Aviation MRO software functionality has progressed in the past five years. What has happened to the technology behind this wave of new functionality? Does it matter what software technology and language is used to create MRO software?

MRO software technology strategies

Software has changed significantly in the past decade. While the functional footprint of most of the main software providers has grown deeper and wider, vendors differ in their approach to technology. These differences create limitations, and opportunities to create different solutions for the end-user in airline maintenance departments. But do these differences really matter? What new trends are the vendors investing in to create the next solutions for maintenance and repair operations (MRO)?

Historical perspective

It was until only recently that mainframe computers were used for large enterprise applications in airline maintenance divisions. These systems were monolithic and specific to each airline. They used flat databases, with no relational structure possible in the data model, and also needed a lot of specialist support. They were limited in terms of user interface and integration to other computer systems, and could certainly not be deployed on mobile computing platforms like laptops.

In the mid- to late 1990s, applications for aviation MRO began to emerge that used Microsoft (MS) Windows operating systems, or used software with a similar look and feel. They usually operated in a client-server mode, with some of the applications loaded on the user's personal computer (PC), and some on the main database server.

Underlying these applications were new relational databases, such as MS SQL and Oracle. Many of the vendors using these technologies overcame the problem of distributing the application around the organisation by using software like CITRIX or, latterly, the MS Terminal Server. These so-called ultra-thin client solutions meant that the end-user computer simply connected to the main server over a local area network (LAN), or a dial-in telephone connection without the need to have the MRO software loaded locally. This simplified the administration, and enabled a central database to be used. Everyone had access to this so it could be updated, thereby ensuring that data was clean, uniform and validated. Hardware costs were reduced, since the end-user PC did not require a high specification for memory or processor speed.

Many older mainframe systems suffered from time lags in terms of data input and processing. The older, flatter databases could not provide the level of validation and relational integrity offered by these modern systems.

The next step forward came with the internet. As people became familiar with using an internet browser, their expectations were raised. Ease of use, clarity and simple navigation were expected. The underlying technology was also maturing and evolving.

Two main web-based technology platforms for applications like aviation MRO emerged. The first was Java, originally developed by Sun Microsystems and released in 1995. The second, called .NET, was created by Microsoft in the early 2000s, and was motivated by the emergence of Java and the growing need for web-based applications.

These applications generally provide a richer end-user environment in which to build MRO applications, and some cost savings in terms of deployment, support and life-cycle costs.

A new conceptual wave

Service-oriented architecture (SOA) aims to answer the problems that many information technology (IT) and business departments face when building a strategy for future applications.

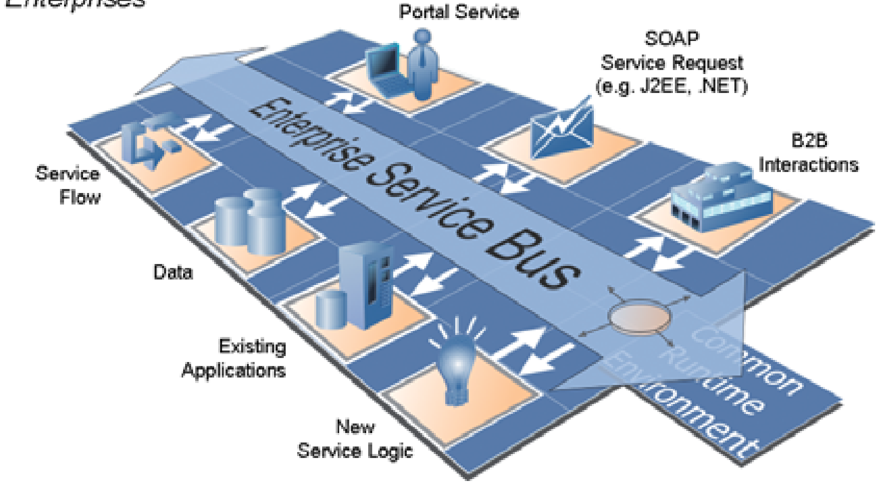
SOA is simply an IT architecture that supports service orientation, based on open standards which can include both Java and .NET applications. A service is any well-bounded, defined and repeatable business task that can be invoked in a standard manner. The scope may be as simple as a one-step task setting up a new mechanic's e-mail address, or as complex as opening and creating a non-routine task card, which involves several steps and has a number of possible outcomes.

Services can be nested. This means that one service can call another to perform a sub-task, or can even determine whether another task is called or not. This type of service integration and the linking of outcomes allows applications to be both flexible and integrated, which are two key requirements in modern business and the basis for true enterprise agility. SOA therefore enables the modelling, design, assembly, deployment and management of flexible, integrated applications from re-usable services that are independent of the applications and computing platforms on which they run.

One of the key elements of an SOA is the idea of an enterprise service bus (ESB). This new architecture exploits web services, messaging middleware and intelligent routeing, and becomes the SOA foundation. According to a recent report by the Gartner Group, the ESB acts as a lightweight, ever-present integration backbone through which software services and application components flow. It enables integration between loosely-coupled applications and services. Distributed applications can then comprise granular, re-usable services with well-defined, published and standards-compliant interfaces. The

An Enterprise Service Bus (ESB) is a new architecture that exploits web services, messaging middleware, intelligent routing, and transformation. ESBs act as a lightweight integration backbone through which software services and application components flow.

A Framework for Incorporating Web Services Across and Between Enterprises



applications send messages through the ESB to receiving applications, and also generate and consume messages anonymously. In terms of integration, web service open standards support external collaboration well. This is important for MRO software, since many aim to be the best available and so need to integrate with other applications, such as finance or human resources.

While much attention has been focused on services that can be extracted from traditional, operational applications, other types of services can be just as easily incorporated into an SOA. Among the most important of these are portal and collaborative services that are the basis for user interaction with the workflow, and are key to the improvement in user productivity required today. Portlets form the key means of user interface, and task pages provide the basic business functions. An orchestrated workflow is a business process that involves the presentation of task pages and portlets by the portal server to a user. This is based on an understanding of the particular user's role and circumstances, which support the most appropriate navigation and linkage of these tasks and portlets to the user. For maintenance, this opens the possibility of new concepts of operation or ways to deploy an MRO tool for the aviation community, particularly given that most airlines want to integrate with Airbus's FlySmart and the Boeing Maintenance Performance Toolbox software.

IBM has reviewed a number of industry projects across the aerospace industry. It found that the main benefit of SOA for users was increased flexibility in the way functionality evolved from older legacy applications to more modern ones. Most users experienced a corresponding impact on profitability. Risk and the time to develop enhanced functions were reduced, as was the overall cost of projects.

IBM uses a switchboard analogy to describe the SOA infrastructure. As with the old telephone systems, the infrastructure is what allows the front-end application to talk with the business application, regardless of where each is located. The business application can be either new or existing, packaged or custom built software, and provides the actual service.

Why care?

Why should airline executives care about the technology platform or architecture for their MRO software? SOA is an extremely dynamic approach to technology, where benefits in one area ignite benefits in others. Subsequent projects benefit from the established SOA infrastructure through concepts like re-use. IBM says that SOA is like a house, where the hardest and most expensive part of the building is the foundation. Once this is laid, the rest is easier and faster.

So does this herald a new era of cheaper MRO software? There are a relatively large number of commercial off-the-shelf (COTS) MRO software packages. The cost issue centres on the ease and flexibility for each vendor to respond to individual functional changes and interfaces with other packages like technical documents, electronic flight bags (EFBs) and the ever-growing number of add-on packages on offer. SOA seems to offer hope that costs should drop for these types of previously expensive tailored implementation efforts. A counter argument is that the flexibility of SOA could in fact drive costs up as individual airlines start wanting individual versions of the application that SOA offers. The IBM study indicates that SOA is most suitable for large IT projects with complex integration needs, particularly integration with external partners. It also states that SOA is an appropriate strategy for IT solutions supporting business processes that change frequently. Since aviation maintenance is heavily regulated, this last statement suggests that SOA will not necessarily deliver to the aviation industry all the benefits that other SOA projects have generated.

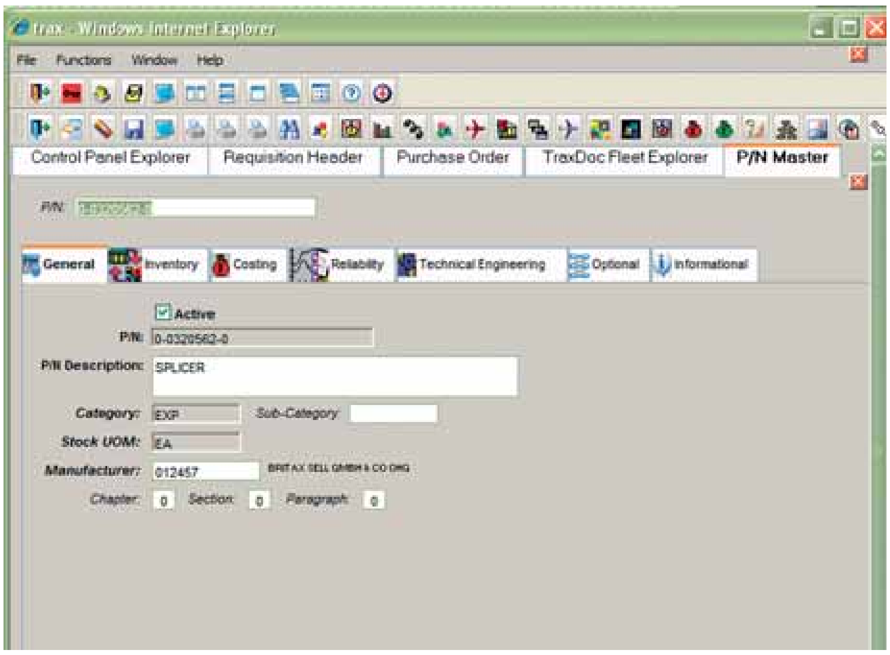
JAVA versus .NET

SOA certainly has to be understood as a concept by airline executives, who also need to establish where vendors' own internal development architecture is heading. SOA may give an edge to some vendors in the battle for new and enhanced functionality and integration capability in the MRO market. It may also lead to a reduction in software prices over time.

There is a definite split, however, in the market in terms of another underlying technology: programming language and framework. Java and .NET are the two main programming technologies used to generate most MRO applications. What are they and what are the differences between them?

Java is a programming language originally developed by Sun Microsystems and released in 1995. The language derives much of its syntax from C and C++ languages, but it has a simpler object model and fewer facilities. It has become popular for internet-based applications. For larger and more complex applications across a company's organisation, the so-called enterprise edition specification was originally released by Sun Microsystems in 2001. Java Platform, Enterprise Edition or Java EE was introduced in 2006 as an evolution from J2EE. Sun released much of Java as free/open source software during this time. JEE is used by software companies to provide functionality to deploy fault-tolerant, distributed, multi-tier Java software, based largely on modular components running on an application server. One advantage is that it is platform-independent for developing, building and deploying web-based enterprise applications on line.

At the client tier or user side of an



P/N:

Active [✓]

P/N: 0-0320562-0

P/N Description: SPLICER

Category: EXP Sub-Category:

Stock UOM: EA

Manufacturer: 012457 BRITAX SELL GMBH & CO OHG

Chapter: 0 Section: 0 Paragraph: 0

TRAX has decided to start migrating its client-server version of the application to a .NET technology. It is still using Powerbuilder to make the user screens look exactly like the old product, to ease training issues.

application, J2EE supports pure hypertext markup language (HTML), as well as Java applets or applications. It relies on Java Server Pages and servlet code to create HTML or other formatted data for the client. Java Database Connectivity (JDBC), which is the Java equivalent to ODBC, is the standard interface for Java databases.

Larger software companies have even developed their own JEE frameworks. One example from the MRO market is SAP with its NetWeaver Application Server. Other software vendors that have chosen the Java development route are Swiss AviationSoftware with AMOS, MXi with Maintenix, and MIRO with AuRA.

.NET

The Microsoft .NET Framework is a software component that can be added to the Microsoft Windows operating system. .NET provides a body of pre-coded solutions and manages the execution of programmes written for the framework. The .NET Framework is a key Microsoft offering, and is intended for use by most new applications created for the Windows platform. Originally written for the Microsoft relational database SQL, there is some capability to run the application on Oracle databases.

Pre-configured .NET solutions cover a range of programming needs in areas including user interface, data access, database connectivity, web application development, and network communications. The .NET programming functions are combined with their own code to produce applications. An important difference is that .NET is only fully available on Windows platforms, whereas Java is fully available on many platforms. Microsoft's implementation of .NET is closed source, whereas Sun's reference implementation of Java is becoming open source.

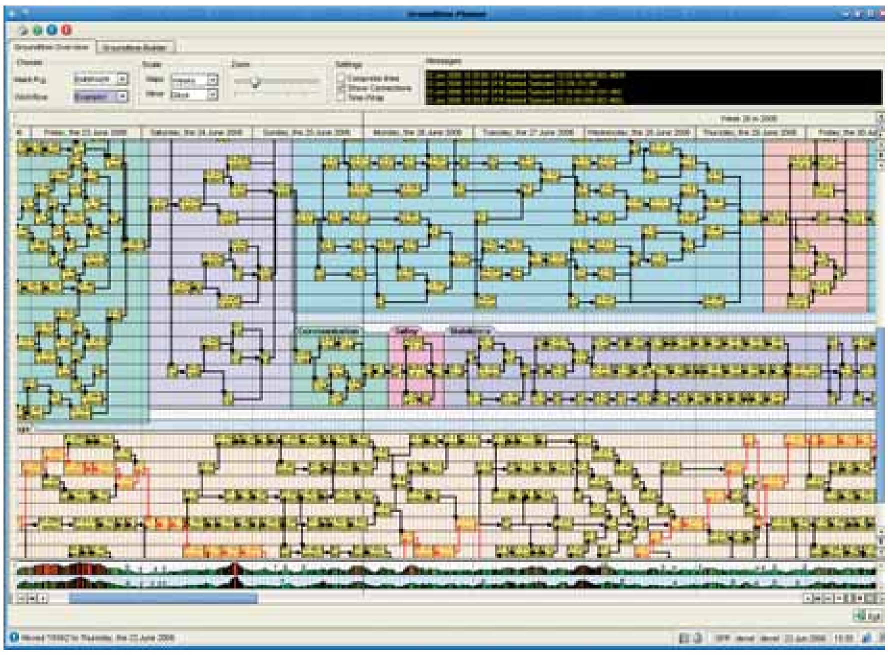
.NET is growing in popularity for building web applications, due to the availability of tools for building using the technology. There is also a growing skills base of .NET professionals available to build these tools. There is no doubt that Microsoft's future is solid, so the technology seems to come with a degree of future-proofing.

On the .NET side there are a number of strong MRO software players including Russell Adams (RAL), TRAX and Visaer, which have chosen to re-architect their products using the Microsoft toolset. Interestingly, these are all vendors that have recently embarked on their new developments, with the exception of RAL, which started a re-write of its application three years ago.

Hedging your bets

Of course, for those vendors with significant resources the choice of web technologies does not have to be only Java or .Net. Ramco is a vendor that can deploy in either environment. "Ramco's MRO solution can be applied to different target technologies," says Jim Fitzgerald, president global A&D, aviation & MRO solutions at Ramco. "The solutions are available in either .Net with Microsoft SQL server database, or in Java with Oracle Database on Unix, AIX or Windows. The product has also been rolled out on other platforms and technologies like IBM's DB2 database. The platform development environment is Ramco's business process platform (BPP), formerly known as VirtualWorks. This flexibility is provided by our patented BPP framework. The solution is based on a model-driven architecture, so it is web services enabled, collaboration-ready and business-process-component driven. The final application is largely independent of target technology, and can be made available on multiple platforms and combinations thereof. We believe that this provides us with cost advantages for generating and delivering the application in a new target platform. We can also make changes to the current application based on the customer's requirements, or add features for new releases. These benefits are available irrespective of the choice of target platform.

"The application is available off the shelf in either Microsoft SQL Server or Oracle databases," continues Fitzgerald. "We can deploy or deliver the application on other database choices made by the customer. The customer gets a ground-up, web-architected solution on its choice of hardware platform, so they are able not only to pick the technology option, but also the modules or business components they require. They can also choose to integrate specific areas with existing systems, or replace them with Ramco's business components. Another important point is that business processes change continuously, so our BPP framework has the additional flexibility to adapt to customers' specific needs. The key benefits of BPP are that business processes can be initially specified and captured to create a repository. Changes or new applications can then be visualised in their final form before development. This reduces surprises and risks later in the project. The impact of these changes can be measured and understood prior to execution. A customer extension to mobile devices and browsers is also available, providing mobile and off-line access to our application in specific business process areas, such as journey log entries, using personal digital assistants (PDAs) or mobile platforms. Ramco's strategy is to follow the latest developments in IT thinking. For example, our development has been using a SOA framework. Our on-going research and development, together with our technology partnerships with the platform and operating system organisations, are a strategic value to its customer base. We claim to be the first to have developed and deployed an MRO application in a pure web-component environment in late 1999, when other software product



AMOS MaintExecution.pngSwiss has opted for Java to implement its MRO solution, providing a web interface for their customers.

vendors were still releasing their old client server applications."

The .NET approach close up

Most specialist MRO software vendors have to choose one technology path. One of the early adopters of .NET was Russell Adams (RAL). "We started to re-write the system in late 2003," says Richard Vorias, group business development director at RAL. "We chose to write it in the new Microsoft .NET language with Bombardier, one of the customers of our older system. The main reason for choosing this technology was that it tightly integrates across the MS Office suite, which means that we could use those other applications, like MS Excel, seamlessly in our functionality. Another important factor was that many of our key customers were using the MS SQL relational database to underpin their business. When we thought about usability, we also felt that it was important to offer maintenance and engineering people a look and feel that they were used to in their day-to-day lives. MS is immediately familiar to most people.

"In 2005 we rolled out the new .NET application on MS SQL Server," continues Vorias, "and started to see more benefits for customers in our new approach. As well as the tight integration, we were quickly able to offer flexibility in terms of programming new add-on functions, modifications and screen layouts. Users would say 'if only it could do this' and we would say 'no problem'. What helped us to deliver this new flexibility was the availability of high quality programmers in the new .NET technology. We have been lucky to recruit and retain an excellent software development team at RAL."

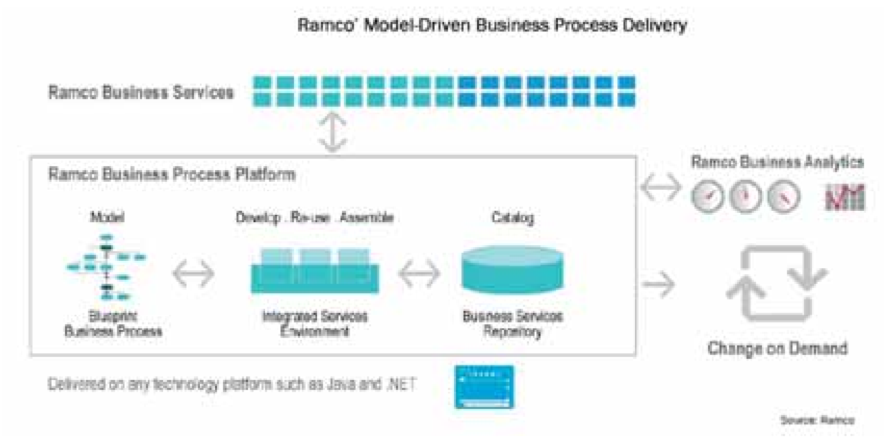
In early 2000, the choice between .NET and Java was not clear cut, and the advantages of the two programming technologies were being hotly debated. NET had some limitations in terms of providing an Oracle database capability, which was one reason why many were dissuaded from using .NET applications. Steps have recently been taken to broaden the range of available relational databases with which .NET can operate. Some of these problems are removed with the advent of my SQL, which now enables Oracle databases to run with .NET. But there are some nice features of .NET that provide end users with important benefits, particularly for mobile environments like aviation MRO. One example is the use of 'smart-client' architecture. A smart client is an application that uses local PC processing, consumes web services and can be deployed and updated from a centralised server. Old client/server solutions have been notorious for their problems with deployment and installation, which have been the main reason why many companies decided to build web applications. Using a centralised server has made the application available on a simple web browser. Smart client is an evolution of these solutions that focuses on making an application extendable to any front-end environment that supports web services. However, smart-client technology is not a silver bullet and its use needs some careful attention. "We saw that .NET would enable maintenance users to work off-line when they were remote but not connected to the network. This can be vital in a hangar, or out on the flight-line. With a .NET smart client, part of the application and data are transferred to the user's local PC, so that they can continue to work on that data. The data is then re-synchronised when they come under the wide area network (WAN) umbrella. This is a small but important point. If the user relies on an internet or LAN connection to use the system, their mobility is limited".

There are of course some potential downsides to having smart clients. First, the client's PC has to be of a higher specification to allow it to run local applications and store parts of the database. This implies that some higher costs will be incurred for hardware. Most basic PCs are more than adequate, however, so the issue only really applies to organisations with older hardware in the hangar and around the engineering support departments. Second, clients running software locally will have to be updated with the latest versions of the software application. This includes patches and bug-fixes and so on.

"With our smart-client configuration, the system is still centrally administered," says Vorias. "Just as the off-line user re-synchronises data when they come back on line, they will receive any application updates at the same time. This means that they will always be using the latest version of the system."

If large or complex updates have to be completed, this could cause delays in accessing the system. But most people today are very familiar with booting up their PC and waiting while the latest MS Windows patch is downloaded and configured automatically.

The whole re-architecting process took RAL 24 months from the start until its first release of production software. During this time RAL took the opportunity to add some additional functions. One of the first customers was MyTravel, based in Manchester, UK. "Working with MyTravel was important to develop and hone some of the airline-specific functionality," comments Vorias. "We have expanded the application to include an e-techlog on board the aircraft and complete flight operations systems for planning and scheduling crew and aircraft rotations. The integration between flight operations and maintenance is a critical area for many operators. While we have made our own application, we can also integrate with some of the big operations software names like Rocade, Bornemann, Lufthansa Systems and AIMS. This is because we have made the links already, and re-connecting them to another software package is simple."


Ramco' Model-Driven Business Process Delivery
Ramco Business Services
Ramco Business Process Platform
Model
Blueprint Business Process
Develop . Re-use . Assemble
Integrated Services Environment
Catalog
Business Services Repository
Ramco Business Analytics
Change on Demand
Delivered on any technology platform such as Java and .NET
Source: Ramco

The Ramco Business Process Platform automates the process of building SOBAs by integrating dynamic process modeling, a Business Services Repository, and a Service composition environment.

Evolve or die

Flexibility seems to be one of the main advantages with .NET. It is a less expensive development environment and modifications and enhancements can be brought to market more quickly. There are also some advantages in terms of easily tailoring individual screens. This is a perennial issue for end users trying to adopt a new system *(see Successful MRO system implementation, Aircraft Commerce, February/March 2007, page 48)* and any technology that can help users become more comfortable, more quickly, will have an edge over the competition.

RAL claims that it can easily re-configure screens and layouts to suit each customer. Controlling these multiple configurations might be an issue, but RAL says that its technology can handle this. "The additional flexibility of .NET lies in data presentation," explains Vorias. "While we can embed MS Excel charts within the application, we can also use the native MS report-writing capability to help structure data presentation more effectively. We are now experimenting with executive dashboard techniques in .NET. These can present data at a summary level for busy managers. This includes traffic lights, coloured warnings and key performance 'speedometers'."

Flexibility not only applies to the software screens, but also to the underlying business logic. Many software vendors claim to be able to mould their software to match each individual customer's business workflows. This is a marketing and sales story, which sounds good, but falls short during a real implementation. This is because it is too complex to change and adapt the workflow rules without breaking or compromising an interlocking business function. RAL claims that re-configuring business logic is at the core of its approach. "We can re-write rule-based logic in our .NET architecture really easily," claims Vorias. "The .NET technology is an enabler to our overall philosophy of adaptation and flexibility to match customer requirements. Many of our competitors struggle to match this. We are happy to see that some of the recent software vendors, who were on old technology, have decided to join us in the .NET camp. Of course we are five years ahead of them already, and intend to stay that way."

Joining the .NET herd

One of those joining RAL in the .NET camp is Trax, one of the most successful specialist MRO software providers. Trax has been selling its client-server-based software for several years. Only recently did it decide on a web-based technology path and choose the Microsoft route. But it has taken an interesting approach. "The current production platform for the application itself is Windows 32 but running on Oracle," says Chris Reed, managing director of Trax Limited. "However, we are moving to .NET technology during 2007. In fact we use a mixture of .NET and Java to provide us with the ability to develop a common and standard platform. This allows us to use already developed code in our application, and leaves us free to focus our efforts on our client's business needs. There are definite cost benefits. For example, a customer that wishes to deploy our application as a .NET web application over Citrix remote desktop will save in Citrix licensing costs. In terms of SOA, Trax already supports various methods of SOA with our system, such as making it easy to exchange our system's data with our client's other systems, as well as accessing the data and functionality of the system through a client's own desired means.

"However, since .NET is new and somewhat of a 'virtual machine', there are still some functionalities that are not yet supported," explains Reed. "For example, there are some hardware devices that were created to directly interact with the operating system, such as data readers. Commercially the .NET version is offered as a free upgrade. Currently we deliver a Windows32 EXE application. Soon we will deliver the same application as both a Win32 EXE and a .NET Webform application. For us, it is the same system, just two different ways to build and deploy it. Our current customers will be able to use both deliverables, or choose between one or the other at their own discretion. We are still using Powerbuilder at the user frontend, and in most cases people will not notice any difference in the screens they are using in our old and new versions of software."

Summing up

Airline executives, and not just their IT departments, need to be aware of the major issues surrounding JEE, .NET and SOA in particular. While the starting point of any MRO software project has to be the individual airline maintenance and engineering functional requirements, some attention must be paid to the vendors' technology path. It seems that there is no general answer as to whether .NET or Java is a better route. Each has its benefits. Ultimately an airline's decision comes down to how the software vendor has used the technology to structure a business application, and how much flexibility an individual customer wants in terms of flexing the initial off-the shelf software tool to match its business needs. One thing is for sure. Another new technology will soon inevitably overtake both .NET and Java, and software vendors will need to be able to invest sufficiently to stay on, or ahead of, the technology curve in order to survive in the long term. It is interesting to note the trend at the moment for vendors who have recently embarked upon web-based technical refreshment of their products. Most have opted for the .NET route, largely it seems because it is less costly to move to and can be done more rapidly. Some argue that Java is a better environment for more complex enterprise applications. It seems that it is more important to look at the underlying architecture, more than just the programming language. AC