

---

---

---

---

---

---

---

---

# COBOL – continuing to drive value in the 21<sup>st</sup> Century

The landscape for COBOL asset ownership

Reference Code: CYBT0006

Publication Date: November 2008

---



## ABOUT DATAMONITOR

Datamonitor is the world's leading provider of online data, analytic and forecasting platforms for key vertical sectors.

We help our clients, 5,000 of the world's leading companies, profit from better, timelier decisions.

Through our proprietary databases and wealth of expertise, we provide clients with unbiased expert analysis and in-depth forecasts for seven industry sectors: Automotive and Logistics, Consumer Markets, Energy, Financial Services, Healthcare, Retail and Technology.

Datamonitor maintains its headquarters in London and has regional offices in Frankfurt, Hyderabad, New York, San Francisco and Sydney. See [www.datamonitor.com](http://www.datamonitor.com) for further details.

## TABLE OF CONTENTS

<b>ABOUT DATAMONITOR</b>	<b>3</b>
<b>TABLE OF CONTENTS</b>	<b>4</b>
<b>INTRODUCTION</b>	<b>6</b>
<b>EXECUTIVE SUMMARY</b>	<b>6</b>
<b>COBOL CREDENTIALS</b>	<b>7</b>
COBOL programs are still powering many parts of the public and private sectors	7
COBOL has outlasted an array of competitors during its lifetime	7
In its many areas of strength, COBOL is the ‘best in class’ language for business applications	8
<b>COBOL ECOSYSTEM AND COMMUNITIES</b>	<b>9</b>
Demand for COBOL skills remains strong	9
Developer communities are active	10
Vendors are still strongly committed to COBOL	10
<b>COBOL ACROSS TECHNOLOGY PLATFORMS</b>	<b>10</b>
COBOL is available on a wide range of platforms	10
Integration is a major strength of COBOL	11
<b>ALTERNATIVE STRATEGIES TO COBOL ASSET OWNERSHIP</b>	<b>11</b>
Drivers towards alternative strategies	11
Language Conversion Options	11
Quality Assurance Issues	12
Business Case Issues	13

<b>CONCLUSIONS</b>	<b>14</b>
<b>APPENDIX</b>	<b>15</b>
Acronyms	15
Ask the analyst	15

## INTRODUCTION

Private and public sector organizations' investments in software written in COBOL (Common Business Oriented Language) have accumulated to a massive extent globally, over the course of its almost half-century history as the leading platform for business applications.

Against the background of technology change, and other shifting market forces, enterprises advisedly evaluate periodically how their existing software assets measure up against future needs.

This White Paper, sponsored by Micro Focus, is an analysis of real-world factors pertaining generally to the future of COBOL-based software assets,

## EXECUTIVE SUMMARY

COBOL has long been in widespread use for writing business applications, and many organizations continue to rely on critical business software written in the language to deliver commercial, competitive advantage. Developments in COBOL-related software products and capabilities now position it as widely available across enterprise-strength IT platforms, and able to integrate extensively with other software used for developing and running applications. However, although COBOL is still being used to write new programs, other languages are now taught more widely in educational institutions, and COBOL is perceived as lying within the less 'sexy' part of the spectrum of programming languages – consequently there are some concerns that COBOL expertise will dwindle as its developers in older age brackets progressively retire, and too few younger developers fill the breach. To address this, several leading software companies are successfully promoting training in COBOL within higher education, and some suppliers of IT services are maximizing their potential opportunities to address resource deficits by training numbers of their workforces in COBOL. Largely, market forces can be expected to balance any excess in demand, by increasing the supply of resources.

Research into organizations' options for transferring COBOL-based assets to other technologies reveals that many have negative experiences. These include issues with retaining the very considerable value that is inherent within their existing assets – many such undertakings involve high risks with little realizable reward, and eventually entail far greater resource commitment than was envisaged. Fortunately for those daunted by poor prospects, we believe that COBOL will remain the enterprise workhorse it has been for decades to come.

## COBOL CREDENTIALS

### *COBOL programs are still powering many parts of the public and private sectors*

COBOL has been the prevalent language for developing business applications throughout the greater part of five decades. Systems and applications written in COBOL remain in widespread use within the vertical sectors that spend some of the world's largest IT budgets – such as Finance, Government, Manufacturing, and Telecoms – as well as numerous others. Recent statistics quoted to Datamonitor by IBM reveal the massive scale of intellectual property accumulated:

- Around 200 billion lines of COBOL code are in live operation.
- 75% of the world's business data, and 90% of financial transactions, are processed in COBOL.
- There are 1.5 – 2 million developers, globally, working with COBOL code.
- Around 5 billion lines of new COBOL code are added to live systems every year.

According to one assessment, the past investments of organizations owning COBOL-based assets represent around \$2 trillion of stakeholder value. In common with the ownership responsibilities associated with other asset types of such scale, companies and public sector organizations have a duty to ensure that benefits arising from their COBOL code base are protected and maximized, and that any associated risks are understood and managed.

### *COBOL has outlasted an array of competitors during its lifetime*

Although its heritage extends to less than half the lifespan of the internal combustion engine, COBOL can be thought of as IT's equivalent to the ubiquitous power source of automobiles, and other human needs for productive energy. Both technologies enabled the human population to benefit in very many ways from new possibilities, and undertake endeavours of far greater value than are possible without them. The internal combustion engine, and the mass production of motor vehicles, brought individual choice to the mass market – by making car ownership feasible for so many, and enabling people to travel more widely, and also by enabling the easier mobilisation of goods, and therefore the spread of consumer choice. COBOL has enabled countless organisations to benefit from the efficiencies and advancements available from using IT systems. Both technological advances, too, have continued to work well, and to match operational and efficiency demands across a wide range of requirements, and – critically – to accommodate modifications and updates to improve their fundamental technology as user needs advance.

Throughout the decades in which COBOL attained its vast user base, a great number of other programming languages have been developed – but almost all have been superseded as platforms for business applications (consider examples such as MANTIS, Gener/OL, FORTRAN, MUMPS, Forte, Pascal, and Smalltalk – all highly popular in their time). In some cases such languages have been platform-specific, and disappeared due to the rapid cycle of hardware technology development. Many others have fallen by the wayside due to rapid changes in the commercial framework that is necessary to provide investment for language development – but a very significant proportion were cast aside when something more efficient and powerful became available. COBOL itself replaced the use of other languages within many organizations, as its availability increased across hardware and operating system platforms.

Although there have been many real advances, the cycle of technology obsolescence is a phenomenon that at times has led to wasteful treatment of business assets in the form of computer programs. A long-held belief in some quarters – with which Datamonitor concurs – sets some responsibility with development staff who become bored with one language, and are keen to enhance their CVs with the latest, up-to-date skills. That responsibility must be shared by the many IT managers who have been surreptitiously pressurized into adopting new technology as a result, with vendors often also playing a role promoting change. Unfortunately, this has happened many times without being properly considered in a business context, a process that would look at the return on the investment that is necessary to make such a transition. The cycle looks set to continue, the most recent manifestation being that even relatively modern languages such as Java, and some variants of C, are labeled as 'legacy' languages by some within developer and vendor communities.

### ***In its many areas of strength, COBOL is the 'best in class' language for business applications***

In a study made by Gartner in late 2007 of how programming languages age ("Assessing the Age of Software Languages and tools", by Jim Duggan), COBOL was classified as being in the 'Adult' lifespan phase, reflecting factors such as the availability of skills, tools, regular releases, and other signs of health, vibrancy, and stability. Visual Basic 6, whilst being a much younger language than COBOL, was classified as 'Elderly', due to its less healthy 'vital signs', and Ruby (one of a crop of languages that have only come to the fore in the post-Millennium period) was classified, like COBOL, as "Adult".

Clearly COBOL is still in a strong position in comparison with other programming languages, as well as having the lengthiest and strongest heritage of being standards-based of any programming language – in short, COBOL well and truly 'looks good for its age'. One of the earliest high-level programming languages, its development began in 1959, when development of COBOL standards was begun quickly by the American National Standards Institute (ANSI), which eventually produced 3 standards, issued successively in 1968, 1974, and 1985. The International Standards Organization (ISO) worked with ANSI committees later, to develop the COBOL 2002 standard.

The language was designed to be oriented towards business problems, and to be machine independent, and capable of continuous change and development. Expressing this in more widely-used terms, COBOL is a less 'geeky' language than most. If COBOL were a commercial product it would surely be one of history's most successful, in that it continues to meet those design criteria and to adapt successfully to a huge extent of change. For example, typical programming constructs were procedural when COBOL was devised, but it has long since fully supported object-orientation. Having adopted many such improvements, the language remains simple and, as it is readable without technical skills, is self-documenting to an extent that aids maintainability, and developer productivity. The range of business problems that COBOL supports is very broad, but its great strengths are in heavy-duty, high-performance data processing, including file storage – and it offers the great advantage that applications are portable to other platforms that support COBOL.

## COBOL ECOSYSTEM AND COMMUNITIES

### *Demand for COBOL skills remains strong*

It is well-known that many types of IT skills are in short supply, and that some organizations have had to tailor and time IT portfolio changes to match skills' availability, because of recruitment difficulties. The market for developers with COBOL skills is no exception to the tendency for demand to exceed supply of IT skills generally, although the productivity benefits of working with COBOL (and especially the modern developer tools that can be used with the language) have the effect of reducing the shortfall somewhat. However, the population of COBOL developers is distinct in that its age profile is older, on average, than that of developers of languages that are less mature. This 'grey-haired' characteristic is due to so many COBOL developers having maintained their skills throughout multi-decade careers. One result is that, each year, a proportion of the longstanding base of COBOL developers is likely to reach the age where they no longer carry on working, and as time goes on the number of developers that have had long careers working with COBOL is likely to dwindle.

To alleviate this situation, a number of initiatives have been instigated. Some vendors with interests in the market for COBOL-related products are working with the academic sector in order to encourage institutions to offer training in COBOL skills. Micro Focus, for example, via its Academic Connection programme, is involved with over 70 universities in 14 countries across the world, from which thousands of trained COBOL programmers graduate annually. IBM is similarly involved in encouraging COBOL training, in partnership with well over 450 academic institutions, and CA supports a number of universities' work in this field. Micro Focus and IBM both enhance the attractiveness of COBOL for developers by providing Integrated Development Environments (IDEs) for COBOL, which take away some of the routine work of programming, automate and speed up the code-test-debug cycle, and make the developer's experience more stimulating, as well as more productive.

Some organizations requiring COBOL skills make use of the resources of services providers (also known as outsourcing companies). Application development and maintenance are particular areas of expertise offered by a number of service providers, and many maintain a large workforce with COBOL skills in order to undertake client projects, or broader portfolio management of COBOL-based assets. A large sector of the services industry has grown in India (as well as a number of other 'offshore' locations), and India-based service providers are known to have tens of thousands of trained COBOL resources. Some companies have taken advantage of the general IT skills base in India to build so-called 'captive' development functions – named because like their onshore equivalents they are part of their owning organization, rather than being 'for hire' – and by setting up such a 'captive' organization those organizations that do not favour the option of using an external provider could leverage a growing base of COBOL skills in India.

Those with a belief in market forces would foresee graduate training, and formal training for service providers' employees, bringing sufficient resources to the market place in order to meet the overall demand for COBOL skills, including any shortfall. If demand fluctuations were volatile then the in-built delay of responding with training and recruitment might be a distinct challenge, but this is not a likely stumbling block given the stability in the COBOL market. Another factor to note is that COBOL programming roles tend to require developers to have business awareness skills as well as those relating to the programming language. There is a mutual benefit as developers strengthen this aspect of their portfolio, as the opportunity to gain strength in business skills tends to reduce the 'churn' amongst developers in the COBOL market.

### ***Developer communities are active***

Many languages that have gained increased popularity during recent times have been successful in doing so in part due to encouraging developer involvement, via online interest groups and other participation facilities that are collectively termed ‘communities’. Developers typically share experiences and insight in such communities; collaborating, and gaining what amounts to support facilities, informally. A number of communities relating to COBOL issues are well-known, including the following examples:

- [www.cobolportal.com](http://www.cobolportal.com) sponsored by Micro Focus.
- Numerous, openly available and membership-based information and news sites, such as drdobbs.com, cobug.com, and cobolplanet.com.
- COBOL user groups within business professionals’ collaboration sites, such as LinkedIn.

We definitely expect COBOL-related communities to continue to increase in number as greater numbers of younger developers join the COBOL-skilled workforce, as younger age groups have a greater affinity with social technologies, and their facilitation of collaborative techniques.

### ***Vendors are still strongly committed to COBOL***

Due to the continued commitment of many end-user enterprises to COBOL, the global market for COBOL-related products is estimated by Datamonitor to be considerably over US\$1 billion annually. The three largest vendors involved commit substantial resources, in many ways, to customers’ and the markets’ well-being. Executives from all three have been interviewed by Datamonitor in the course of preparing this White Paper, and each company believes the future of the COBOL market to be both long and strong. Similarly to those companies that are the largest investors in combustion engine technology, vendors that service the needs of COBOL users have strong interests in providing support for the needs of their customers on an ongoing basis.

## **COBOL ACROSS TECHNOLOGY PLATFORMS**

### ***COBOL is available on a wide range of platforms***

A founding ideal of the COBOL language was that code should be transferable across technology platforms. Whilst many programs need to include code that performs integration with, for example, a proprietary database type, the degree to which this adversely affects portability is limited to easily identifiable code changes. The value of investments in COBOL assets is maintained in the majority of circumstances, even if the platform on which they are based becomes obsolete, as very few environment-related factors affect functionality or the COBOL code itself.

Compiled COBOL is supported on mainframe (z/Linux or z/OS), Linux, UNIX, and Windows environments, via tools from several of the leading vendors. The adoption by an increasing number of organizations of Service Oriented Architecture (SOA) has ensured that COBOL service routines can be easily exposed as Web services.

### *Integration is a major strength of COBOL*

COBOL programs can be used as modules by other COBOL programs, or by programs written in other languages. There are no restrictions within COBOL regarding what technologies may be integrated – any that can exchange parameter information will work. For example, using Micro Focus tools COBOL components can be integrated with those created either in Java or in .NET. This integration is part of the reason for COBOL's longevity – just as it has been adapted to changing platforms, the business services in COBOL have been re-usable throughout the emergence of new technologies for UIs, databases, or middleware.

COBOL can be object-oriented, and integration with SOA is readily available, one option being that COBOL programs parse incoming XML messages, and generate outgoing XML messages. Integration with software solutions such as relational databases (RDBs) were at one time typically achieved by including statements relevant to the RDB within the COBOL source, and running a module prior to compiling the COBOL code to 'pre-process' the RDB-related statements into COBOL 'CALL' statements, which execute the RDB modules themselves at runtime. Some modern COBOL environments such as that from Micro Focus provide automatic and transparent access to the RDB Management System (RDBMS), and are not only making it easier for COBOL programs to access business data stored in local databases such as Microsoft Access or Oracle, but also include tools to allow analytical tools such as these to access traditional COBOL data as if it were a RDBMS.

## **ALTERNATIVE STRATEGIES TO COBOL ASSET OWNERSHIP**

### *Drivers towards alternative strategies*

With regularity, views are expounded that enterprises need to plan to discontinue their use of COBOL, typically naming a number of forces that lead to this conclusion including the prevalence of alternative languages, developer preferences, and the prediction of shortages of skills in COBOL. This White Paper has discussed each of those earlier, and now outlines considerations that should be included in any plans to conduct translation of COBOL programs to other languages.

### *Language Conversion Options*

Planning to re-craft an existing application in another language can be a very inexact, high-risk undertaking. This is not due to technical unfeasibility, but to practical uncertainty leading to difficulty in estimating costs. The main element of expected costs for most application migration undertakings is that of labour, which unfortunately is one of the most expensive resources in the equation. Piecing together a program in a new language, with the objective of recreating a version of an existing program, is a highly involved task to say the least.

As a potential productivity aid, there are some code translation offerings on the market, which read in COBOL code and generate equivalent code in another language such as Java. Although such translation software tends to be expensive, the labour cost saved might make this a feasible approach, especially for organizations with extensive COBOL code bases. However, several drawbacks are associated with some such offerings, or the approach of direct code translation:

- Direct code translation often neglects to understand the application before conversion, leading to old inefficiencies being replicated, and potentially adding new issues.
- In some cases machine-generated code is more complex and harder to maintain than the original code.
- These tools have mixed records of success, especially over the long term.

An important factor to consider before embarking on code translation is whether the COBOL code base could be rationalized before being translated. There are now analytical software tools that can assess a whole application and highlight areas where functionality appears to be duplicated, or where modules exist but appear never to be used by the application. Both of these situations have a tendency to become features of ageing applications, as more and more changes are made over years of use, this being especially true where governance and use of standards may have been insufficiently stringent during periods of the past.

However, performance of the converted business application should be an advance consideration, as there have been many examples of converted business applications showing poor performance, and thus a poor return on the investment of time, effort and money. Overall, at the end of a massive conversion project, there is a real risk of ending up with the same (or less) functionality, code that is harder to maintain, slower processing, and no discernible added business value. In such circumstances, organisations often look back on the strength, flexibility, reliability, and excellence of the core business system they had, and repent the sacrifice made on the altar of mitigating any risk of inadequate access to COBOL skills.

### **Quality Assurance Issues**

A major cost factor in application migrations is whether the application in the source environment is to be replicated in the target environment, or is to incorporate refinements and improvements. If a like-for-like approach is chosen, migration is likely to be simpler and it will be easy to compare results of a test duplicated within each of the environments. Usually, however, there will be a need to incorporate elements of database redesign – quite often this is caused by technical changes between one platform and another (such as support for a legacy database type).

These matters are quite separate from changes required by the business which, in some migration projects, are propounded forcefully in an attempt to include much-needed functional change within the opportunity (as the business may see it) of a project already making changes to an application.

Any database changes necessitate corresponding changes in code wherever the affected fields are used, and add complexity to comparing test results from pre- and post-migration environments. Obviously business-led changes are likely to lead to application changes which modify existing functionality, and if so would also need alternative approaches to the comparison of test results.

Any migration from an existing platform, or change to the code base, increases the risk of introducing errors, potentially making a reliable but older application into one that is less aged but also less reliable.

### ***Business Case Issues***

There is no such thing in today's world of business as an 'IT project' – any project requires an investment of time and resources that can only be justified in the context of the business benefit from the undertaking. In the case of migrating applications from a COBOL code base, such justification is likely to require evidence that the cost of code maintenance exceeds the value delivered by the COBOL-based application. As older applications often run on unshared hardware, the operational costs of that platform may also be a factor, and in some circumstances considerations such as discontinuation of hardware support might be undeniable motivations to go ahead with migration.

In Datamonitor's view, though, almost any proposal to instigate a migration project on the sole basis of changing an application's COBOL code base to another language would fail badly at the business case stage. A business would have to accept an Armageddon scenario for COBOL skills to justify a project on that basis, and any such belief would be sorely misguided. With little hard evidence, a 'possible COBOL skills crisis' might be seen as redolent of the Millennium bug fiasco which, rightly or wrongly, so damaged IT practitioners' reputations. The proposed scale of spending, and the complete lack of visible benefit, might also be unfavourable compared with Y2K projects.

In any case, for almost all businesses the costs of moving applications from a COBOL base, with risks taken into account outweigh the benefits of wholesale migration. However, Datamonitor believes that more discrete approaches may be easier to justify. For example, changing the technologies that provide applications' user interfaces can often be founded on a good business case, incorporating extra 'stickiness' for customers, and the facilitation of customers' use of new client devices such as their mobile phones

In order to form a business case with broad benefits, and thereby balance the necessary investment in converting aged systems, some organizations have to include transformation of business processes, rather than solely conversion of programs. Although this approach may well achieve higher-value benefits, the costs and risks of such undertakings are both likely to be much greater. An example of an organization recently having decided to take this approach is the US state of California, which plans to adopt SAP-based processes for HR management, in place of its existing systems and processes based on COBOL programs. The state has been forced, to some extent, to take this step by its inability to undertake essential business changes due to lack of resources to deal with features inherent in its existing systems – however, the estimates of the total cost of the forthcoming programme of change are very high by any standards – as much as US\$177 million, according to reports on the InfoWorld news Web site.

Not every organization's migration needs, however, result in such a massive potential cost. Another local authority within California has migrated its COBOL code to native .NET, and its database to Microsoft SQL Server, rapidly and painlessly. This migration project avoided rewriting code, the system testing process was radically simplified, solution performance was maintained acceptably, and there was no need to retrain users since the existing interface screens were kept.

## CONCLUSIONS

COBOL remains a foundation of the core IT assets of many organizations across the world, and also is perfectly fit for this job. The language itself has been adapted over many years to cater for the demands of changing technologies, and demands from different IT environments, such that its integration capabilities are a primary strength. In cases where it is used for purposes that correspond to its strengths, the only valid concerns about challenges arising from ownership of COBOL-based assets arise from the ecosystem that needs to continue to supply developers trained in COBOL. It will remain COBOL users' own responsibility to enable resources that have the requisite technical training to work to gain the appropriate business skills that are necessary in a modern organizational environment.

Major players within the relevant areas of the software vendor, and IT services, markets are committed to continue to serve customers with COBOL-related needs, and in discussions with Datamonitor they speak of future opportunities and continued innovation in these markets, rather than any expectation of changing their focus.

Contrary to any ideas of the end being in sight for COBOL after almost 50 years as a platform for enterprise IT needs, it seems as likely to still be in use in 2050 as any other language currently in widespread use. Winston Churchill said, of the political system, "Democracy is the worst form of government except for all those others that have been tried": COBOL may well be the equivalent in terms of computer programming languages.

## APPENDIX

### *Acronyms*

<i>CICS</i>	<i>Customer Information Control System), an online transaction processing system from IBM</i>
<i>IMS</i>	<i>Information Management System, a database and transaction management system from IBM</i>
<i>JNI</i>	<i>Java Native Interface</i>
<i>XML</i>	<i>eXtensible Markup Language, an open standard for describing data</i>

### *Ask the analyst*

Alan Rodger, Senior Research Analyst

Butler Group

[alan.rodger@butlergroup.com](mailto:alan.rodger@butlergroup.com)

