. (A big benefit as application enhancements can continue throughout the conversion project - without any training requirements for the bulk of the programming staff.)
- **Dedicated support** to see the project to a successful conclusion.

Results

- **Saved many man-years of effort** in moving from the old vendor.
- **Applications are executing twice as fast**.
- Future **savings in maintenance** and **run-time fees**.
- **Successful transition** to a mainstream COBOL vendor.
- **No interruptions** to the continuous rollout of customer application updates.

Customer B - Takes Character-Based Reports and Screens to Graphical Formats and Improves Customer Satisfaction

Starting Point

- **Building character applications in Micro Focus COBOL**.
- Most of the applications generate reports; **each report must be designed and coded by hand**.
- The **data** is stored **in indexed files**.
- Providing customers with access to the data has been a problem.
- Increasing **customer demands for GUI's**.

Goals

- **Find tool for designing and producing reports** using varied fonts, colors and graphics, without excessive coding requirements.
- Provide **customers** with the ability to **create** their **own ad-hoc reports** from the application data.
- Transition **character applications to GUI**.
- Achieve these results within six months using existing staff and budgets.

Solution

Fujitsu COBOL Enterprise Edition containing: Fujitsu PowerFORM, Crystal Reports, Fujitsu PowerCOBOL, *plus* Dialect Converter for Micro Focus COBOL, *plus* Crystal Reports To Fujitsu COBOL

Fujitsu COBOL offered the required tools and a straightforward transition - quicker to complete and a better fit than the alternative of re-writing in VB.

Key Product Features

Fujitsu PowerFORM

- Graphical report design tool provides fast WYSIWYG report creation.
- Reports written using COBOL file-handling syntax offer a quick learning curve for programmers and straightforward rewriting of the existing report-creation code.
- Control of feeder trays and duplex printing - a much-desired bonus.

Crystal Reports/Crystal Reports to Fujitsu COBOL

- Inclusion of Crystal Reports in Fujitsu COBOL saves researching other vendors for the ad-hoc reporting piece.
- Crystal Reports to Fujitsu COBOL gives access to Fujitsu's data files from Crystal Reports - satisfying the customer's requirements with **no data file conversions**.

Fujitsu PowerCOBOL

- **Attractive GUI solution** requiring **minimal learning** for COBOL developers.
- Complete design / coding / debug / production package.
- Integration with standard COBOL works well for Customer B whose application was structured so that most functions were modularized - these are quickly "snapped-in" to PowerCOBOL.

Dialect Converter for Micro Focus COBOL

- Reliable conversion from the Micro Focus dialect.
- **No need for conversion services** as the converter covers all Micro Focus COBOL extensions used by Customer B.

Fujitsu COBOL

- **Reliable, high-performance** platform.
- **Cost-effective offering**, enabling the project to be executed within current software budgets.
- Supports existing COBOL report formats, so the move to PowerFORM's graphical-style reports can be staged.

Results

- Key customers **quickly provided** with usable and **professional-looking reports**.
- **Project completed within required time-scales and budgets**.
- **Plan** in place for **transition to GUI**.
- **Increased developer satisfaction** with resilient production code and responsive support service.

Customer C - Gains Great Cost Savings and Fast Deployment to the Windows Environment by Migrating a Mainframe COBOL System to Mixed Visual Basic and COBOL

Starting Point

- Mainframe **CICS COBOL** application providing reporting for the reinsurance industry.
- **Demands from customers for graphical** PC-style **interfaces** and **PC response times**.
- Competitive requirement to reduce product costs.

Goals

- **Move application to PC's**.
- Provide **up-to-date graphical interface**.
- **Build product on ODBC** databases, giving flexibility in the actual database used.
- Minimize transition costs by **maximizing code reuse**.
- **Reduce product costs**.

Solution

Visual Basic Front-End with Fujitsu COBOL
Visual Basic was chosen at an early stage as the appropriate tool for creating the user interface. However, the mainframe batch COBOL programs had to be supported in a tightly integrated way. Fujitsu COBOL offered the best solution.

Key Product Features

Visual Basic

- Popular interface creation package.
- Kept at the leading edge of interface design by Microsoft.
- Ready access to Visual Basic resources.

Fujitsu COBOL

- **Seamless integration** with Visual Basic.
- **Excellent compatibility** with mainframe batch COBOL programs.
- **Built-in ODBC access**.
- **High performance**.
- **Free run-times**.

Results

- **Successful transition to the PC**.
- **Increased customer satisfaction** with product interfaces.
- **Reduced product development costs**.
- **Reduced product costs** for customers.

Customer D - Redefines Business Focus by Moving a Mini-Computer COBOL Application to a Windows-based Internet Solution

Starting Point

- Medical laboratory **reporting system developed** on Wang platform **over several years**.
- Anyone wanting a copy of a laboratory report **fills out a request form, submits it**, and is **then sent a paper copy of the report** some days later.
- Application written and owned by a consulting company, always on the lookout for opportunities to increase its salability.

Goals

- **Bring application to the Internet**.
- **Cut out the form filling** - let clients access the (password-protected) medical reports themselves.
- Give **instant access** to the reports.
- Add **online viewing** of laboratory graphs.

Solution

Use Fujitsu COBOL CGI

The code used for the Wang COBOL application was mostly ANSI standard apart from the screen handling area - which was being removed anyway. Thus moving the code to Fujitsu COBOL was a fast, smooth operation. Creating HTML versions of the existing screens and providing HTML output using the Fujitsu COBOL CGI feature was straightforward.

Key Product Features

Fujitsu COBOL

- **Reliable, ANSI standard** COBOL system making it easy to transfer code.
- **Powerful debug tool** that made it easy to track down any server-side problems.
- **No run-time charges** - a big winner over other possible COBOL vendors.

Fujitsu COBOL CGI

- Simple interface for creating CGI "scripts" (the common term for CGI programs).
- Prototype HTML feature allows HTML chunks to be designed in HTML editors with the variable parts substituted at run-time. (Typical CGI programming requires the program to output all the HTML line-by-line.)
- Provides **future portability** to other communications protocols, such as ISAPI, with no coding changes.

Results

- Customer D now offers medical laboratories and their clients a radically new way of doing business.
- Much faster turn-around of laboratory reports.
- Accomplished in a fraction of the time (estimated between 1/10 and 1/20) it would have taken to write the application from scratch.

Customer E - Maintains Investment in Code and Staff by Incorporating Batch COBOL Code into a Microsoft Internet with ASP Solution First Prototyped using Visual Basic

Starting Point

- **Thousands of mainframe batch COBOL programs** that access an **SQL database**.
- **25 COBOL programmers** who know the system inside out.
- Customers **request information by phone** or form. **Batch jobs** are then **run** and the **results sent** (on paper) to the customer.

Goal

- Provide a **web solution** that lets the **customer request data** and have the **data returned via the web**.
- Produce the solution quickly, in the **most cost-effective manner**.
- **Use the existing programming team**.

Solution

Fujitsu COBOL with ASP/COM Interface Routine *plus* **Microsoft IIS, ASP, and SQL Server**
The viability of moving to the Web was prototyped using Visual Basic, but rewriting the whole application in Visual Basic was going to be expensive and require extensive retraining for the programming staff. Initial reviews of making COBOL work with the ASP and COM models did not look too hopeful until the Fujitsu development team came up with a COM interface routine that made it possible to use Fujitsu COBOL in the ASP/COM world.

Key Product Features

Fujitsu COBOL

- **Mainframe-compatible** COBOL support makes code transition easy.
- Common object format, making it **easy to integrate** with other parts of the solution, such as Visual Basic.
- Integration with Microsoft's SQL Server provides **straightforward access to data**.

Responsive Support
Fujitsu listened to Customer E's goals and came up with an appropriate solution.

Results

- Customer E's clients can obtain their reports themselves - with results delivered online.
- Transition to the web was **fast and highly cost-effective**.
- Programming staff kept focused on delivering function to clients - with **minimal retraining** required for working in the web environment.
- **Big cost savings** in moving the system from the mainframe to a high-powered PC environment.

Trademarks

Fujitsu Software Corporation
A subsidiary of Fujitsu Limited
Developer Tools Group
3055 Orchard Drive
San Jose, CA 95134-2005
Phone: (800) 545-6774
(408) 428-0300
Fax: (408) 456-7272
Email: cobol@adtools.com
Web: www.adtools.com