MIDDLEWARESPECTRA

incorporating FINANCIAL MIDDLEWARESPECTRA

Contents

May 2004

Volume 18 Report 2

Runway — integrated middle-ware for the travel industry

Mark Lenahan
Senior Architect
SITA

Management introduction

Mark Lenahan is a Senior Architect at SITA. He is currently involved with SITA's Runway portfolio of products and their development. Runway was originally developed by an Irish software company based in Dublin called Eland (www.elandtech.com), which SITA acquired in late 2003.

SITA provides information and telecommunication solutions to the global air transport and related industries. It has more than 50 years of experience with members and customers in over 220 countries. Not only does it provide global network services — including IT infrastructure — that enable air transport organizations to operate seamlessly, but it offers applications and integration services.

Headquartered in Switzerland, SITA has some 3,400 staff, with more than 140 nationalities speaking 70 languages. The fact that SITA is owned by the aviation community (airlines, airports, aerospace companies — organizations involved in aircraft design, maintenance and communication — plus logistics organizations, international organizations and governments) means that it (SITA) has had to develop the relevant expertise to support:

- ***application services***
- ***end to end desktop and infrastructure services***
- ***network services, including systems integration, outsourcing and consulting.***

SITA's value-added services are offered through SITA INC (Information Networking Computing), which is headquartered in Amsterdam, the Netherlands. Network services are managed through SITA SC, which is headquartered in Geneva, Switzerland.

In this case study, Mr. Lenahan discusses:

- ***how Runway was developed***
- ***how it has evolved to provide advanced middleware and integration to air transport organizations***
- ***the architecture that provides this***
- ***how it can work with other middleware, specifically Tuxedo and MQSeries***
- ***the lessons that have been learned during its evolution.***

Setting the scene

The part of SITA I work for — what was previously Eland Technologies — is primarily a software company. Originally its customers were mainly travel agencies and airlines, which is why SITA bought Eland. Our products fit the SITA business model which is tightly focused on air transport.

Although SITA overall is a non-profit organization, it is divided into two 'entities':

- **the main non-profit part is SITA SC and its primary role is the delivery of networking and messaging between airlines and airports**
- **the for-profit part, which deals more with applications, is SITA INC and that is the part which bought Eland.**

SITA started by providing networking for the airline industry because messages — even prior to the Internet — needed to be sent between travel agents and airlines and airports around the world. As you might expect, some 50 years on, the requirements are broader, deeper and significantly more complex. Although the original focus was on providing networking and messaging — for example from terminals at check-in desks at airports back to airlines which might be in some wholly different part of the world — it has gradually been encouraged by its members and customers to offer applications which sit on top of the SITA network. This is what SITA INC (Information Network Computing) is focused on.

The evolution of Runway

The original Runway was oriented around emulators for applications running on legacy mainframe systems. The airline industry was an early adopter of large mainframes in the 1960s and 1970s. Over time these systems have become both more specialized and more refined. One example of this is that there are really only three or four different types of system in the industry — but these are spread over dozens of different hardware and software environments and protocols.

Eland Technologies entered this environment to meet the needs of users of these different systems, who wanted to simplify their communications. Our emulation product, Link, offers this. Our primary customers for Link were travel agencies — which needed to talk to the multiple reservation systems that the various different airlines possessed. This was, in effect, a kind of two tier client/server environment — the emulator connecting into the legacy server. Link took care of presentation onto the screen.

Subsequent to this, there was a demand for improved access to booking engines and, especially, enabling Internet technologies. To satisfy this demand we took much of the emulator technology and packaged it as a full server product. That server product is what is today known as Runway.

In effect the purpose of Runway has changed from that of Link. Runway's objective is not so much to make legacy systems easier to see from a user's point of view. Instead the focus has changed. The new objective is still to enable connections to legacy airline industry applications, but it is to make these easier to use from a modern applications point of view. Effectively we have evolved:

- **from being a user interface enabler**
- **to offering a pure middleware platform.**

Let me explain a little more about what I mean. Effectively what we had in the emulator was a lot of connectivity software and some presentation software. There was only a limited capability for creating application logic, and this had to use our own scripting tools on the emulator. This was far from ideal — in terms of exploiting modern application server-based or Web-based development tools and techniques.

To bypass the constraints, we took our existing connectivity experience and re-cast it into a server product that acts as a gateway into the travel industry's legacy systems. In addition to that, we added the ability to perform application data mapping: we can convert from legacy systems' native languages into XML or into an object-based interface. In effect our Runway server:

- **exploits the knowledge of the legacy protocols and connections to existing mainframe applications**
- **brings the data to the server where it is rendered into XML (or whatever else is desirable)**
- **offers the result to the modern application(s) where modern application logic can be applied to perform the tasks required.**

In one sense this is like a layer on top of what we had been doing already. But it is more than that. It is a move into providing a formal client/server architecture. It also moved us away from our traditional travel agency customer and more towards being used within an airline as an integration medium. Again, let me explain.

The travel agency dimension

Most reasonably sized travel agencies possess a dedicated IT department. This is, however, probably focused on developing in-house solutions. The value we offer is that these travel agency in-house solutions do not need to be tied to specific legacy systems — which the travel agency is then obliged to use. Effectively we decouple new application developments from the legacy applications they have to use.

Why is this important? There is considerable competitive pressure on travel agents from the Internet. The latter enables anyone to connect to anything, if the appropriate back end connections have been made. For example, you or I as end users can have several different browser sessions open to different airlines simultaneously.

Generally, airlines do not provide integrated tools to travel agents. The way airlines talk to travel agents is via an intermediary legacy host system called a Global Distribution System (or GDS). In effect, an airline buys its reservation facilities from a GDS provider like Sabre, Galileo/Apollo, Amadeus or World Span (the big four).

The way the business model works explains why change is occurring. The GDS providers charge airlines for distribution. They only charge travel agencies for that which covers GDS travel agent support costs and, if a travel agency sells above a certain value, the GDS provider basically offers everything free to the travel agency. In some cases it even does custom application development for the travel agency.

From a GDS provider viewpoint the objective is to encourage ever more intense use by the travel agency of that GDS, which is then charged to the airlines. The airlines, however, do not much like this. They are paying for their distribution costs twice — to the GDS and to the travel agent — which can cost US$8-10 per ticket. Today, one of the imperatives for airlines is to try to reduce the cost of using their GDS; another is to obtain more of what they call direct sales — in effect selling over the Internet and bypassing the GDS and even the travel agencies. Direct sales are the big business challenge at the moment in the airline industry. Every single airline has among its sales targets the desire to increase direct sales. The need has been emphasized by low cost carriers — like Ryanair, easyJet and JetBlue — all of which have significant Internet sales operations (they need to when they are selling US$25-30 tickets).

Now think about the travel agency. This is being squeezed. Customers are going on the Internet. Airlines are trying to reduce what they pay to travel agencies. The GDS providers are also exposed. Furthermore, a travel agency may need the facilities to access two or more GDSs in order to provide a full service across all main airlines to customers. But this can be expensive.

Runway fits in where, say, a travel agency wants to improve its use of a GDS. A travel agent would:

- **put Runway 'on top of' the GDS**
- **develop its own applications to access a first GDS**
- **potentially expand from one GDS to two or more.**

This offers two major advantages:

- **the first is the ability to use multiple GDSs; one application might talk simultaneously to both Sabre and Galileo**
- **the second is related; the travel agency is no longer tied to a specific GDS, as it was in the past.**

This creates new business opportunities for the travel agency — perhaps because it can advertise that it can find the cheapest fares (which is now possible because of access to more than one GDS). In addition, because the travel agent is no longer tied to a specific GDS system, its agents need reduced training and less specific training.

This means that costs reduce as well as opening up the capability to switch (say) from Sabre to Galileo or Amadeus without re-training. In turn, travel agents can now renegotiate with the GDS provider, for example to reduce costs or obtain additional facilities.

In the travel agency instance, what Runway has done is to act as middleware between:

- **the applications that the travel agency writes for its agents to use**
- **the GDS.**

There is one other dimension to this, although it would also be fair to say that the Runway option is probably only relevant to larger travel agencies. This is that a travel agency with Runway can bypass the GDS system altogether and connect directly into some airline reservation systems. The airlines that support this like it because travel agency sales effectively become direct sales.

Again, Runway acts as middleware sitting between the travel agency applications and the airline's applications. The middleman in the distribution has been cut out, which is why Runway has a number of big travel agency customers — even though (as I will explain) the product is even more attractive to airlines.

Runway and airlines

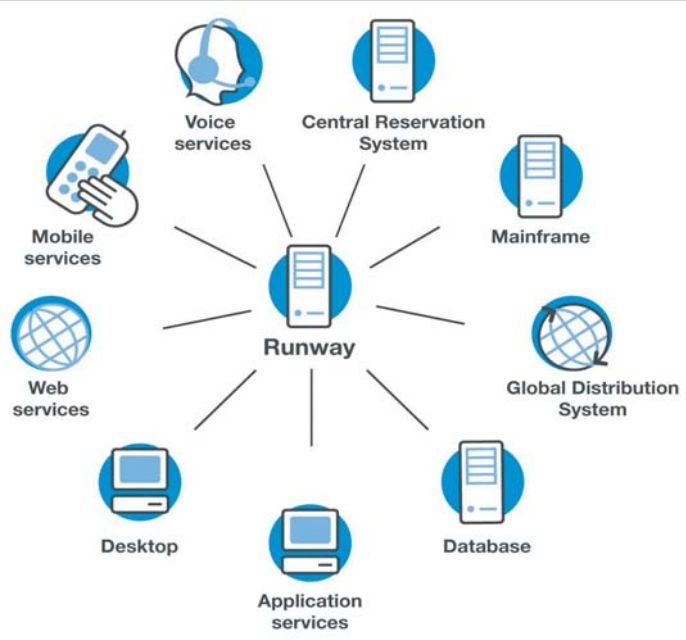
An airline looks at Runway quite differently. The GDS dimension tends not to be there (although it could be). Rather Runway is used internally, between in-house legacy systems (like the reservation system, the inventory system, the airport system, etc.) and new applications. This area has become strategically important to airlines because many of the 'traditional in-house legacy systems' are no longer run 'in-house': they have been outsourced to third parties.

Although it is well known in the travel industry, it is not so well known outside that so much outsourcing of key applications has occurred. Even some of the very largest airlines pay external companies for their reservation hosting. This is rationalized, in many instances, by the airlines often owning parts of reservation companies or GDS companies. At the smaller end, new (or small) airlines are more likely to buy reservation system expertise rather than attempt to build their own ones.

Given this, Runway is mainly used internally within an airline to solve its own internal application integration problems. The return on investment comes because an airline does not have to reinvent the wheel in terms of connecting its legacy systems to its new applications. Runway shields the new applications from those legacy systems, yet opens up access to those multiple legacy systems.

For example, we have at least one customer which has recovered its Runway licensing costs simply by re-negotiating its reservation hosting agreement on the strength of the fact that, with Runway (much as with a travel agency) it could now consider changing its chosen reservation

Figure 1.1: Runway I



systems because its applications were no longer tightly coupled to the existing reservation system.

Another key differentiator is that Runway comes with 'embedded knowledge' of legacy airline systems. We have worked with nearly all the various different airline systems (Figures 1.1 and 1.2) at one time or another. This means we have been able to incorporate knowledge and logic in Runway which accelerates the way in which new applications can work with legacy travel systems.

Indeed, this is also what distinguishes Runway from so-called integration engines or application servers like WebSphere or WebLogic:

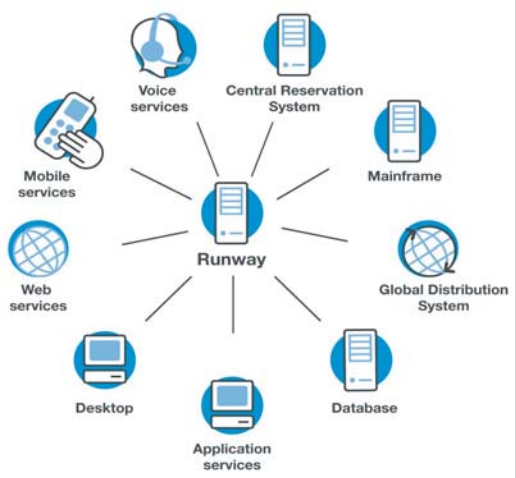
- **first, we possess the understanding of the legacy systems**
- **second, we offer an approach that is customized for the specific peculiarities of the travel industry**
- **third, we do not have to satisfy 'generic' integration requirements that lie outside our industry.**

To give you a flavor of what I mean, the airline industry is utterly different from (say) the finance sector. Most legacy airline systems run on IBM's VM or TPF operating systems or on Unisys (Sperry) ones. In contrast, the finance sector concentrates on MVS with IMS or CICS, with some Unisys (mostly from the Burroughs heritage). In addition, there are special sets of protocols that are specific to the travel industry (which are quite different from those used in finance). We still use legacy protocols that are peculiar variations on X.25 and so on.

My point is that although we specialize in connecting to the likes of Sabre, this is only half the battle. You need to communicate, and you also need to be able to send Sabre commands and understand Sabre responses. This means that there is more than merely communicating if you want to work with the modern world of client/server, Web-enabled applications, application servers, Web Services, etc. To do this you need to map from legacy formats into something like XML, and vice versa. What Runway delivers is the ability to abstract the various different travel industry legacy host systems into forms that modern applications can use.

We deliver this added value middleware in not only the communications but also in travel industry-specific XML mappings. With these in place, it is now easier and faster for airline developers to create new applications as well as change existing ones — without having to deal with, or alter, legacy systems direct. This produces significant and short-term returns on investment — because middleware (Runway) is delivering the integration capability.

Figure 1.2: Runway II



Typical host connectivity supported:

- **MATIP including Types A and B, HtH**
- **X.25, AX.25**
- **Galileo and Apollo over IP (TCD v1 and v2)**
- **Gabriel/Horizon**
- **TN3270**
- **IBM MQSeries**
- **Unisys INT1 over TP0**
- **TCP/IP sockets**

A business to business dimension

But that is not all. There is yet another side — in terms of enabling business to business (B2B) integration. Unlike the car industry, airlines have taken time to start forming consortia to assist B2B integration. Part of the challenge is that there are so many different possible permutations:

- **airlines to other airlines**
- **airlines talking to GDS companies**
- **airlines interacting to airframe and engine manufacturers**
- **airlines communicating with regulatory agencies and governments**
- **airlines working with complementary business partners — such as car, hotel and insurance companies or caterers**
- **and so on.**

This variety explains why so many airlines are interested in adopting Web Services. Modern B2B integration technology can enable new combinations of businesses. But this is only achievable if they can provide some form of XML service-oriented integration that will also link into all those legacy applications (Figure 1.3).

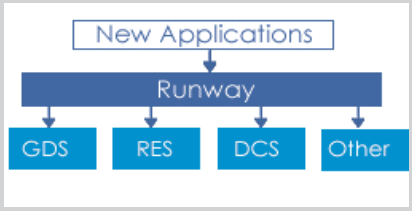
Runway architecture

At its simplest, Runway is basically a C++ application running on UNIX. It is a single code base that runs across multiple operating systems. It has all been developed in-house. In one sense Runway is, however, different to most middleware. It is an application which sits in the middle providing the integration capabilities between legacy applications and modern applications. It is not a 'middleware' piece — such as you might regard MQSeries or SQL*Net or whatever. In Runway, its totality is an incarnation of middleware: it sits in the middle, between the legacy and modern applications.

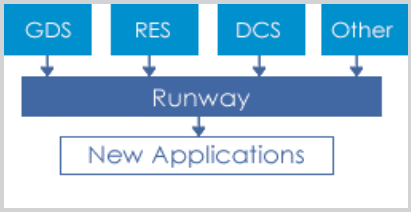
The only third party elements we use are some components from the Apache project — such as Xerces — which gives us our XML handling. Where we provide SOAP, we do not implement that ourselves. Instead we use a Web Server, such as Tomcat or WebSphere, WebLogic, etc. All of these have a SOAP container and we write the services within that container.

Effectively Runway is an arbiter (between the new and old). It is an integration server. An application server will connect to Runway which will take the incoming XML message and turn this into a format — for example, to check availability or to create a booking — which the legacy system can understand. Runway converts the traffic (requests and responses) and deals with all the application specific features that are needed to work with those legacy applications.

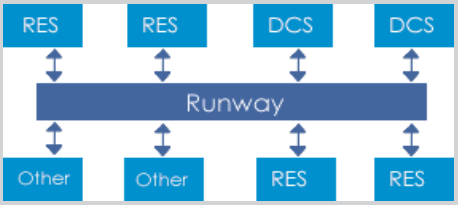
Figure 1.3: Typical integration scenarios



New applications access legacy host data and services



Legacy host applications access new applications and data



Legacy host applications in a multilateral hub

If you were to draw a picture of this you would have the legacy applications at the bottom and the blanket over these would be Runway. Above that would be the new airline applications. The benefit is that new applications can be developed and can work without dependencies on legacy systems either inhibiting capabilities or making each new application too expensive to create. The result is that an airline can interchange the legacy systems underneath the blanket with modern applications above — without necessarily affecting the underlying applications.

It gets better. You obtain re-use. Runway is more than just a way to negotiate better reservations hosting. The way we find Runway being used more and more is to build a message set on top of the legacy applications. Because this is in XML, you can then access it from any number of new applications. This means that you can start building applications which had, previously, had the massive hurdle of legacy integration to overcome. With Runway, you bypass that hurdle.

For application servers to talk to Runway there is an API, or you can use our Web Services layer (the Web Services will run in a Web Server SOAP container). In this situation you do not have to use any Runway-specific code in your application.

Transaction handling

For transaction integrity, if required, we can support Tuxedo as a transaction monitoring environment. For example, you can talk to Runway using Tuxedo. We also support fail-over and load balancing in our basic API. When using Tuxedo or SOAP, we support whatever external fail-over or load balancing mechanisms that you might choose.

Runway is also cluster friendly. Most of our customers will run Runway on a cluster of two or three or four physical machines. Hotwire, for example, runs Runway on a cluster of twelve Solaris machines. Actually, that is not quite accurate: the numbers in the cluster go up and down depending on the Hotwire operational requirements. They call their ISP and say 'we want more blades in our server': the ISP configures that to provide computing on-demand.

Let me now widen this dimension. We do not attempt to do some of what (say) IBM's MQSeries does. Similarly, we do not do some of the things that Tuxedo does. Yet it is both interesting and worthwhile to draw contrasts between Runway and both of these popular middleware products — not least because we have several customers who are users of Tuxedo and/or MQSeries who have bought Runway (and still use Tuxedo and/or MQSeries).

Tuxedo

In the case of Tuxedo, Runway is not a transaction monitor. If you want to write code that runs in a transaction-monitoring environment, you really need a transaction monitor like Tuxedo. Say you want to write code and you know that it (your code) is 99% available but in 1 in 100 transactions produces an error. You know you can achieve better than this by wrapping that code in a transaction monitoring environment. You do not reduce the error rate but you do vastly increase the availability. No matter what that code does, no matter how badly it behaves, a transaction monitor means it is always available.

This is one answer to higher availability systems — especially when you are doing in-house custom development. In in-house development you will likely never possess the security of having hundreds of thousands of customers all using the same code. That is why a transaction monitor assists. It is almost a matter of self-discipline.

Now in the case of Runway we can work with these — the transactions that you may be writing. Some of those transactions maybe access a legacy system. In this instance, a transaction will fit with Tuxedo above Runway and Runway will be the blanket that separates those transactions from the legacy systems.

In some cases people will regard the transactions themselves as being part of the legacy environment and they might wish to provide additional application code written in Java or J2EE. They could use Runway to combine some of those Tuxedo transactions with direct access to applications running on different legacy systems. Or they could run Runway above the transaction processing layer (although it is more usual to have the Tuxedo transactions running above Runway). Runway is simply providing the legacy integration logic.

For some it is much easier to write transactions. Transactions are the business logic that sits outside (say) the legacy reservations system. This why people in the airline industry are using Tuxedo or typically writing new applications in C or C++ or Java. They do not want their J2EE applications coupled to legacy ones just as much as they do not want their Tuxedo applications coupled to legacy applications. This is why Tuxedo and Runway can sit comfortably side by side.

MQSeries

In practice MQSeries — which is also widely used in the airline industry — is similar. MQSeries is a persistent queue manager. We do not offer an equivalent in Runway either.

We do not have persistent routable queues even though you may be better off transporting something like XML via an MQSeries queue — because you can then perform all kinds of data dependent routing as well as having applications that receive that XML be completely independent from any legacy host.

Again you could have MQSeries operating between an application and a legacy host. But you might find it much easier to use Runway as an adapter. This contrasts with the MQSeries Integrator (MQSI) approach, which would put MQSeries on the legacy hosts. But you do not always have this option if the legacy host is hosted by a third party.

In this specific instance I would have to say that MQSeries Integrator is a competitor to Runway. There are instances where you might prefer to use MQSI instead of Runway — or Runway instead of MQSI. They are different forms of middleware, with MQSI being generic while Runway is specific to the airline industry. Indeed, there may well be occasions when having both makes sense.

In practical terms I would say that MQSI might be used better where many legacy applications already use MQSeries. That said, Runway can even help enforce MQSeries or reinforce an MQSeries strategy, because it provides access to legacy hosts where it is not possible to install MQSeries.

Lessons learned

My lessons learned may seem obvious — but they remain applicable. The first is: educate the customer on exactly what your product or service is, especially with regard to what you do (and don't do). Fortunately for us there has not been a big mismatch between expectations and delivery. But even a small mismatch can turn into a big issue.

Similarly, be aware that regardless of what you say, sometimes people hear what they want to hear, especially when they are focusing on a persistent problem. The way around this — as a vendor — is to:

- **focus on the customer problem also**
- **be clear on exactly how much of the problem you will be able to solve.**

As a customer (or vendor), do not jump on band wagons especially with regard to new technologies and development tools. Look at what new technology claims to do. Look for stability. Ask for examples of proven RoI. Then figure out how you will obtain your RoI. Sticking with this RoI theme, as a vendor you cannot reasonably 'tell' people what their RoI will be. You can only give examples. Sometimes it is surprising how much return an organization can make out of a software product.

In a broader sense, customers will constantly surprise in the ways that they will find to use your product, often for purposes you did not imagine. That is the mark of generic software. The trick is to enable this and to learn from it, for example by enabling or doing customization.

As such, do not be a stickler for 'product purity'; it is the customer who decides what is useful in a product, not you. We have learned to encourage our customers to take our products in new directions — and have benefited.

Management conclusion

There is a growing trend in the airline industry — which has traditionally been quite insular and, in some cases arrogant, in terms of its belief in its own technology — that middleware matters. The need for integration is driving this. Over the last three or four years much has changed. But what has not is the dependence of airlines on existing legacy systems. But these do need to work with modern tools and techniques, and not least the Internet and Web Services.

In this case study, Mr. Lenahan uses Runway to illustrate how it is applicable to two different parts of the travel sector — airlines and travel agencies. He has also illustrated how the value add of middleware that is tightly focused on a specific industry can be rather different to more common middleware components and yet can still work with the latter (as the MQSeries and Tuxedo examples show).

Grids for the enterprise

Shane Robison
Executive Vice-President and Chief Strategy and Technology Officer
Hewlett-Packard

Management introduction

Shane V. Robison is Executive Vice President and Chief Strategy and Technology Officer for the Hewlett-Packard Company. In this role he is responsible for shaping the Company's overall technology agenda and for leading strategy and corporate development efforts including mergers, acquisitions, divestitures and partnerships. He leads the technology and strategy councils as well as the development of future technology roadmaps, working closely with HP's business units and HP Labs.

Before this, Mr. Robison was Senior Vice President and Chief Technology Officer of Strategy and Technology at Compaq Computer Corporation and President of Internet Technology and Development at AT&T Labs.

In this analysis, Mr. Robison offer his, and HP's, view of the Grid and Grid technologies. As becomes clear, he has a distinctive sense of where Grids are going, the broader context into which they fit and why they are important now and will continue to be important in the future.

Driving forces

Grid had its beginnings in the mid-1990s in scientific computing; but the notion of distributed computing has been around for decades. Grid was originally conceived and designed in this community to allow access to computing resources that were geographically dispersed. The notion was that underutilized resources in places other than where the researchers were physically located could be used. Also fundamental in the formative thinking was the prospect of sharing access to data — typically in the form of files — that was being jointly produced and used by collaborators in disparate locations.

In the interim these notions have further developed. They are still driving forces behind Grid, but they have been significantly extended to encompass more of IT than just computers and data. As I detail later, Grid enables a loosely-coupled, service-based IT environment. It is the broad spectrum of IT 'resources' that can be a 'service' that elevates Grid beyond just scientific computing. Importantly, Grids will have applicability in a larger cross-section of the IT world, specifically the enterprise. Enterprise is the commercial, large and medium business IT space.

In this context, HP is focused on delivering Grid technology to the enterprise to enable our customers to derive benefits in the form of reduced costs and complexity and to allow for doing things that could not be efficiently achieved before.

What is Grid?

Before going further, there is a need to define some terms related to Grid. This is especially important since so many different Grid definitions exist.

I offer the HP point of view on Grid. If you accept the perspective I offer here, either as a whole or in part, then you will grasp the strategic importance of what Grid, true Grid, can offer you. Grid can offer strategic importance whether you are a CIO, IT manager, IT administrator or an end user wanting, singularly or in collaboration, to solve problems.

HP is focused on the management and execution of Grid services. This sounds simple. Before you accept the seeming simplicity of this, I must further dissect the statement and explain a Grid service.

In the context of the Grid we at HP believe you can render virtually any IT resource in the form of a 'Grid service'. Think of this service as you do for a service you receive in the consumer sense; you have a need for it, you find it, you request or purchase it and finally it is provided to you.

All parts of what I think of as IT can be rendered as Grid services:

- **computer systems**
- **a quantity of computer cycles**
- **storage space**
- **a printer**
- **some printed pages**
- **an application**
- **a data file**
- **a database**
- **a set of records in a database**
- **and so on.**

Once these IT entities are in the form of a Grid service, then the Grid infrastructure itself will allow them to be:

- **registered**
- **discovered**
- **provisioned**
- **accessed**
- **shared**
- **removed**
- **managed**
- **monitored**
- **metered**
- **even billed for.**

These last sentences describe, albeit briefly, a powerful concept. There is, however, yet another aspect. Grid services and the supporting Grid technology enable the secure sharing and access to these services by members of a virtual organization. A virtual organization is one or all of the following: ephemeral, geographically distributed, in separate ownership or management domains, and has specific membership. The high-level view is, then, that Grid enables a loosely-coupled, service-based IT world.

Hewlett-Packard is focused, therefore, on the industrial-strength management and execution of Grid services in the scientific space and, as importantly, in the commercial enterprise.

Grid technology enablers

There are three important enablers that make Grid technologies viable. They are:

- **the global reach of the Internet**
- **abundant bandwidth between entities connected to the Internet**
- **the development of open standards for Web Services and Grid Services.**

The global reach of the Internet means that geographic proximity is no longer required to conduct commerce, science or business. The ability to access IT resources from anywhere, to anywhere, is today almost pervasive. This is a direct result of the growing use of IP (Internet Protocol) addressing being used on IT resources. Over the coming years it will be even more so as IP addressing will be applied to ever more things in our business and personal lives.

Abundant bandwidth between entities connected to the Internet permits the exchange of substantial amounts of information which could be in the form of data or applications. The available bandwidth is growing at significant rates, further 'shrinking' the distance between resources — wherever these may reside.

The development of open standards for Web Services and Grid Services embraces a broad set of definitions, at various stages of evolution and industry acceptance, which permit applications to communicate. The core Web Services specifications permit applications to discover each other and communicate over the Internet using standard languages and protocols such as XML, WSDL, SOAP and UDDI. The adoption of this core technology by the Grid community — and development of extensions such as OGSI (Open Grid Services Infrastructure), designed to meet Grid requirements — has been a key development over the past two years. Grid Services extend Web Services with a set of interfaces and behaviors defined by OGSI, in areas such as service state and transient service instances.

Grid business drivers

With the above enablers and the maturing state of Grid technologies I see these, in concert, as being directed toward solving several emerging mega-trends in IT:

- **the rate of accumulation of IT gear is significant: servers, storage, PCs, laptops, printers, switches, routers, PDAs, etc. are filling up data centers and offices — and with this volume comes complexity and difficult management and utilization issues**
- **centralization will be the bane of tomorrow: co-located resources managed by a single central authority will eventually, if not already, become unsustainable; multi-national companies are not centralized and corporate data centers are, of necessity, not all in one location**
- **the ownership of the content of science, commerce and consumers is dispersed, but not necessarily public — it resides in geographically removed locations, within separate ownership domains and separate management domains; however, if it is not accessible to collaborators, partners, customers and colleagues then it has little value**
- **geographic dispersion of virtual teams is essential, and it is happening today; very few of us only interact with colleagues in the office next door**
- **the Earth's rotation leads to a 'follow the sun' model; while it is sleep-time at one organization's data center it is prime-time at another and spreading computing loads across otherwise lightly loaded resources provides better RoI.**

Somewhat glibly, I can envision all roads — driven by the business enablers I mentioned earlier and aided by these technology enablers — leading to a distributed and dynamic resource sharing model that is loosely-coupled.

Why Grids?

If you are a CIO or IT manager in the first decade of the 21st Century you are probably being asked to do more with less. Global management of IT requires that CIOs be able to guarantee the security and reliability of the systems and support every aspect of a business. So this really places a new level of importance and focus on the IT function. IT must be able to operate simultaneously, in three dimensions — predictably, flexibly and reliably — to provide a significant return on IT.

The corporate IT function is now accountable for driving down costs, creating new value, making IT as a service more agile, secure, reliable, flexible and adaptable, to any sort of change. In today's business environment that is the only constant. Virtually all CIOs are faced with one thing: that is, they demand more from their IT infrastructure and from their IT partners.

What is needed, therefore, is a platform for managing change which links business and IT together in real time. HP's late CIO, Bob Napier — who successfully managed our own internal IT infrastructure through the largest merger in IT history — coined a phrase that we in HP think really captures the problem: "every business decision that you make triggers an IT event or a series of IT events."

One of the things that the Grid will allow you to do is tie the business architecture through service level agreements to the IT architecture. At that level it allows for the necessary operations to provide a flexible and agile architecture, on a global basis.

Proprietary architectures with vertically integrated IT stacks have become islands of automation today. They are prohibitively expensive to manage and maintain through any significant change. CIOs are specifically demanding a new enterprise architecture, one that is open and welcomes change and modularity so that it can be evolved in an incremental fashion. Grid technology can provide this.

Grid mythology

There is a certain amount of 'mythology' surrounding Grid. This is partially a result of misunderstanding, over-loading of terms, overlapping technologies and hype. When considering technology insertion and adoption in the enterprise none of these is acceptable.

Let me pick just a few of the more common misunderstandings and debunk them here.

Myth: Grid computing is just about scientific computing and does not fit commercial applications; ***Reality***: Grid computing has been driven by new and demanding applications from the worlds of technical and scientific computing. This is just like Internet and the Web used to be.

Myth: Grid is a cluster. ***Reality***: clusters are systems on a private low-latency LAN and designed for time-to-solution. Grid can exist on the public Internet and are generally not a time-to-solution application.

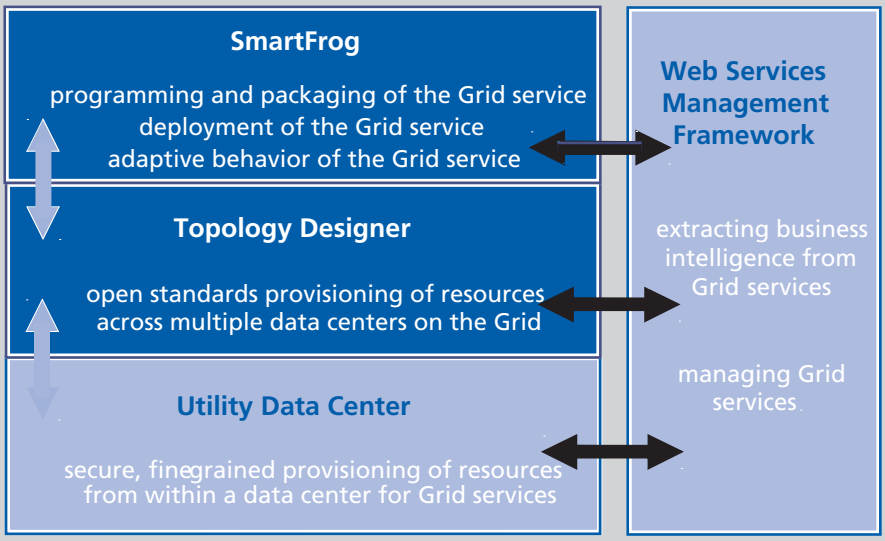
Myth: Grid computing is restricted to a particular class of applications, those which are 'embarrassingly parallel'. ***Reality***: Grid computing is about running application workloads on a pool of shared resources. Scientific workloads which are easily parallelized are a particularly good fit to this model. Research and development into commercial grids is being driven by the desire to run commercial workloads across a pool of shared resources, either in an enterprise Grid or managed utility service provision model.

Myth: Grid computing will never work because of {security, resource management ... insert your favorite technical issue here}. ***Reality***: Grids tend to be deployed today in scenarios where these issues are easy to address, typically within a single administrative domain. Innovation in {security, resource management etc}, especially for multi-domain Grids, is driving the efforts of system vendors and growing numbers of independent software vendors and start-ups.

Myth: ***Grid*** computing will never work because of {licensing, server hugging, charging ... insert your favorite business or political issue here}. Reality: the drive towards the Grid/utility/service-centric/ adaptive model is unstoppable. Over time software, storage, server and IT service businesses will change to adopt this model-driven by the demands of customers.

It is misleading that some people label the Grid, by itself, as a product. The reference implementation of the Grid, the

Figure 2.1: HP contributions to making Grids work



Globus Toolkit, is just that: a toolkit. There needs to be infrastructure to support it — (say) servers, instruments, storage, etc. At the highest level an application needs to invoke the Grid middleware to obtain access to, or enable sharing of, the resources in the form of Grid Services. This may be in the form of a portal to more easily connect a 'user' to a Grid. However, Grid middleware alone is not a complete solution.

When customers are ready to evaluate Grid as a technology for their environment or their problems, it is often valuable to do an evaluation of deploying Grid as a solution. This can involve an analysis of the situation and the problem to be addressed to simply determine if Grid is in fact the right answer. If Grid technology is appropriate then it still requires:

- **a scoping of the effort**
- **actual deployment**
- **the enablement of the environment.**

It may even extend to maintaining the Grid. Nevertheless, the net result is that energy needs to be applied in order to derive value from Grid.

Management and execution of Grid services

When a commercial enterprise considers technology adoption, its requirements are more stringent than in the scientific space. Knowing this, HP is focused on the management and the execution of Grid Services. This is the barrier to entry for enterprise customers.

The ability to manage — including monitoring, metering and billing — the IT resources rendered as Grid Services is crucial. If there were to be no ability to manage these services, the enterprise would be powerless to understand or control the IT environment. Without control there is risk of inoperability or failures which would in turn affect the business. Additionally, as I have mentioned previously, the growing amount of deployed IT equipment necessitates a robust management solution.

In the commercial space Grid Services are expected to be more complex than the simpler compute-intensive batch processing common in the scientific arena. There will be multi-application services with SLAs and possibly dependencies. Being able to robustly and simply 'execute' these types of services is necessary for enterprise customers. As a rendered IT resource is provisioned to a requestor, there still remain technologically challenging tasks to ensure successful delivery of each service and its results.

For the above reasons HP is focused on the management and execution of Grid Services. HP, I believe, has developed several technologies that address these needs.

Challenges and requirements for commercial Grids

To fulfill the promise of Grid we, as a vendor and as an industry, have much work to do. This is because hard problems exist and solutions to them are required. Here I highlight a few of the difficult issues and what, as an example, HP is doing about them.

The hard problems that need to be addressed for commercial enterprise use are:

- **fully open standards**
- **heterogeneity**
- **complexity**
- **robustness**
- **trust and security.**

The only way that Grid will be able to provide the universality it offers is if it is built on open standards. Another part of the universal applicability is to ensure it can interoperate completely in heterogeneous IT environments. In this way, all IT resources — regardless of manufacturer — can participate and be rendered in an enterprise Grid.

The complexity of the deployed Grid solutions in the enterprise will require a robust management solution to reduce and mask this complexity — making the most efficient use of the enterprise CIO or IT manager's time and resources. The body of code for the Grid middleware must be robust and rigorously tested. For example, software best practices need to be applied to significantly limit risk and downtime. Furthermore, the establishment of trusted and secure solutions is a baseline necessity for Grids to be deployed beyond the four-walls of a single enterprise.

Figure 2.I illustrates four technologies — UDC, Topology Designer, SmartFrog and WSMF — that are inter-related and which address some of the hard problems in making Grid suitable for the enterprise.

UDC: The Utility Data Center is a wire-once, dynamic, programmatically configurable, data center with fine-grained resource allocation and hardware security from resources in a shared, virtualized resource pool. The UDC lets organizations allocate and reallocate resources on the fly, transforming the data centre from a static repository of applications and data into a dynamic computer-power generating facility. This HP product has value to the enterprise

but I suggest that the marriage of UDC with Grid is particularly powerful. If a UDC is presented with a request for a Grid Service that it cannot currently meet, it could programmatically reconfigure to meet the request. For UDC and Grid, the whole is greater than the sum of the parts.

In Topology Designer, HP Labs has developed a software interface to enable the design of Grids or UDCs or data centers, via an open, easy to use GUI. This reduces the complexity of defining and managing Grids.

In SmartFrog, HP Labs. has developed a software technology to enable the execution of complex service 'packages' when they are invoked as a Grid Service. For example, consider a rendering application that has SLAs, version requirements, input data coherency requirements, etc. SmartFrog has a descriptive language and agents to reliably deploy and execute such Grid Services.

WSMF is HP's Web Service Management Framework. It was designed to manage Web Services using Web Services. It is being refactored to do likewise for Grid Services. Importantly WSMF allows for management, monitoring, control, metering and event handling for infrastructure, applications, Grid Services and business practices. As an open, heterogeneous standard, WSMF has been submitted to the WSDM working group of the OASIS body. WSMF has been used to allow operational (IT) and business (sales and revenue) views of such Services.

The future

Hewlett-Packard believes that Grid technology is a significant technology that will permeate the IT industry and affect how science, business and e-commerce are performed. We are all at the beginning of the path to this. But it is not going to occur as rapidly as some predict.

This is the normal evolutionary route that new, and complex, technologies follow. Technologies that are as broad as Grid always evolve in an organic way.

HP, as the world's largest consumer IT company, also has an interest and intent to push Grid technologies into the arena of printers, appliances, PDAs, etc. Reflection on the points I made above — about what Grid can provide to the enterprise — map to the consumer and appliance space. For example, think of a printer as a Grid Service that can be discovered and provisioned by a Grid user. The fulfillment of this is further in the future but it yields a unifying structure to all of IT.

The call to action is to focus on open standards and industrial strength implementations of Grid technologies such that the commercial enterprise is enabled to utilize the technology. This is not a statement that the scientific space is not important. On the contrary it is in the scientific arena that the ideas are formed and the early deployments are tested.

History may well show that Grid is the next wave of IT technology.

Management conclusion

Mr. Robison, as the CTO of Hewlett-Packard, is in an excellent position to assess the relevance and practicality of Grids. He has customers to satisfy, both externally as well as internally. The concept of virtualized computing which combines all sizes of device is both broad and deep.

HP, as a vendor of Grid solutions, has started to package up key technologies for the commercial world. This matters. It is probably the most significant of Grid progress. Not only is HP investing but its customers are being offered products and solutions to buy. This takes Grids out of the 'wholly do-it-yourself' category and into the familiar world of IT.

The inherent contradictions in Service Oriented Architectures

Tom Welsh
Consultant

Management introduction

Just as we were coming to terms with Web Services, the IT industry has started talking up the notion of a Service Oriented Architecture (SOA). There seems to be little agreement about exactly what an SOA is, but that has not stopped analysts and vendor representatives — although, not necessarily, customers — enthusing about the potential benefits.

Yet, until the supporters of SOA put their heads together and agree on a reasonably firm definition, their claims are bound to be marred by inconsistency. In this analysis, Tom Welsh:

- ***examines some of the most glaring contradictions in recent pronouncements***
- ***considers the likely organizational difficulties that lie in the path of anyone seeking to deploy a pure SOA (whatever that might be).***

Service Oriented Architecture

Allegedly the market for 'SOAs' will be worth $4.5 billion by 2005 and $43 billion by 2010, according to ZapThink, an analyst firm that specializes in Web Services and SOAs. But what exactly is SOA? Considering how much has been said and written on the subject in the last two or three years, it is remarkable how far away the industry seems to be from reaching any reasonably coherent consensus on this question.

Reduced to their essentials, most definitions of SOA tend to fall into a few categories:

- **'SOA means more or less the same as Web Services'**
- **'an SOA is a generalized and abstract version of Web Services'**
- **'an SOA is any distributed system that involves the invocation of remote services'**
- **'an SOA is an application that models business activities as services'.**

Obviously, these definitions overlap substantially but they are by no means equivalent. For instance, the first severely limits the scope of SOA, but implies that it is available today. In contrast, the fourth leaves most technical questions unanswered.

Although the term 'service-oriented' can be traced back to the 1980s, SOAs only began to be talked about after the introduction of Web Services in 2000. In his book Web Services: ***Building Blocks for Distributed Systems***, Graham Glass even defines the acronym SOAP to mean 'Service-Oriented Architecture Protocol' rather than the official 'Simple Object Access Protocol' (even though this is no longer simple, either). As founder and CEO of The Mind Electric (since acquired by webMethods), Glass is one of the world's leading experts on both Web Services and SOA.

Recently, vendors and others have begun talking about SOAs, rather than Web Services, for a variety of reasons. Some do not wish to be restricted by what they consider an over-narrow definition of Web services. IBM, for example, usually says that WSDL is the critical standard for SOA, and that both HTTP and SOAP itself might some day be replaced to advantage. Others, such as Iona Technologies, wish to take advantage of their track record in building distributed systems with CORBA and other forms of middleware. It is in their interest to define SOA broadly enough to embrace CORBA along with Web Services.

In addition to these technical interpretations, there is a radical view which holds that the distinctive characteristic of SOA is its emphasis on business activities rather than on technology. This is calculated to appeal to non-technical decision makers, by:

- **promising to give them direct control of operations**
- **hiding the gory details of software development.**

But, since the whole concept of 'services' in the context of SOA is purely technical, it is doubtful whether such optimism can be delivered in full.

SOA, Web Services and Mr. Dodgson

A thorough inspection of the literature on SOA and Web Services reveals a pervasive — if consistent — woolliness and vagueness. Few, if any, statements are actually untrue or self-contradictory.

But, after a while, it becomes clear that some of the things being said are mutually contradictory. In other words, while individual comments may be consistent, the views expressed by advocates of SOA as a group contain some glaring inconsistencies.

While reviewing some propositions about SOA that I found unsatisfactory, it struck me that I had seen something similar before. After some reflection, I made the connection when I realized that I was thinking about Lewis Carroll's sorites. (Although best known as the author of Alice in Wonderland and Through the Looking Glass, Carroll — whose real name was Charles Dodgson — was an Oxford mathematician and logician, as well as being an author of accomplishment.)

A sorite is "a form of argument in which a series of incomplete syllogisms is so arranged that the predicate of each premise forms the subject of the next until the subject of the first is joined with the predicate of the last in the conclusion". Here is an example of this type of puzzle, from Carroll's book Symbolic Logic:

- **all babies are illogical**
- **nobody who is despised can manage a crocodile**
- **illogical persons are despised.**

From these three propositions, it is easy enough to work out that 'no baby can manage a crocodile'. Carroll explained how the process of inference could be formalized, by breaking down each proposition into the simplest possible statements, thus:

- **B: it is a baby**
- **L: it is illogical**
- **M: it can manage a crocodile**
- **D: it is despised.**

These can then be manipulated according to the rules of logic, to produce the conclusion 'if it can manage a crocodile then it is not a baby'. This is equivalent to 'no baby can manage a crocodile'.

Most sorites with more than four or five propositions are quite difficult to work out in one's head; they demand recourse to pen and paper. For instance:

- **all the dated letters in this room are written on blue paper**
- **none of them are in black ink, except those that are written in the third person**
- **I have not filed any of them that I can read**
- **none of them, that are written on one sheet, are undated**
- **all of them, that are not crossed, are in black ink**
- **all of them, written by Brown, begin with 'Dear Sir'**
- **all of them, written on blue paper, are filed**
- **none of them, written on more than one sheet, are crossed**
- **none of them, that begin with 'Dear Sir' are written in the third person.**

(To avoid driving anyone to distraction, perhaps I should reveal that the answer to this one is 'I cannot read any of Brown's letters'.)

Sorites and SOAs

What does all this have to do with SOA? Consider the following set of statements:

- **Web Services are so called because they use open Web protocols**
- **open Web protocols allow any two computers to intercommunicate, without having to agree on a single common middleware product**
- **HTTP is an open Web protocol**
- **Web Services use either SOAP or XML-RPC as their application protocol**
- **none of the other specifications built on top of SOAP work with XML-RPC**
- **SOAP may run on top of HTTP, but may also run on top of other protocols such as JMS or SMTP**
- **to be effective in an SOA, Web Services must be fast, secure and reliable**
- **JMS is faster, more secure and more reliable than HTTP**
- **JMS does not allow any two computers to intercommunicate, without having previously agreed upon a single common middleware product.**

Note that all nine statements are plausible ones — and, in fact, all of them represent views that have been expressed by one or more influential supporters of SOA. It is only when they are laid out together in this way that their incompatibility becomes noticeable.

One of the most commonly stated arguments in favor of Web Services is that, because they use 'open Web protocols', they can easily be used to link any two computers. This convenience is contrasted with other middleware such as CORBA or RMI, which has to be present at both ends. (Actually, SOAP has to be supported at both ends too, but let that pass for now.)

On the other hand, vendors like IBM often point out that SOAP over HTTP is not fast enough, reliable enough or secure enough to serve as a basis for enterprise SOA. It suggests using JMS or some other message oriented middleware (MOM) product, such as MQSeries, instead.

And this is where the inconsistency arises. JMS is nothing more than a definition of a standard interface. It cannot necessarily link two computers unless they both have the same proprietary MOM product installed.

Also, the only real justification for the name 'Web Services' is the use of SOAP over HTTP (as HTTP is 'the' Web protocol). Arguably, substituting JMS for HTTP means there is no longer any reason to call the result Web Services.

Adding to the sorite confusion

Here is another, shorter example:

- **Web Services make use of dynamic discovery**
- **any business-critical SOA must be extremely secure**
- **UDDI is the default way of providing Web Services with dynamic discovery**
- **UDDI is not particularly secure.**

One of the distinguishing features of SOA is supposed to be dynamic discovery — the ability securely to look up

services based on what they do or where they are hosted. The standard way of doing this is through UDDI. But it turns out that UDDI is not very widely used. One of the reasons is that it lacks adequate provision for security.

Other pseudo-sorites can readily be constructed. I hope, however, this is not necessary. It should now be clear that many of the beliefs and assumptions cited in favor of SOA are mutually incompatible.

That does not mean that an SOA will not work, nor that it is a bad idea. But it does highlight the need for caution when evaluating vendor presentations and their premature enthusiasm.

The organizational implications of an SOA

As we have seen, there is a lot of uncertainty as to what SOA means. There is just as much confusion about how one should be implemented.

It so happens that this lack of clarity is mostly confined to the technical domain. Everyone seems to agree that an SOA is beneficial to a business, because it focuses on 'business activities'.

The following explanation is a good example of this point of view. It was given by Michael S Pallos, a senior solution architect for Candle in an article entitled ***Service-Oriented Architecture: A Primer*** that appeared in the December 2001 issue of the EAI Journal.

"SOA is the aggregation of components satisfying a business driver. It comprises components, services, and processes. Components are binaries that do specific tasks. These binaries each have a defined interface and usually one job (e.g., 'validate user' or 'obtain credit rating') to do well. A service is simply a grouping of components (executable programs) to get the job done. Example: 'process loan application'.

"The key focal point of an SOA is the business process. The grouping of components satisfies the process, letting the application process pattern more closely represent the business. This higher level of application development provides a strategic advantage, facilitating more focus on the business requirement. The business driver is satisfying the requirements, as opposed to the technical mechanism of application development. A service is built using components. These components are executable programs. So development and ownership costs, as well as implementation risk, are reduced."

The thrust of Pallos' argument is clear: an SOA opens up applications, splitting them into individually accessible services, each of which performs a distinct 'business activity'. Moreover, it is implied that these services can be quickly and easily 'glued together' to create applications that do exactly what their business sponsors want — in sharp contrast to the 'bad old days' when IT departments collected requirements, before taking months or years to deliver applications that then — with the passage of time — turned out to be unsuitable.

If this really were possible, it would no doubt be advantageous for all concerned. But leaving aside the question of technical feasibility, there are serious organizational issues to consider.

In the first place, there would have to be a complete transformation of the way in which IT projects are carried out. In current practice, software is written by programmers who need to work together in a co-ordinated way for weeks or months. This requires that they work to some kind of plan, whether a formal design model or simply a set of ideas that they have talked over. The overhead of maintaining a development team is large enough that these are mostly concentrated within an IT department.

The classic way of getting a new application written is for one or more sponsors to approach the IT department with a wish list and some funding. The developers assess the desired functionality, perhaps gather more detailed requirements and — a critical step — decide how the new application will fit into the existing corporate infrastructure. Then the new software is written, possibly within the framework of some methodology ranging from eXtreme Programming (XP) to the Rational Unified Process (RUP). Tests are run (usually including a period of user 'acceptance') and eventually the new application is handed over.

Building an SOA

Now imagine what would happen in 'the Organization' that has decided to build its business critical applications along the lines of Pallos' ideas. Some major differences come to mind.

In the SOA scheme of things, services are self-contained units of business activity such as 'process loan application' (Pallos' example). These are designed in such a way that each service carries out a single, well-defined task and returns a crisp answer. Furthermore, the messy details of how the task is carried out are hidden, not only from the business user but also from any developers who include the service in the applications that they build.

Because each service encapsulates certain business processes, it can only be created by people who fully understand how those processes work. In other words, all the services that deliver accounting functionality must be written by (or with the help of) the accounting department. So how will the requirements for these services be determined? Perhaps 'someone' in the accounting department is going to:

- **enumerate all the business activities that this department ever engages in**
- **document exactly how they should be done, and are done (not necessarily the same)**
- **write the necessary code.**

Delivering against this scenario is impossible, though. The accounting department already has mounds of software which:

- **have already cost a huge amount of time, money and effort to write**
- **are known to work (well enough, at least).**

There is simply no question of re-writing everything: the cost would be prohibitive (and the RoI doubtful). Never mind: the conventional solution to this dilemma is to implement the necessary services as a series of 'wrappers' — mere interfaces to the existing business logic. (There are several technical problems with delivering via this approach but, because we are taking an organizational viewpoint, we will overlook such difficulties.)

Now we come to the question of determining the complete set of requirements for the accounting department's services. If they are to be used (or 'consumed', as the SOA jargon has it) by the accounting department only, there would be no point in creating them in the first place. The whole point of an SOA is to make each department's functions available to everyone else in the organization who might need to use them.

Implicitly it becomes necessary to canvass all potential users of the accounting department's services, both inside and outside the department, before finalizing any list of required services. If the list must be complete by a given date, this will entail actively gathering requirements from the 'whole organization' — a substantial task. Alternatively, it might be deemed more practical to provide a channel through which requirements can be submitted, and accept fresh requirements at any time. In that case, however, the accounting department must be prepared to maintain its own 'just in time' services development team, ready at any moment to spring into action and provide new or modified services.

SOA and maintenance

There is more. (In the world of software, there is always more, except time or money.) Programmers have a rule of thumb that about two-thirds of the lifetime cost of any application stems from maintenance — including enhancements and modifications as well as bug-fixing. This can rise to 95% or more in some cases, but is rarely much less than half. In other words, once the accounting department gets its services up and running, it can expect to spend still more time and money keeping them up to date and otherwise acceptable to its broader user community.

Every time the accounting department wants to make changes to its own applications, for example, it will have to agree them with all the external users — a process that could soak up a lot of time and money. That can be avoided by the technical expedient of never changing an existing service, but adding a new one whenever different behavior is required. On the other hand, the number of services will likely grow alarmingly in such a scenario.

Since the cost of making changes throughout an SOA could become unacceptably high it is, therefore, vital to make sure that the initial deployment possesses no features that might have to be altered later on. To take just one example: it is coming to be accepted that the best model for Web Services, within an SOA, is to use SOAP in Document/Literal style. That is to say, Web Services communicate by sending XML documents, which the recipient must interpret as best it can.

However, most Web Services today use the contrasting RPC/Encoded style. If an organization were to roll out an SOA now, using RPC/Encoded, all its Web Services might have to be entirely re-written in Document/Literal style at some time in the future.

Distributing the IT Department

The consequences of splitting up the IT department's overall responsibility for corporate applications, and farming them out to each individual department, could also be far-reaching. If each department's software was only to be used within its own confines, the potential for harm would be limited.

It is the prospect of multiple inter-dependencies between services that are written and maintained by wholly independent groups that causes the real trouble. Yet, does this

not strike at the very heart of what is most attractive about Web Services?

Finally, whatever may be said about 'information hiding', the fact remains that most programmers like to consult the source code of all software that they use — especially if they did not write it themselves. In theory, all they need to know about is the interface to a service, which tells them all they require.

But this is one of those cases where theory and practice are poles apart. Unless each interface is supplemented by reams of precise documentation — which is, moreover, assiduously kept up-to-date — it is only too likely that misunderstandings will arise, possibly leading to applications delivering the wrong results. Oh dear.

Management conclusion

There is much enthusiasm for SOAs and Web Services at present — in the vendor community. Interestingly, some modest research does not reveal an equivalent interest coming from users or even IT. The difficulty is that although the conceptual foundations for SOAs and Web Services do contain much that is promising, they are not fully baked.

For example, the many differing arguments put forward by SOA advocates are interesting and provocative, but cannot always be taken at face value. There are at least two types of objection, which Mr. Welsh has encapsulated in this analysis:

- ***first, many benefits are claimed for SOA, some of which are actually incompatible with each other; while this is not necessarily a bad thing, it is too early to start ruling out features that have not yet been given a thorough trial in the field***
- ***second, even if one overlooks the issue of clashing technical features, SOAs are likely to pose some challenging organizational problems; while there is something to be said for handing off responsibility for the relevant services to each department in an enterprise, the costs and commitments and potential confusion of such a decision should not be underestimated.***

Decision makers should be aware that, for instance, Web Services may offer:

- ***complete openness***
- ***or high levels of reliability***
- ***but not (as yet) both at the same time.***

All in all, the SOA concept may well be the future of middleware. But much more work remains to be done before implementing it is quick, easy, cheap or risk-free.

Where are we going with Grid computing?

Mark Lillycrop
Principal Analyst
Arcati

Management introduction

Grid middleware technology has reached a difficult stage in its development:

- ***as a concept, it is now reasonably well understood by the IT industry***
- ***as the basis for commercial products and services from vendors, it is beginning to mature***
- ***but it is losing the hype and excitement that usually accompanies 'the next big thing'***
- ***as a solution for specific technical problems, it is quietly disappearing from view — as it becomes more accepted as part of the IT infrastructure.***

Now is probably as good a time as any to consider Grid computing from the perspective of the various interested parties:

- ***the technology providers***
- ***users***
- ***potential users.***

In this analysis, Mark Lillycrop asks:

- ***where is it (the Grid) now?***
- ***where is it likely to go?***

Grid computing in the numerically intensive world

Few people can have escaped the impact of Grid on the scientific world. From the earliest space projects like SETI (Search for Extra Terrestrial Intelligence), collaborative scientific work has driven the development of Grids.

SETI@home itself was the project that defined Grid as a potentially limitless mass of computer power, harnessing vast quantities of spare PC cycles through its now-famous screensaver. Some three million people have now downloaded the screen-saver, lending 600,000+ CPU years to the SETI effort.

The sheer size of this combined effort demonstrated what could be done by sharing and collaborating in the provision of processing. SETI paved the way for other numerically intensive applications, such as protein folding or analysis of the human genome.

Of course, not all scientific projects stir the imagination in the way that SETI did, or take collaborative computing down to the desktops of Web-connected enthusiasts. Many of the pioneer applications have relied on:

- **the free flow of information between academic bodies**
- **the willingness of universities to pool their data storage facilities plus their willingness to accept loose coupling of processors between multiple data center environments.**

Academic institutions are now becoming extremely adept at extending the principles of Grid to new application areas. Close collaboration over research projects is becoming increasingly common at the IT level.

Impact

Grid computing has probably had its greatest impact to date in medical analysis, automotive design, seismic exploration and other areas where large amounts of raw compute resource are needed on a spasmodic basis. Medical work shares many of the principles and aspirations of academia, with genuine Grids crossing boundaries between institutions and even countries.

Design and exploration work tends, by its commercial nature, to be a less public concern, with large organizations developing informal internal Grids to focus large-scale processing resources on their large-scale projects — when these are needed. The scientific world is likely, therefore, to remain the principle driving force for Grid computing for some time yet.

In contrast, overtly commercial systems which use Grid middleware face several sizeable obstacles to overcome before they can freely exploit Grid-type functions. This contrasts with numerically-intensive applications which are often only are limited by the technical expertise available to make them happen.

In practical terms, much of this (Grid middleware implementation) expertise is currently coming from vendors. They are understandably keen to demonstrate the potential of Grids to both the scientific and business communities alike (q.v Shane Robison of HP, on page 10). In particular, they are trying to move:

- **away from the more esoteric, home-grown applications**
- **towards packaged Grid offerings that can yield rapid economies and reduced development cycles.**

For example, IBM has focused much of its attention on the aerospace and automotive sectors to date. It has recently announced the results of a Grid project for the Canadian automotive supplier MAGNA STEYR. In collaboration with Grid software specialist Platform Computing, MAGNA used a Grid-enabled version of the CATIA design package to reduce its clash testing (the testing of component combinations within a vehicle subsystem) from 72 hours to 4 hours. IBM cites other companies, such as the French Petroleum Institute and Japanese electronics manufacturer OMRON, which have similar plans underway to help speed product development.

There is, if this is evidence, a huge potential for Grid applications within the manufacturing design sector. IDC estimates that the market for Grid-enabled manufacturing systems will hit $2.6B by 2006, with the Grid business as a whole rising to $13B by 2007. However, since companies rely on new product design for their competitive edge, this kind of application necessarily involves a much more insular approach to Grid exploitation than the high-profile, shared scientific projects discussed earlier.

As with many key technological developments — the Internet, for example — what started as a largely altruistic attempt to share and optimize resources on an egalitarian basis is gradually being swayed by profit considerations. Numerically intensive Grid applications are likely to develop in two largely parallel directions in the future:

- **those driven by academic, medical and public interest research needs, which will typically harness compute resources across the Web and intranets**
- **those offering commercial value; these will be more tightly controlled and more expensive.**

Data-intensive Grids and the commercial environment

This leads us to what has long been perceived to be 'the crock of gold' for suppliers in the Grid software arena — the commercial world. While Grid computing enables us to tackle scientific projects that would otherwise simply take too long to complete, the longer-term attraction — it is argued — will come from the cost savings and resource utilization improvements that can be achieved across the broad business community.

Ultimately, in this vision, we will see transactional Grids appearing, with a Grid-style infrastructure supporting a range of Web-facing commercial applications that range from financial, retail and other service-oriented companies. However, as has been pointed out in previous **MIDDLEWARESPECTRA** analyses, mainstream Grid transactions are not yet ready for the commercial world. Furthermore, security management is still rudimentary, plus there is a general reluctance among many commercial organizations to take the leap into Grids until they are convinced that there are sufficient realizable benefits.

The 'foot in the door' for suppliers of Grid solutions until now, therefore, has been 'number-crunching' applications. But, increasingly, Grid techniques are being applied to data analysis within those same organizations — to what is known as 'business analytics'.

This includes areas such as data mining and corporate risk analysis, which have taken on increased significance in recent months as organizations battle to make sense of the vast amounts of customer data stored within their systems. Data mining, for example, is no longer just about identifying market trends. Increasingly it is about:

- **compliance with regulations**
- **understanding individual customers and customer behavior**
- **ensuring that the business in not placing itself at unacceptable levels of commercial risk.**

In view of the amount of data involved, and the speed with which companies are expected to comply with changing regulation, Grid computing has many distinct attractions in this area. Moreover, many companies have found that traditional data mining applications are not efficient consumers of IT resources. Too many data analysis projects are one-offs and would benefit from having available flexible computing resources at specific times. In many ways, this is a small step from the more scientific applications discussed earlier, although work is still going on in bodies such as Globus and the Parallel Computing Center, and in specialist companies such as Avaki, to develop standards and common techniques for exploiting data Grids.

At the same time there is, however, a growing number of case-studies which illustrate the ways that Grid computing is being applied to financial analysis tasks. In the February 2004 issue of the **MIDDLEWARESPECTRA**, the case of RBC Insurance of Canada was cited, which reduced a 2.5-hour commercial analysis job to 10 minutes and an 18-hour task to 32 minutes.

Other large companies in the financial sector are taking notice. They are investigating whether Grid techniques, when combined with Web Services, offer a way to harness their distributed IT resources and allow large-scale calculations and simulations to be performed on the fly.

In January of this year, Charles Schwab — so often at the forefront of new IT developments — announced details of its Grid-based retirement plan advice service, a joint project with IBM. The application is highly complex in nature: it is intended to enable advisors to use highly sophisticated modeling tools to calculate the potential return from various investment scenarios — in front of the customer.

In view of the scale of the calculations involved, the system would have been impractically slow — or expensive — unless Schwab was able to optimize the utilization of resources by using Grid techniques. What Schwab has done is to spread the compute function over 12 machines (IBM Intel-based xSeries 330s, running Red Hat Linux and DB2). This doubled the peak workload that this one application could handle, accelerating the simulation processing by a factor of at least 10.

Other companies see the real value of Grid computing in its ability to bring the technology together with Web Services. At a US financial conference in February, JP Morgan and UBS talked about their small-scale implementations of Grid-based Web Service applications. They see a role for Grid in supporting an 'adaptive environment', where resources can be applied on demand and application failures can be automatically re-routed. But this environment is still experimental: the partners in this project continue to look for ways to make the solution scale up.

Grids and Web Services

In the commercial world, there is now a widespread desire that Grid and Web services should converge in the not-too-distant future. To this end, the Open Grid Services Architecture (OGSA) has been formed under the auspices of the Global Grid Forum — with a view to bringing the two concepts together in a consistent, standards-based way.

The OGSA philosophy is basically that, whatever vendors claim about the potential of Grid and Web Services, they are both pursuing similar, compatible goals. The problem is that, while the Grid is more mature, it is clearly oriented towards scientific and numerically-intensive applications, whereas nascent Web Services have an inherently more commercial bias.

As a result, OGSA is bringing together the best of both environments, such as:

- **Grid Resource Allocation and Management (GRAM)**
- **the Meta Directory Service (MDS)**
- **the Grid Security Infrastructure (GSI)**
- **the Simple Object Access Protocol (SOAP)**
- **Web Services Description Language (WSDL)**
- **Universal Description, Discovery and Integration (UDDI).**

The first three of these come from the Globus/Grid community while the last three originated in the Web Services world. At the same time, standards are being extended to support 'persistent services' — as required by transactional business applications. This is a step beyond the 'transient service instances' of the scientific environment. That said, there is still some considerable debate about the way that .NET and J2EE will fit into this scenario.

Grid in the wider world

As discussed earlier, the formalized convergence of Grid and Web Services is already of considerable interest to large companies like Charles Schwab and JP Morgan Chase. But progress in this area is also fundamental to the growth of Grid computing beyond large enterprises.

Grids, thus far, have been a mixed blessing for Grid IT vendors. The biggest market has been in large accounts, and here the interest has been driven by a need to save money. If Grids offer a way to streamline IT operations, to economize on storage or to delay a processor upgrade through improved utilization, that:

- **is good news for the customer**
- **but does not help the vendor in the short or medium term.**

Where IT vendors want Grid technology to have a real impact is as a basis for utility computing among small and medium-sized companies. The reason is that this is where it (Grid computing) is expected to offer a real chance of expanding the overall IT market (not least by hiding the middleware dimension from those small and medium size businesses), as opposed to helping existing large users to consolidate resources and save money.

Utility computing in its broadest sense is still a ways off. The idea that companies will be able to access IT resources with the same ease and control that they currently obtain heat and water is attractive, although not easy to achieve.

But there has been substantial progress over the last couple of years. What we still lack are the tools for metering, measuring and monitoring IT resources on a truly ad hoc basis. But every month we come closer to this goal.

This is certainly where the services vendors are concentrating their efforts. Grid technology enables the creation of virtual server farms — boosted in time, no doubt, by surplus processing power and storage resources on the desktop. This is a highly cost-effective way of gathering power that can then be distributed on a utility basis to customers.

For users, utility computing offers many benefits beyond simple cost. It permits organizations to:

- **rid themselves of onerous IT management, infrastructure and support tasks**
- **obtain a much clearer picture of the value that IT contributes to the business as a whole, and to specific business tasks in particular.**

Utility computing, therefore, holds out the promise of allowing smaller businesses to be much more confident in their use and deployment of IT. Many observers expect to see some killer utility applications appear soon.

However, once again, IT service companies need to be sure that Grid-based utility services do not undermine traditional IT service sales. The slow arrival of utility computing is as much to do with marketing expedience as with any technical limitations.

The shape of Grids to come

The more you look at the development of Grid technology, the more you realize how diverse the Grid — and middle-

ware — marketplace really is. Technically, we seem to be moving towards convergence. Bodies such as Globus and the OGSA initiative are providing a vendor-neutral platform for managing the development of standards, particularly in the tricky areas of security and management that will guarantee a future for Grid in the commercial world.

The real differences are at the cultural level, where there are conflicting power bases within the Grid world that have very different styles of working. In this respect, the parallel with the Internet is apt: the 'net thrived for decades as a means of communication between the academic and military communities. The World Wide Web opened it up to commercial exploitation. But even then there was a lengthy period of wrangling and soul-searching while the interested parties argued about the viability of 'doing business' over the Internet.

Grid computing is following a similar path, in that the standards developed to support the sharing of resources between participating universities (and even between desktops, in the case of SETI) are now being tuned and adapted to meet commercial criteria and expectations. What makes the Grid different, however, is that scientific and commercial users will probably choose to diverge in the future:

- **in academia, and in the medical world, openly sharing IT resources is not only acceptable, it is a positive sign of co-operation; as such, 'public' Grid projects will favor a less restricted form of Grid deployment**
- **in contrast, commercial users will likely seek to confine the use of Grid to trusted parties, and to add further controls for managing security and service levels.**

Furthermore, there are a number of unknown factors that will affect the development of Grid computing in the future. Like all de jure standards, it is unlikely to remain open for ever. Vendors will want to 'add value', and to shape Grid computing in the way that best suits their particular strategy.

While OGSA may offer a meeting place for Grid and Web Services, the latter are already the subject of competition and political in-fighting within the industry. Depending on the relative strength of support for .NET, J2EE and other emerging standards, Web Services might have a very different look in five years' time. In turn this will affect the way that Grids are implemented.

Management conclusion

Depending on your point of view, the real value of Grid technology is that it:

- ***supports and encourages the pooling of IT resources between interested parties***
- ***offers a new order of magnitude of effectiveness in IT resource utilization***
- ***promises to open up new markets for IT that were beyond the horizon of traditional IT.***

As we move forward, however, it will become increasingly clear that 'Grid middleware' is basically just an enabling technology. Its real value will be delivered through the business applications that are layered on top, ones which take advantage of Grid's inherent flexibility to provide a basis for utility-based computing.

The impact of standards: EAI, ESB and Grid computing

Tony Leigh
Director of Operational Delivery
Sensima Research

Management introduction

In this analysis, Tony Leigh of Sensima (based in Salisbury, England — www.sensima.com) examines the evolution of real-time application and data integration. In particular he looks at:

- ***market priorities and outlook***
- ***the CIO's agenda***
- ***the case for EAI, ESB and next generation distributed databases and application servers***
- ***how new — and existing — business savvy technology companies are changing the nature of enterprise applications***
- ***how Grid computing is coming to commerce (with examples from Oracle and Sybase)***

The result is, he argues, delivery at last on many of the long promised benefits of integration convergence.

Market priorities and outlook

When E*Trade's Architecture Consortium Exchange (ACE) group wanted to match its growing investor-led network demand by using its existing network infrastructure, it was initially faced with making an investment that would have cost tens of millions of dollars (according to various case studies published on the issue). Instead, it adopted an alternative approach by which it was able to:

- **offset the investment**
- **cut infrastructure costs**
- **deal with peak traffic performance**
- **deliver content faster.**

It achieved this by introducing a new breed of real-time infrastructure software. For a fraction of the alternate investment (E*Trade has estimated, with the benefit of hindsight, a $5 million reduction in potential capital expenditure and a 97% reduction in back-end database traffic), it obtained a solution through the adoption of SpiritSoft's real-time, JMS- enabled SpiritCache software.

Performance and efficiency are the measures most relevant to executives and shareholders today. With performance and efficiency delivery comes the ability to enhance shareholder value. Nothing (or little) else is important. Put another way, every organization in every business sector is, or should be, obsessed with these two factors.

That said, financial market volatility, coupled with economic uncertainty, has resulted in unnecessary investment inertia in many organizations. The will to invest as well as to experiment has taken a hard since the hey days of the dot.com bubble.

IT investment has been particularly hard-hit. Many massive investments in middleware infrastructure and business applications, made in those 'good times', are perceived to have failed to pay off. As a result, CFOs and CIOs continue to run scared of instituting new major IT investments. Instead most seem to prefer to entrust their problem to outsiders — exploiting the panacea solution of the moment, outsourcing.

The CIO's agenda

Today's CIO agenda has been shaped to reflect this. Nevertheless, the barriers to delivery of a truly integrated enterprise remain the same — regardless of the location of IT and business process operations.

In 2004 and 2005, it is our analysis that the focus needs to return to the underlying issue, which is: how rapidly to deliver an integrated enterprise that provides both a measurable and a tangible RoI. In this context, today's CIO agenda reflects three priorities:

- **outsourcing, primarily to cut people costs**
- **extracting the maximum productivity from existing technology investments**
- **managing volatility and uncertainty (as discussed above).**

Into this mix must be stirred one consequence of IT becoming a commodity. This is that it has now to be managed as any other resource is in the business — whether cash or inventory or human resources.

No longer is IT protected, as a separate fiefdom that is impenetrable through its complexity. As with other resources, the objective is to make best use of the potential performance and efficiency of available IT functions and assets. Unfortunately, outsourcing is often only a limited response, and one that is primarily associated with 'improving' people performance, costs and efficiency. (On the plus side, though, it is accepted that — when managed well — outsourcing can enable organizations to remove redundant, non-core, elements from the value chain.)

Then there is the technology dimension. For example, take Business Process Outsourcing (BPO). This offers a glimmer of potential. However, contrary to media hype and supplier machismo, the reality of automating multi-dimensional business processes is light years away, for all but the simplest processes.

In this light, complex, varied and often unique processes within a broad-based financial services organization cannot be compared with (say) outsourced directory services or order fulfillment processes. Financial Straight Through Processing (STP), when delivered, provides an efficient means to integrate, control and orchestrate complex business processes. Successful STP is made possible through a combination of well defined:

- **process steps**
- **appropriate line of business applications**
- **integration software (middleware)**
- **infrastructure.**

In a real-time, highly volatile market place, information needs to be shared, decisions need to be taken and systems and infrastructure need to be connected in order to make STP happen. Yet, after the protracted, Internet fueled bull-run in the late 1990s, financial markets are even less certain than in the past. For example, trading volumes are:

- **less predictable (in growth terms)**
- **more highly volatile (in absolute day to day terms).**

As the levels of market activity undoubtedly influence broader business and investor confidence, unrelated industries need to be able to exploit these peaks with confidence. According to SpiritSoft, Rabobank has reportedly integrated legacy middleware transports by deploying JMS based ESB middleware, while maintaining STP integrity for complex customer and process interactions. (Rabobank provides financial services and products to the Dutch retail and business markets, operating in 34 countries around the world.)

The CIOs challenge: volatility on-demand

Volatility on-demand is, therefore, one of the most difficult challenges for a CIO to address. It requires the ability to shape dynamically configurations of systems and data that traditional, and statically configured, infrastructures cannot provide.

Imagine the implications of requiring the immediate distribution of data to many instances of an application running throughout a global wide area network. Similarly, imagine the performance bottlenecks that can arise when database access peaks at levels way above the expected.

In trying to deliver enterprise integrated applications, many integration software solutions seem to assume, or have assumed, that either static or predictable growth in demand would be present. In reality, yesterday's usage profiles are very different from those of today.

Return on previous investment

That said, it is also true — at the opposite extreme — that many organizations remain over-sized with too much IT infrastructure; they are frequently nervous, or unsure, about how they should reduce this IT commitment. Some of this has arisen because many past Enterprise Application Integration (EAI) and middleware solutions required investments that were influenced, or rationalized, by the expectation of:

- **continued growth**
- **static configurations remaining applicable.**

We can now see that economic pressures and initiatives, including outsourcing, have changed the equation. Application and data integration needs are more dynamic and demand variable scalability (upsizing as well as downsizing, together with the enabling of geographical flexibility) than was previously expected. One consequence is that expecting the anticipated returns on previous investments is no longer practical (or acceptable). New ways of thinking are required. Nevertheless, this presents a great opportunity to forward thinking management, especially management that considers alternative or complementary solutions as a result of the following factors:

- **the acceptance of a new series of open standards; these have been evolved with the help of both commercial and developer community input and have been heavily influenced by the success of the open source funding model that brought Linux**
- **the arrival of a new breed of agile and business savvy technology companies that have invested to create solutions and products based on these open standards; many of these lack the bloated overheads of traditionally staffed proprietary EAI providers.**

For example, SpiritSoft was formed as a result of a small group of technology experts having a desire to provide and support a lightweight message oriented middleware solution complying with the JMS1 specification. Unlike most other middleware vendors, SpiritSoft went further and provided the only complete Enterprise Service Bus (ESB) solution that combined, J2EE-compliant JMS standard middleware, XML, JCACHE-based caching (to address performance, productivity and volatility on-demand) — with an entry level cost of a few thousand dollars. This satisfies the cost conscious CIO, as it presents significant processing benefits as well as an attractive RoI.

Additionally, the SpiritSoft Bus-based architecture provides integration with third party middleware toolsets from the likes of TIBCO, IBM and Microsoft, as well as J2EE application servers (including BEA's WebLogic, IBM's WebSphere and open source offerings — like JBOSS).

The new agenda for technology investments

In Sensima's view, the shift towards outsourcing, even if it takes many years in evolving, represents a strategic change that will ultimately be unstoppable. The potential to reduce expenditure is too huge to ignore.

While India and China battle to provide outsourcing services (primarily people) at the lowest cost in the East,

There are Grids and there are GRIDS

When is a Grid really a GRID? This is becoming a highly relevant question.

As is the case with many new technologies, standards often trail commercial or academic developments and experimentation. Indeed, many of the EAI and ESB products and technologies mentioned in this analysis have both used and contributed to the development of new standards. JMS, JCA and other standards for messaging and application connectivity emerged after many years of proprietary product development. Grids/GRIDs are no exception and both warrant definition:

- **Grid computing is the commercial solution to an organization's processing — namely leveraging the existing PCs, servers and mainframes to ensure that processing requirements can be distributed across a network of hardware; it reduces the need for future hardware upgrades and ensures that processing bottlenecks can be smoothed out as well as reducing the risk of computing failures impacting a business**
- **GRID more broadly refers to the academics' concept of a global network of interconnected processors, where huge processing capacity is available to all (but especially scientists and researchers) to process and analyze vast quantities of data, or perform huge numbers of iterative calculations in support of areas such as meteorology, protein folding, particle accelerator analysis or drug research simulation.**

companies in the West will continue to own, operate or manage vast arrays of computer and communications hardware which will be distributed throughout their Global Wide Area Networks (GWAN).

Hardware connected within GWANs will embrace a diverse range of desktop PCs, servers and mainframe computers, each often designated to perform specific tasks:

- **desktop PCs for application and information access, and data entry**
- **servers for I/O intensive queries**
- **mainframes for compute intensive time critical operations as well as being the guardians of massive corporate databases.**

Many of these hardware resources, expensive when considered in aggregate, remain underutilized for many hours every day. Which is where Grid computing becomes important.

In one of its manifestations, Grid computing enables underutilized computing capacity, when connected to a network, to be orchestrated (using a Grid topology) to provide a collaborative processing capability. While still at an embryonic stage of development, Grid computing offers great potential to:

- **harness vast infrastructure resources**
- **release previously dormant processing power (for example, think of all those CPU cycles not being used on desktops overnight).**

This otherwise dormant processing power, when activated using Grid-type middleware, can be applied to:

- **performing research and calculation intensive operations**
- **offloading processing outside of normal business hours, perhaps to support outsourced business operations in different time zones**
- **providing greater resilience for disaster recovery, thereby protecting global firms from disruption.**

Powering a Grid

Several software vendors provide an alternative approach to a formalized grid topology, which may satisfy the needs of the large corporations. One of these, Spiritsoft, introduces 'traffic shaping' functionality which distributes data close to its needed location.

SpiritSoft's ESB (Enterprise Services Bus) solutions also combine enterprise integration and middleware capabilities with advanced data caching and distribution functionality. This unique combination means that overall processing capacity is improved given that servers access data in local caches (managed and updated by the ESB) rather than accessing databases over latent local or wide area networks potentially thousands of miles away (Figure 5.1).

ESBs with traffic shaping capabilities provide the potential to enable demand within a Grid or today's GWAN to influence dynamically the location and frequency of data distribution and caching, the adoption of an ESB a strategic imperative on the CIO's agenda.

Accessing and distributing data within a Grid is critical to

the success and viability of the Grid concept itself. Without data to process, distributed nodes within a Grid remain inactive and worthless.

In this overall context, it is relevant that both Sybase and Oracle have outlined plans — and possess early stage product offerings — which seek to support distributed Grid processing. While the direction taken by each company is quite different, their principle and common objective is to produce products that provide:

- **convergence of multiple data sources**
- **seamless distribution of processing**
- **distribution of application deployment logic**
- **a transparent, managed, Grid-like infrastructure.**

Oracle 10g

Oracle's approach is based on its Oracle 10g product family. According to Oracle Chief Executive Larry Ellison, this is 'the rebirth, the relaunch of our database business'.

Oracle's approach (Figure 5.2) is based on combining inexpensive hardware (for example, multiple cheap blade servers) with three complementary Oracle components:

- **its database**
- **its application server**
- **enterprise Grid management.**

As a natural evolution of Oracle's previous 9i strategy, Oracle 10g provides a scalable component-based solution that has deliberately been positioned for the expected growth in the commercial use of Grids — primarily those that are intra-business rather than inter-business. The Oracle 10g:

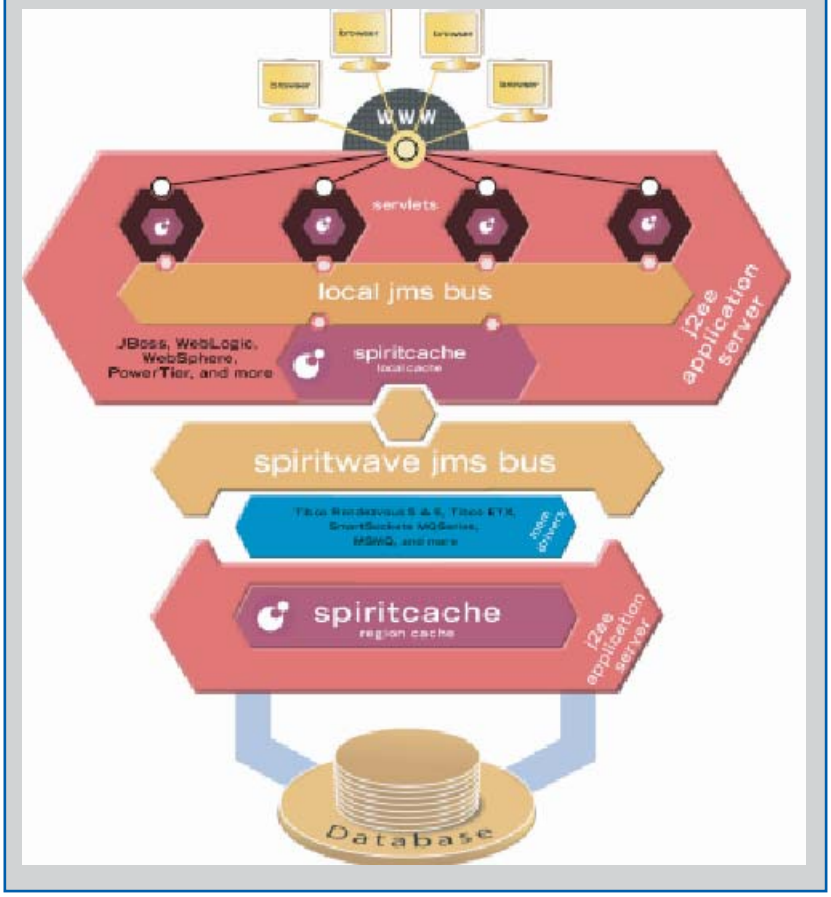
- **database provides the foundation backbone to store and persist application data**
- **application server provides the essential run-time execution container for business application logic and security/credential management**
- **the enterprise Grid manager automates the distributed management burdens that are usually associated with managing tens to hundreds or thousands of nodes.**

The last of these is, arguably, the most significant capability. Configuring a Grid of computing components can be complex (as academic institutions have discovered and proved). If this configuration and management complexity had to be accepted in the commercial world, it would likely doom Grid computing to failure.

One fundamental precept justifying Grid computing is that large numbers of computers (especially underutilized ones or inexpensive ones) can be combined to deliver greater processing power on-demand as well as greater resilience potential. Oracle 10g addresses this challenge through the use of auto discovery of new servers.

Auto discovery is not new. It is a well established principle that has been, and is, successfully exploited by network

Figure 5.1: A SpiritsSoft implementation



management and monitoring software providers. By exploiting standards and protocols (principally SMTP and TCP/IP), network management tools have been able to:

- **detect (or sniff or discover) server and application configurations**
- **dynamically build a map of communication between servers.**

Grid computing takes similar principles a step further. In many ways, Grid computing is the next natural progression of RAID and of clustering. Software and hardware versions of RAID (Redundant Array of Inexpensive Disks) have made it possible for commercial organizations to create large pools of disk/data storage devices (SANs/NASs) with built in resilience and performance load balancing. Indeed, Oracle 10g is heavily dependent on RAC, Oracle's approach to clustering.

In this context, Grid computing is the next step on from RAID and clustering. Grids make it possible for commercial organizations to:

- **create large pools of disk/data and processing capabilities**
- **combine this with resilience and performance load balancing**
- **exploit loosely coupled arrays of inexpensive servers and, increasingly, desktop PC's**
- **provide computing on-demand.**

In many ways, Grid computing is, therefore, about:

- **putting intelligence into the network**
- **drawing out performance through use of that intelligence.**

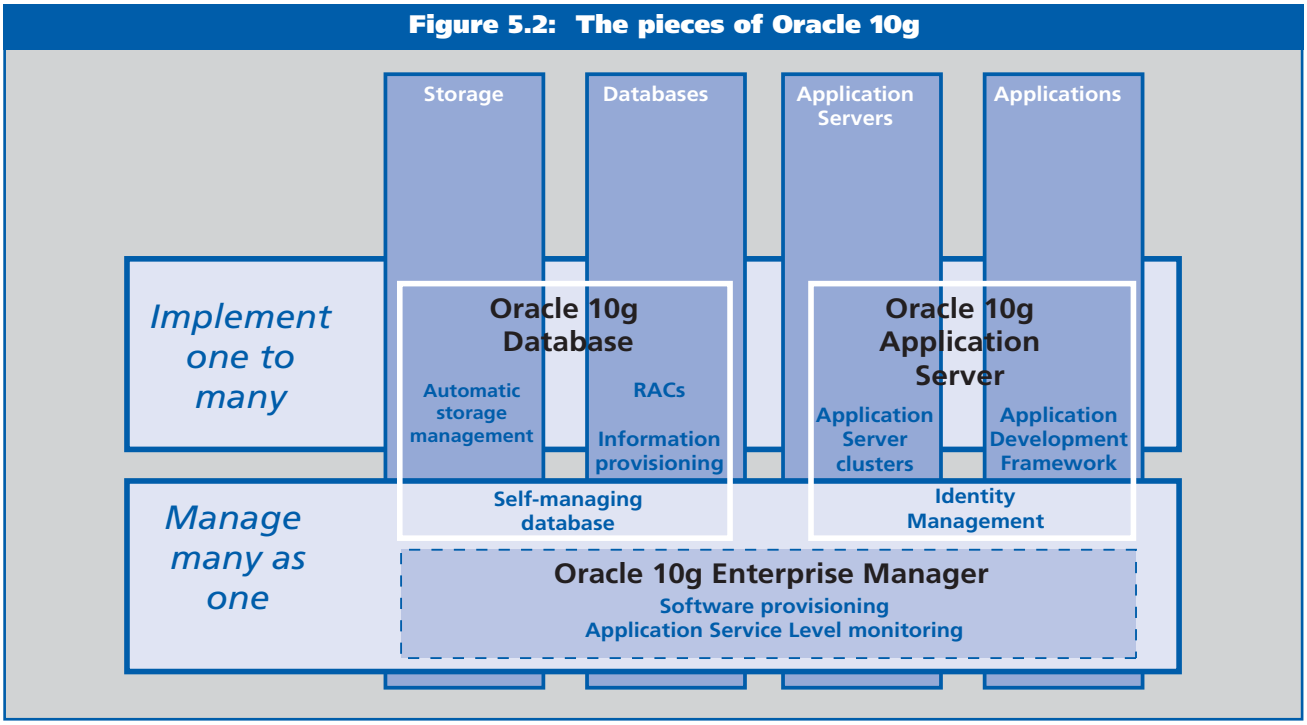
Sybase Adaptive Grid

Sybase's approach is altogether different. Arguably it is significantly more advanced. Indeed, in Sensima's initial analysis and when fully available, it will likely represent one of the most compelling practical implementations of distributed computing.

Sybase's aim is to leverage the emerging standards from Grid organizations such as the Globus Alliance and the Global Grid Forum. These provide an approach which supports a Grid system architecture that is based on Web Services concepts and technologies (specifically the Open Grid Services Architecture or OGSA). These standards, when implemented, will:

- **support access to distributed data storage**

Figure 5.2: The pieces of Oracle 10g



- **dispense with the need for repeated login or installation of a distributed file system**
- **simplify the use of remote systems (by providing a single standard interface for requesting and using remote system resources for the execution of processes)**
- **support secure authentication**
- **provide communication over an open network.**

These standards will be the basis for any modern network. What is different is their early integration in the proposed Sybase approach. This is designed to ensure a much simpler integration of existing systems, applications and even architectures. The proposed Sybase approach utilizes a Data Grid concept, requiring the collaboration of different hubs as opposed to the more traditional transactional clusters approach. An Intelligent Service Manager supports the whole process, effectively enabling the implementation of the Grid technologies to be completely transparent within the traditional Sybase product. This approach effectively means that, while the Grid technology is supported, it is invisible when installed to implementers and users. In essence it differs, therefore from the Oracle 10g approach is two significant ways:

- **first it is more overtly based on OGSA and related standards**
- **second its reach is not limited to being within an organization (as is the case with Oracle 10g); it envisions inter-organizational Grid computing being possible (although this will be wholly dependent on credible security, end to end).**

Management conclusion

The pressure on CIOs to achieve IT solutions that are cost effective, installed on time, within budget and provide a complete, strategic solution to the challenges of a global enterprise will increase. Past failures will no longer be tolerated, in these cost conscious times, and the drive towards outsourcing and distributed processing will put further pressure on the demands of a global IT infrastructure.

However, as new standards emerge, and low cost smart products appear to leverage the capacity of existing hardware and networks, the failures of past global implementations can be avoided. The journey towards standardization is likely to be just as fraught as the EAI maturity curve, except that — with Grid computing in hand — the rewards are likely to be much more powerful.

The CIO's dilemma

Mike Gilbert
Director, Product Strategy
Micro Focus

Management introduction

As IT leaders entered 2004 they looked hopefully at spending predictions to determine their investment strategy for the coming year. According to Gartner, "CIOs and other IT executives will have to spend the next 12 months holding down costs while innovating for the future". This is the CIO's dilemma for 2004 — balancing:

- ***the management of short-term issues like cost and complexity***
- ***while at the same time investing in the future.***

This same Gartner analysis goes on to offer some New Year resolutions to help balance costs against agile innovation. In one of these, CIO are urged to decide which legacy systems should be replaced and which should have their life extended.

In this analysis, Mike Gilbert of Micro Focus, examines three key business imperatives which can simultaneously:

- ***hold down costs***
- ***innovate for an agile future***
- ***enable decisions to be made about what to do with legacy systems.***

Apparently contradictory challenges

Each of the three imperatives (above) — taken on its own — is a formidable enough challenge. Taken together, each seems to run counter to the other two:

- **how can IT hold down costs while funding innovation and legacy systems replacement?**
- **how can IT innovate with no new budget and a portfolio of ageing legacy systems and software?**
- **what can IT do with its legacy that will reduce costs and facilitate agile innovation?**

Nevertheless, if there is a way to reduce costs and increase agility plus decide the future of legacy systems, then most CIOs will sleep better. The good news is that there is a way.

To understand this, it is necessary to turn the world over, to look at things from an unconventional perspective ... Conventional wisdom tells us that new is better than old. New computer hardware is undoubtedly better, faster and cheaper than ever before. Prima facie, replacement of old computers by new computers is necessary for organizations to be able to deliver their products and services better, faster and cheaper than their competitors.

Certainly consumers have come to expect year on year improvements, and expectations fuel this cycle. In the same way, the Internet must be better than old proprietary networks. Similarly the word processing package being used to write this analysis is better than a typewriter. Windows XP is much richer in function than DOS and Linux is cheaper than any proprietary operating system.

But does this also mean that:

- **new applications are better than legacy applications?**
- **COBOL business processes are obsolete because they can be re-written in Java?**
- **human-readable business requirements and specifications should be re-written in XML because this is machine-readable?**

The longueur of applications and business processes

Business procedures written down and captured, whether in computer language or English (or any other spoken language), may be relevant for many decades, or even for centuries. For example, the city of Massri Treasurers used double-entry accounting (implemented by all major financial packages today) as early as 1340 A.D.

According to Gartner, 75% of the world's business is still processed by COBOL applications. These applications amount to some 180—200 billion lines of code. That inventory of business 'scripts', written over the last 50 years, is equivalent to about 10 million books. As a point of reference, the British Library Public Catalogue provides access to over 10 million texts. Most large enterprises may, therefore, have an application reference library of several thousand volumes which define and automate the way they do business.

In a recent report, Sandvik Coromant estimated that it had invested 1,000 man-years of development in its legacy applications. These applications 'capture' business processes developed over many years. In Sandvik Coromant, as for many enterprises, those business processes are the necessary ingredients for being able to do business today — and in the foreseeable future.

In this context, executives at Sandvik Coromant have acknowledged that there is no guarantee that any large-scale technology conversion would not become obsolete in three years time. Technology moves on just as surely as business processes mature and stabilize.

Furthermore, computers and software infrastructure are now IT commodities available to everyone. But applications, specifically their implementation and operation, are unique to each enterprise's way of doing business.

These applications, therefore, embody the data, processes, rules and concepts that are uniquely intertwined with the people who run a business. This is what makes each business different from its competitors.

In the well-worn analogy, the bathwater (the legacy platform) is dispensable, but the baby (the application) is not. The challenge remains: how, then, can IT unlock an application which contains valuable business processes from a legacy platform that is restricting the ability to hold down costs and innovate for that agile future?

Unlocking the value of the legacy and holding down costs

The fact is that core business applications — and the language used — can be preserved as one upgrades the hardware and software environment with better, faster and cheaper versions. This is possible because the software environment for COBOL applications is not only replicated on contemporary platforms — like Linux and Windows — but it is fully integrated with the latest technology features possessed on those platforms.

For many organizations, the traditional mainframe is no longer the most cost effective platform from which to operate core business services. Fierce competition between hardware platform vendors has driven new server technology to 'enterprise scales' — with prices that mainframes cannot match.

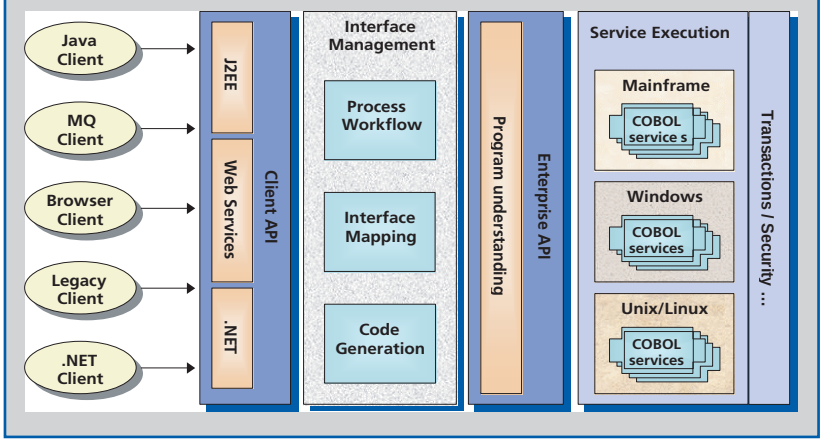
Moving an application to a low-cost platform (such as Linux or Windows) can reduce or altogether remove mainframe operating costs that are currently locked up in IT infrastructure budgets. Because the software is preserved, this move need not disturb business continuity any more than a conventional mainframe upgrade. For example, call center employees can still access 'green screen' transactions — even though these are now running on a Windows server, from the same terminal emulation program running on desktop computers. For some, this move also avoids serious risk as hardware vendors withdraw support for many of the older mainframe types.

For many organizations, a moderate initial investment to set-up such a new platform will yield significant year-on-year cost savings. RoI can often be achieved within a few months. Budget amounts that were previously allocated to annual infrastructure costs can then be released and redirected to fund new business value enabling projects.

Unlocking the value of the legacy and innovating for an agile future

What has also become clear with time is that core business applications which have been moved to a new platform are much more easily extended when using contemporary features of that new platform. New IT user communities (such as Web clients) can now access the same business logic and traditional data via the Internet, XML and Web Services.

Figure 6.1: COBOL in .NET and J2EE



For example, new technology makes it possible to:

- **re-use and extend core business applications in a Service-Oriented Architectures to build an agile 'services' infrastructure for innovative business development (but see Tom Welsh on page 16)**
- **run COBOL applications inside (say) Microsoft's .NET framework and fully exploit the connectivity and productivity features of its latest versions of Windows as well as integrate seamlessly with VB.NET and C# applications**
- **build quickly competitive client-facing services**
- **establish innovative delivery channels using diverse but integrated technologies including core COBOL applications, Java applications, packaged applications and many other key business services; this is achieved through the Internet, Web Services, J2EE and leading-edge EAI and middleware technologies.**

Furthermore, mainframe IT staff can now team up with the new IT community of Java, Linux and .NET developers. Traditional applications can be developed and extended using Visual Studio, which brings unrivaled richness and ease of use features (Figure 6.1). These help IT to become more agile. Indeed, COBOL teams can now enjoy the productivity of using the new paradigms of application development — including UML and Model-Driven Architecture approaches and exploit best practices for rapid innovation and rigorous processes across an IT organization (Figures 6.2-6.4).

This merging of teams and best practices, when properly managed, can overcome the frustrations and delays previously associated with bridging old and new technology centers and architectures. What is so welcome is that previously 'old' but core applications become a vital part of new, agile IT — preparing the architecture and services for the future. Legacy platforms can now be retired without compromising an IT architecture. But core applications can live on, in new guises melded together by architectures, new systems, modern development techniques and middleware — as the following example from the finance sector illustrates.

XML meets COBOL = integration

In the banking world, the key to delivering world-class client service is providing fast and secure financial transactions for

customers. To power monetary exchanges, banks require high-performance IT solutions that expedite reliable transactions. Additionally, today's banks face the challenge of integrating disparate technologies and varying institutional standards, which — too often — can act to hinder information flows.

To build applications that will facilitate and drive faster, efficient financial transactions, Deutsche Bank Luxembourg (DBL) needed seamlessly to integrate disparate platforms on a local as well as on a global basis. The need for a strategic, standards-based integration led it to adopt XML as an interface medium for its new data interchange requirements.

However, because the majority of today's banks' core systems were written in COBOL, DBL required a technology solution to ensure interoperability between XML and COBOL. "At first sight, integrating XML and COBOL seemed rather bold," said Roger Engel, Vice President and head of the IT development team at Deutsche Bank Luxembourg. "Just when we were wondering if such a thing could be done successfully, Micro Focus introduced an XML solution from its 'Net Express Early Adopter Program'. (For us) the timing could not have been better, for we had an immediate answer ... and were able to proceed with the integration."

The integration project that DBL contemplated was by no means child's play. It consisted of combining:

- **'Mozart', a well-established Windows-based COBOL application that DBL had used to develop the input, processing, reporting and routing securities orders**
- **a Portfolio Management System running on a new third party UNIX based system that the Bank had acquired to manage its portfolio activities, including bulk security transactions.**

In addition, not only was DBL looking to achieve interoperability between these two systems but it was also seeking a flexible, future-oriented solution that would facilitate communication between Mozart and the Bank's other systems. As is common in the finance sector, DBL possessed a range of different systems from both internal and external suppliers, and based on different standards and platforms. These systems included Kondor+, SWIFT, Direct Orders and IBP.

Within this heterogeneous infrastructure, Mozart is a vital application. DBL staff use it for the input, processing, reporting and routing of financial instruments, such as

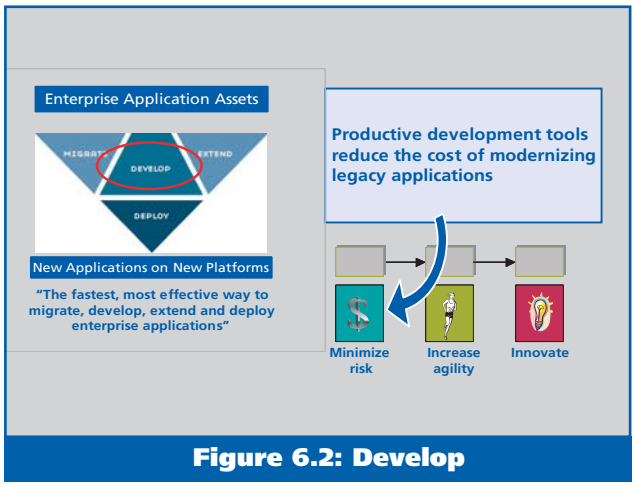


Figure 6.2: Develop

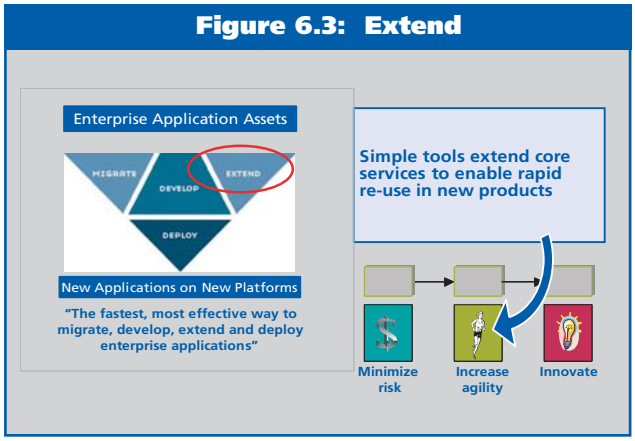


Figure 6.3: Extend

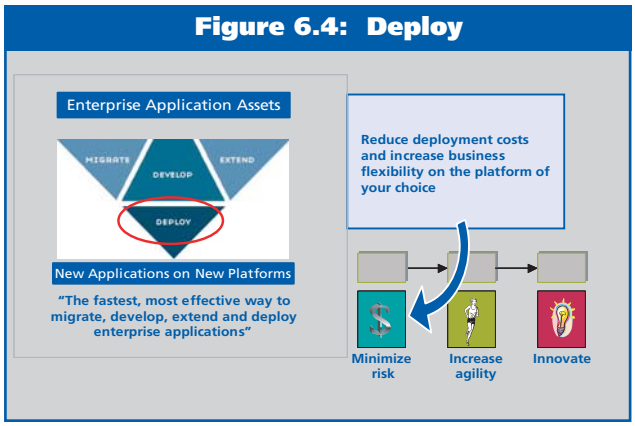


Figure 6.4: Deploy

Money Market, Foreign Exchange, Securities ordering and Payment transactions.

Indeed, Mozart (in DBL) was synonymous with efficiency and time savings, but only on the condition that the system worked smoothly and efficiently with the Bank's other applications. For this purpose, the Bank had already implemented 'point to point' interoperability between Mozart and many of its other systems — via individual interfaces.

Since many of those interfaces were already developed, the simplicity, flexibility and standardization of XML as a medium for data interchange — within the enterprise and across the Internet — made it a logical choice as a means for delivering interoperability in a standards-based way. DBL had realized that it needed XML for a number of reasons, to:

- **reduce implementation costs of future integration projects**
- **deliver improved capabilities more quickly in the future**
- **reduce deployment costs**
- **lower the total cost of ownership for the entire application**
- **reduce annual operating costs per transaction and to use state of the art technologies for data exchange.**

Micro Focus Net Express COBOL/XML support provided a COBOL orientated approach which delivered an easy to use mechanism for the creation, consumption and updating of XML documents from a COBOL program. Based on COBOL file handling syntax, XML-orientated extensions enable the handling of dynamic XML documents and XML features, such as namespaces and attributes. Wizards map XML schemas to, and from, COBOL data definitions.

In a matter of weeks, DBL was able to create a new XML I/O module in COBOL that:

- **consumed the XML document from RPMS**
- **fed that data through to Mozart**
- **when Mozart had completed the transaction, updated the XML document for return to RPMS.**

This enabled the Bank quickly to integrate the new RPMS package with Mozart, facilitating the consistent and timely processing of bulk portfolio transaction requirements while also re-using the strategic investment and logic within the Mozart system. DBL is already considering what it will seek to accomplish next with its COBOL applications. One dimension envisions a transition to Web Services (Figure 6.5).

Management conclusion

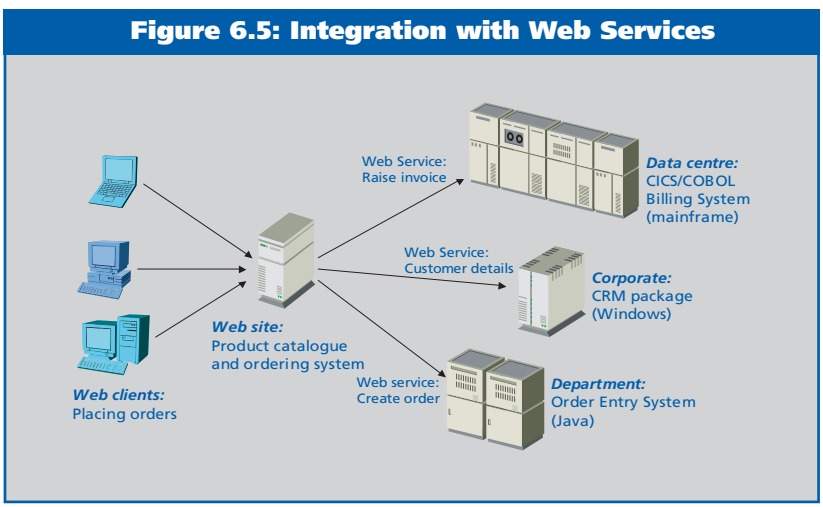
To understand the value locked up in legacy applications, simply estimate the cost of doing business without them. Alternatively, estimate how much it would cost to rebuild that legacy application from scratch.

Sandvik Coromant concluded, not unreasonably, that a wholesale rebuild project would amount to a repeat of the original investment of 1,000 man-years. The risks associated with such large-scale projects are obvious. Rightly, few CIOs would put their jobs on the line for such projects. Even fewer CFOs would sanction the costs in current economic circumstances.

As Mr. Gilbert argues, it is possible now to upgrade to better, faster and cheaper new platforms (not least Linux and/or Windows) in order that IT can re-use existing business processes (and the languages in which they were written). Doing this means that CIOs — and their organizations — can avoid the high cost and risk of 'ripping and replacing' several thousand volumes of business 'scripts'.

Instead, those applications can be readily adjusted by a newly agile IT organization to offer more agile 'services' which can be used over and again in new business initiatives. This is one project where a CIO can address all three of the challenges

Figure 6.5: Integration with Web Services



(described by Mr Gilbert at the start) and can be solved with a single initiative:

- ***costs can be reduced by moving the IT operation to a new (better, faster and cheaper) platform***
- ***agility can be increased by modernizing software infrastructure through the adoption of a Service Oriented Architecture and the merging traditional and contemporary languages and tools to construct new business applications***
- ***legacy issues can be resolved by unlocking core business applications from their legacy platforms — to become agile services running within the new infrastructure.***

This is good news. It offers a clear path forward to resolve the dilemma faced by most CIOs today — how to hold down costs while innovating for the future. Now, CIOs can decide which legacy systems should:

- ***be replaced***
- ***should have their life extended.***

Constructing a digital dashboard

Nick Denning
Chairman and Chief Technology Officer
Strategic Thought

Management introduction

In this analysis, Nick Denning — of Strategic Thought, based in Wimbledon, England — puts forward the hypothesis that all sophisticated businesses need a 'digital dashboard' in order to continue to be successful in the future. Such a dashboard is going to need middleware, and 'components' in particular.

But, before discussing what the result might look like, he examines the rationale and the steps that are required.

Management capabilities

The most valuable element of a company is its management capability. Increasing the effectiveness of management capability through the use of IT is fundamentally about:

- **improving the quality, timeliness and availability of management information to make informed decisions**
- **optimizing collaboration across groups participating in decisions — to improve the efficiency of the decision making process**
- **implementing decisions at the most appropriate point to achieve maximum exploitation of an opportunity**
- **monitoring implementation to ensure effective execution.**

The most important aspect of a decision making process is to be able to take and implement decisions within the decision cycle of the competition. This keeps competitors off balance, unable to take the initiative and only capable of reacting.

Middleware enables the delivery of this enhanced management capability, through the operation of electronic work flow mechanisms. At Strategic Thought we believe it is advisable to separate work flows into two classes — structured and unstructured:

- **structured work flows are those that are fixed, typically operating around applications**
- **unstructured work flows are those we implicitly carry out through email, where we forward an email cc'ing all those we think are interested until we finally agree success and completion of the activity.**

Critical dimensions

Being able to compete is vital in business today. Enabling that competence is no longer simple nor straightforward.

In my experience of working both within my own Company (Strategic Thought) and with our customers and clients it is clear to me that the following can be considered to be essential to the running of competitive businesses:

- **the understanding of a business's strengths, weaknesses, opportunities and threats (SWOT); analysis enables us to construct a business plan — tempering it to select those activities (from the potential of list of possible activities) against which our organization can deliver**
- **maintaining a clear focus**
- **identifying the risks that are faced and then providing appropriate mechanisms by which to manage those risks (this ensures that these plans can successfully be delivered)**
- **generating management information from the work flows to track the progress of work against plans**
- **facilitating regular reviews of plans through feedback loops that can identify changes to the data underpinning the SWOT analyses**
- **providing mechanisms that alert one to specific opportunities and threats that present themselves**
- **determining what additional information and what decision processes are required to decide whether the potential opportunity (or threat) is one where action is required**
- **being able to make the decision and implement it within the available time window and — almost as importantly — within the decision making cycle of the competition.**

If these are the basics, let me now address:

- **why it is important to implement these mechanisms**
- **what the dangers of such a disciplined approach are**
- **what is required to implement this regime.**

Management mechanisms

A critical dimension to successful businesses is the management mechanisms used to deliver products and services to the market. At a recent seminar, given by a firm of financial advisors, the interesting comment was made — that, in determining the value of a business and giving marks out of 10 for the relative value:

- **the quality of management scored 7**
- **marketing and sales scored 2 and products 1.**

No doubt many will argue for other factors or relative weightings. Nevertheless the importance of investing in, and developing quality of, management was well made.

If executives are to provide effective management within organizations that have to deliver 'their mission' in an increasingly electronic world, the focus has to be on advancing management systems to optimize:

- **the decision making cycle**
- **training of staff to be able to participate in the various processes**
- **the efficiency of current processes**
- **risk management to identify issues that must be fed into the decision making process.**

Implementing electronic processes enables the efficiency of these processes. However there has to be a structure to those processes rather than just informal email. Otherwise the processes can stall or break too easily.

If this can be achieved, electronic work flow has the potential to deliver, but only if it can:

- **balance the discipline of a defined process with the flexibility of email**
- **generate management data (Figure 7.1).**

Thus electronic work flow is a key component of any digital dashboard. It must support both operational processes and change processes. In the following sections I will discuss why it is so important to include the management processes within such a work flow model.

Decision making cycle

The decision making cycle in every organization is different. At one extreme there is the structured, formal mechanism for pushing ideas up the hierarchy to obtain the authority to proceed. The challenge here is to encourage ideas to rise up the chain of command without becoming corrupted or diluted.

In others there might be a 'hands-on culture' where senior staff push decisions down the chain of command. Here the danger is that decisions appear arbitrary, because they are handed down (and imposed) from on high.

To improve the decision making cycle, the objectives have to be to:

- **define terms of reference for staff so all understand their involvement in the decision making cycle**
- **make available as much information as is possible as well as enable people to filter and locate the data they need**
- **possess electronic work flow processes that are initiated whenever there is a potential decision to be made, to which can be attached the relevant information as each process proceeds through the decision making cycle**
- **provide as much visibility as possible to every participant in the decision making process of the tasks 'coming down the pipe towards them' so that there are minimal surprises**
- **work at electronic speed and avoid the iterations that typically happen today**
- **once a decision has been made, define the business process to execute the decision, and ensure that the process is carried out to conclusion.**

In this environment, a manager's job involves:

- **reviewing of work currently 'in play' as well as determining how to prioritize work undertaken by subordinates**
- **identifying as early as possible in the cycle what data is required to make a decision and then ensuring that it is provided to the appropriate level of quality when the decision point is reached**
- **determining the point at which there is sufficient information to make the decision that maintains the tempo of the business yet exploits any opportunity arising to best advantage; there is rarely perfect (complete and accurate) information and there will never be unlimited resources so deciding when is an optimum time to act will always require judgement (and risk management).**

Electronic work flow (Figure 7.2) formalizes the process and therefore by definition establishes each person's roles and responsibilities. It also leaves an audit trail providing visibility of processes and so imposes a discipline on participants to work to the agreed model.

Thus a digital dashboard (Figure 7.3) materially improves an organization's compliance with HR and financial legislation by which it is bound. In essence, my message here is: when the time is right, make more informed, better decisions faster and be able to justify them when called upon.

That said, there are dangers associated with using electronic processes:

- **information overload**
- **waiting for perfect information (by the time you have perfect information you will have probably spent too much collecting it and it is probably too late or out of date); equally, of course, people have to be clear about the effort required to implement any decision —**

for an opportunity may not be capable of being exploited in a desired time frame
- **analysis paralysis: too much time is spent on working out what to do but in so doing little ever happens**
- **'the interfering forward' — in essence managers with apparent perfect information start to do their junior's jobs (training staff to do what you want them to do and then leaving them to get on with it is essential — while maintaining management's responsibility to ensure completion as well as offer guidance).**

The overall key to success here is to be able to think and react within the decision making cycle of competitors. When this happens, we must then be able to execute processes as effectively and efficiently as possible with the greatest likelihood of success.

Implementation

What do we need to achieve these objectives? Here are some of my priorities.

The first is 'Management Information Delivery'. We need some form of information portal:

- **which provides access to the information base**
- **through which filters can be applied that enable selection of areas of interest.**

With such facilities, it becomes possible to leverage sources of information including:

- **the knowledge base**
- **an understanding of assets and capabilities (and liabilities and inabilities)**
- **monitoring of work in progress**
- **viewing of risk assessment analyses.**

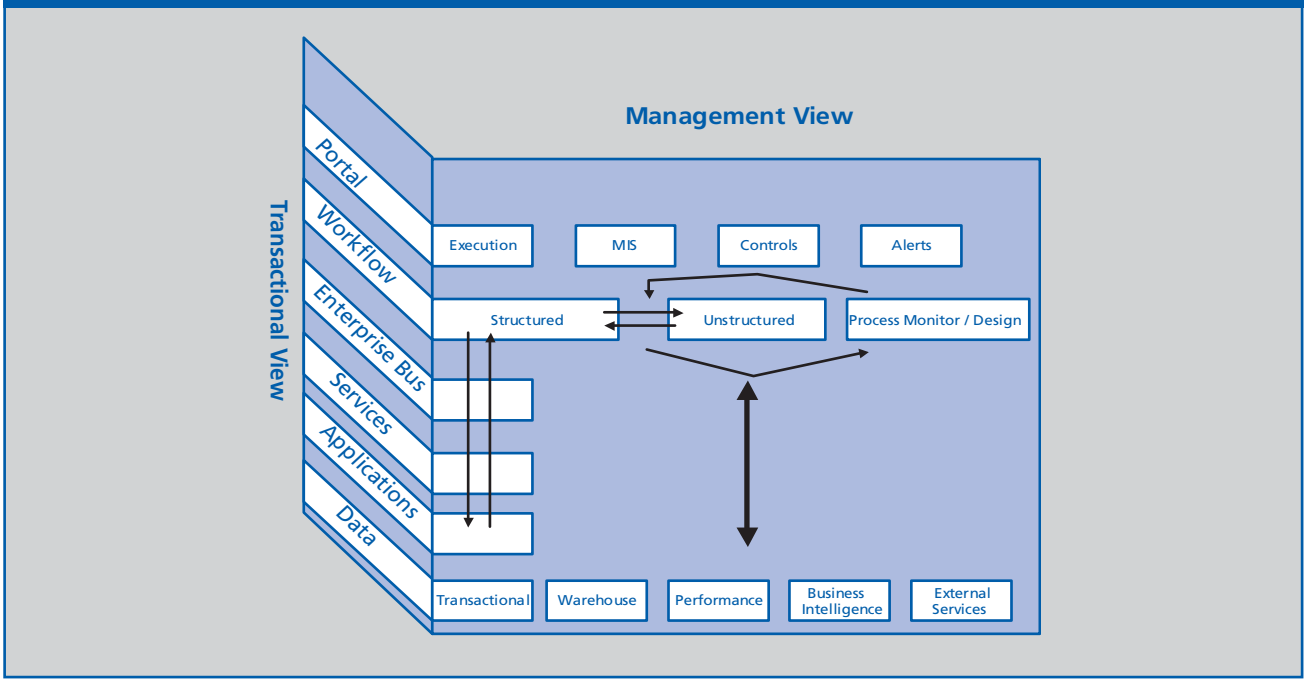
At the same time an alerting mechanism is essential. This can flag areas of interest that reach particular thresholds are ensure that these automatically are raised to the appropriate level.

Yet again, some form of work flow system is required. With this it becomes possible to initiate, manage, control and complete work that needs to be done by both staff and even ourselves.

Process management

An essential element of my overall proposition is the implementation of a process management mechanism through which all processes executed can be recorded and

Figure 7.1: Management perspective



managed. The key aspect — at least to me — of such a process management mechanism is enhancing the capability to:

- **define sets of tasks that must be completed**
- **order the sequence in which they are to be completed**
- **establish the inputs and outputs required for each task**
- **allocate tasks to individuals (for process completion)**
- **initiate, monitor and control the created process through its lifetime.**

To achieve this one needs process templates plus indexes plus search mechanisms so that previous definitions (for example) can be found and re-used or modified for re-use. The objective is to create specific instances of a process which embody previous experience while still being available for subsequent tailoring for individual or new processes that arise as circumstances change. Equally, processes must be available for storage in templates so that they are themselves available for re-use.

Similarly, one needs to define the data that must be contained within each process and made available to each task. One must be able to attach unstructured electronic data to each process as specific tasks are executed, so that the information is available for the next task.

Where tasks involve access to applications and are, therefore, beyond the process itself, the need is for catalogues of tasks plus the associated interfaces (to those pre-defined tasks) that are required for these to operate. In this way one can include such externalized tasks in work flows.

In addition, this process design mechanism must be capable of being operated by staff as a normal element of their work. This means that process creation and management must not require IT support to build.

Looking around today

As I look around today, much of this capability exists already. For example, Microsoft Outlook provides a wealth of relevant functionality. Indeed my own company uses Outlook as far as is practical to achieve many of the objectives above.

That said, it does not quite go far enough. For example, its work flow is weak (if not non-existent — or else we have not found out how to use it). We can, for instance, track related emails and other items but that is about it.

At another extreme, various IBM technologies within the WebSphere family provide:

- **business modeling**
- **work flow**
- **content management**
- **monitoring**
- **application integration.**

But, today, these are predominantly focused on work flow connected to business applications. Traditional work flow understands the 'people' element and communicates with the underlying business applications. The downside is that skilled IT resources are required to implement this type of work flow.

My basic premise is that what we need must fundamentally:

- **be focused on a more ad hoc work flow**
- **support much greater degrees of dynamic collaboration between users focused around a process knowledge base.**

Implementation

When you read this analysis, I do not wish you to take away the impression that my purpose is to have you buy yet more technology. Buying new systems is not what is required. The prime objective is to place into the hands of users the capability for them to do their jobs better.

For the last few years the driver for technology change has been mainly to satisfy the needs of the IT vendors to sell its latest technology. It achieved this — at least until 2001 — by convincing us that the old 'stuff' was no longer good enough and that what we really wanted (needed) to do was upgrade in order to obtain lots of new features. That said, I would also argue that IT vendors have certainly delivered new capabilities into the business environment, with the two most significant ones being (arguably) the Internet and Outlook.

But there is also only a certain rate at which people (and organizations) can change while still being able to run their existing businesses. Not unreasonably, there is only a certain amount of appetite for risk within each business.

Fundamentally, therefore, it is necessary to accept that one size does not fit all and that when exploring any new technology and assessing its appropriateness to be deployed one tries first to address the people issues within each organization — in essence to ask:

- **what capabilities does it possess?**
- **what is its value — as innovator, early adopter, mainstream or even deliberate laggard?**
- **what is the relative risk-adjusted return when comparing any particular proposition against rival investment opportunities?**
- **to what extent can one design and implement a phased approach, and what is the minimum investment to implement and prove any first phase?**

Once an organization has decided to proceed, it is vital to manage the delivery process. Were it to be suggested to an organization that Microsoft Office and Windows (say) be thrown out and all replaced with Linux and OpenOffice, most CIOs would sigh gently, and turn away. On paper the value argument might be compelling. But it is just not going to happen. The personal investment in understanding the quirks of Word or Outlook or Excel or PowerPoint is just too great.

However once portal middleware has been placed on someone's desk with secure single sign-on access to email as well as to other applications and to other facilities on other machines or devices in an organization, the individuals who might previously have resisted a transition might see such a change as manageable and therefore acceptable. That this might initially just be to a browser version of Outlook — who cares?

The implementation objective must therefore be to construct an infrastructure which can simultaneously:

- **deliver capabilities to users**
- **empowers those users to demand progressive and cost effective change.**

The role of IT should, therefore, be to provide the infrastructure to enable subsequent and cost effective change.

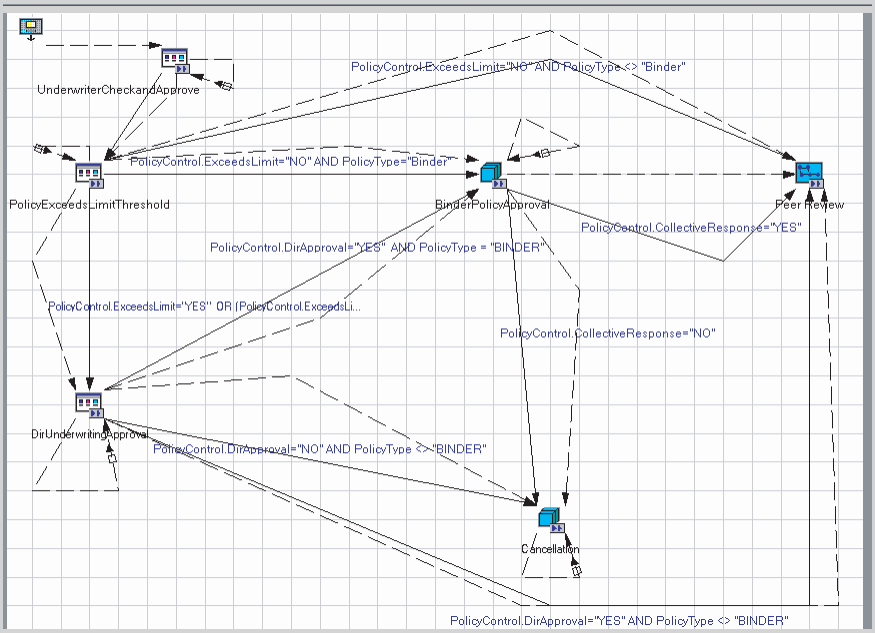
Components and Service Oriented Architectures and middleware

At this point I can circle round to an increasingly common theme — the necessity to introduce a Service Oriented Architecture which exploits components. With this, a user should be able to build the processes to exploit those components without the need to possess large and expensive IT resources to analyze, design, deliver and test the resulting applications.

My point is that this also allows one to focus on where value may be added. To me the places where IT can then provide a significant value add are:

- **enhancing operational performance and quality of service in terms of the security, reliability and performance of the underlying infrastructure; this is complex and requires specialists**

Figure 7.2: Example of work flow



- **delivering new business functions and/or components that can be engineered into business processes**
- **planning, managing and undertaking the complex tasks of migration of the infrastructure and the associated regression testing to ensure that all the existing processes continue to work after the migration to minimize disruptions**
- **working with business units to explain the new capabilities that can be delivered and then matching these to the business needs in a manner that avoids 'geekery' and is, instead, from an IT function that perceived to be part of the overall business team.**

Less is more. A critical aspect of this role of IT is to work out how to present and discuss new capabilities with an extended range of stakeholders (which may include customers and/or suppliers, for example) that should be participating in the discussion. Why should users care whether the business uses J2EE or .NET — provided they understand how to operate the interfaces they are given.

Thus, probably the most important initial decision an organization will make in the future will be associated with its portal strategy and its GUI standards — even though there is relatively little direct business benefit apparent when a portal is first deployed. As so often with other middleware, a portal provides an essential architectural component to remove once and for all that most rapid consumer of resources 'IT fashion'.

Specifically with regard to implementing a digital dashboard, the need is to:

- **filter data to avoid information overload**
- **use alerting mechanisms to identify scenarios outside your current focus but within your area of responsibility**
- **maintain an appropriate management structure to avoid interfere forward**
- **avoid creating onerous processes that do not add benefit**
- **focus on improving decision making speed rather than trying to make perfect decisions**
- **improve the quality and timeliness of information but accept that it will be imperfect**
- **do not try to store all the data received on a just in case basis unless you are certain you can re-use it**
- **shorten the feed back loop so as to influence the process or correct mistakes in good time.**

Where are the quickest returns?

Much of my inspiration for this analysis has come from attending recent lectures and seminars related to the justification and design of the Defence Information Infrastructure (DII) project, US Network Centric Warfare and the UK Network Enabled Capability. These are weighty (and expensive) initiatives. I have been extremely impressed by the quality of thought and the ability to present complex ideas by the people involved in these huge projects.

However they are huge. As a consequence they will take a long time to reach fruition. Nevertheless, the need to deliver the ideas described above is certainly real and is going to happen. It is not just a pure blue sky hypothesis.

In the commercial arena I believe we must focus on the evolution of broadly equivalent types of capability (and delivery). Our steps will need to be in a series of smaller phases which prove that the ideas identified above:

- **can be delivered**
- **provide a return on investment.**

Let me, therefore, consider where first to look for early wins. The following list offers relevant pointers:

- **look for scenarios where a core function of the business is the management of documents; this might be a core competence or it might be a vital supporting function or even observance of a legal one (as in compliance)**
- **seek out areas where there are significant risks associated with document management**
- **establish where it is essential, if only for compliance or similar reasons, that an audit trail is maintained of work carried out in a process and where such an audit trail must not be altered (accidentally or intentionally)**
- **identify areas where the time to take decisions is critical or where tracking of work in progress is important**
- **find out where contractual obligations on the organization exist, especially those where performing identified tasks by specific dates is the measure of delivery**
- **look for where there are financial benefits — or penalties — for being able to demonstrate appropriate management controls, risk management and previous history; examples here might include the ability to obtain reduced insurance cover such as professional indemnity cover or where large fines by regulators can be imposed for compliance failures**

- **identify where the provision of quality of service is a key factor in establishing competitive advantage.**

Which types of business might accommodate these categories? There are many.

The multi-office professional services example

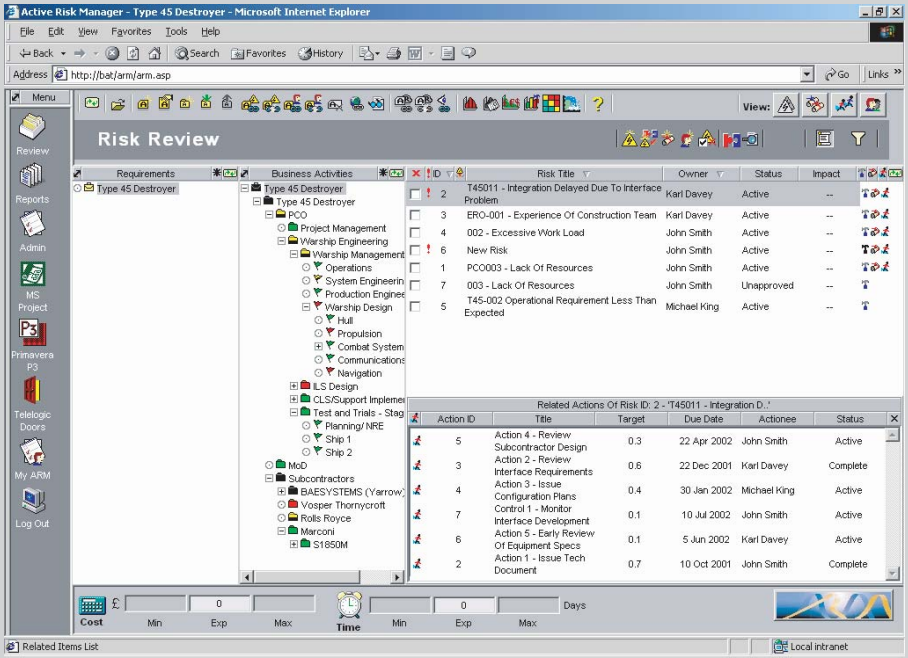
For example, consider a multi-office professional services businesses such as property management. Here there are a large number of documents related to each individual property portfolio under management that need to be held in a consolidated manner. However they may need to be accessed by people in offices scattered over a wide territory, especially where the services related to each individual property in the portfolio must be delivered differently:

- **there are large numbers of contracts involved; each contract must be filed correctly and cannot be allowed to be lost**
- **when rental periods or leases come to their break points, there are key legal requirements to ensure that the customer's interests are appropriately served**
- **it is expensive to fax documents between offices but the delays to executing time-critical processes if documents are mailed can be significant (and the risk of losing documents is also non-trivial)**
- **if problems occur then the cost of failing to respond to (say) a rent renewal notice by a landlord on a long term lease of a large property can be in the order of millions of pounds.**
- **the organization provides a range of services to its clients, delivered by different teams which must be coordinated**
- **highly skilled and expensive engineers have a significant amount of unproductive time; significant efficiencies can be obtained by coordination and communication to enable lower skilled staff to carry out straightforward tasks under supervision — if effective collaboration is possible.**

Thus even in a relatively low tech engineering discipline such as property portfolio management (a task undertaken somewhere in most organizations) there is a need for:

- **document management to control paper**
- **work flow to initiate and track complex processes and to ensure that they are completed within required time frames**
- **monitoring of performance and demonstrating delivery against service level agreements**

Figure 7.3: Risk management digital dashboard



- **integration of processes with external suppliers such as between in-house and retained lawyers, accountants, engineers or landlords**
- **an audit trail of tasks accepted by either party**
- **knowledge management mechanisms, so that specialist expertise can be communicated and exploited across the organization**
- **risk management systems to identify the range of associated risks — from health and safety to environmental, project management and financial returns; all must be understood and managed across the organization, not least to demonstrate to insurance providers that effective risk management is in place (which will either reduce the costs of cover or prevent these from rising)**
- **secure use of the Internet and browser based applications — to reduce the costs of collaboration and IT support.**

The construction example

Take a related, but different, industry — construction. The construction industry has a large number of challenges, as we have seen with the range of problems that have come to the UK public's attention within its rail industry over recent years.

The following characterizes the work undertaken in this sector:

- **multiple offices are located across many sites, between which telecommunications costs must be managed while still ensuring access**
- **there are large numbers of individual project sites where business is undertaken**
- **the ability to establish major cost savings through centralized buying in bulk of standard products is possible**
- **many separate invoices are processed across all sites and projects, thereby introducing major costs to a construction organization if the per invoice cost of processing is not minimized**
- **time management and performance is a major component of each project's cost; it is necessary to ensure management controls are in place to ensure services are delivered to the required quality before payment**
- **there is a need to provide prompt payment to avoid disruption to projects through staff being re-assigned by a subcontractor, or to ensure that required supplies are delivered to site at the right time**
- **the ability to locate, access and annotate in a reliable way engineering diagrams and other documents is critical to the integrity of each project**
- **risk management systems are needed to ensure that the budgets of large projects are managed and the correct mitigation actions can be identified and then taken.**

Common 'factors'

Personally, I suspect that any organization which involves large numbers of people, with requirements to vary (even only slightly) the work that each must do each day, is a potential target for the delivery of an architecture that manages and co-ordinates collaboration — by bringing together portal delivery, business intelligence, knowledge and content management, risk management, process modeling, monitoring and tasking.

The place we should look first for common factors which can then lead to deployment are those opportunities where the reward is greatest and the time to deliver is quickest. Some characteristics stick out:

- **multi-office organizations, where there are significant benefits to be obtained via telecommunications cost reduction and by co-ordinating paperwork and making it available on line to all offices**
- **predominantly 'paper' or word processing activities, ones which do not use business applications (often because enterprise integration takes so much time to engineer)**
- **collaborative processes, between individuals**
- **controllable processes, such as those where major savings may be made (at least from a financial perspective) if these can be streamlined (examples include invoice processing and payments processing)**
- **activities where there are measurable financial risks if failure to deliver occurs**
- **where a business is sophisticated enough to have been able to construct a return on investment (RoI) justification and has the appetite to measure that the RoI is being delivered**
- **an organization has project management capabilities and has the IT expertise to implement this IT capability (or knows that it does not possess this and is prepared to outsource the delivery of the network and the applications) to ensure delivery against service level agreements.**

Management conclusion

One size does not fit all, and each organization must determine its own approach to evolving its management practices using IT, taking into account the nature of the industry, the capabilities of employees and the potential RoI. Much data exists but most businesses are unable to utilize that data effectively. However if all data available could be delivered then there would be information overload, so it is important that filters can be applied so only the most important data is pushed to a user.

Currently most organizations only find out about problems or opportunities sometime after they occurred by which time it is too late to do prevent or exploit. It is important to be able to see what is happening now, and predict what is likely to happen in the future and hence avoid being surprised. This is why the digital dashboard (Figure 7.3) is so attractive — and, potentially, obligatory — for the future.

Once an opportunity has been detected, organizations must determine the optimum point for exploiting the opportunity to maximum advantage. This must involve all participants in the development of the decision so that the final 'GO' decision is effectively only a confirmation. To achieve this, Portal technology is required — to provide the delivery channel, collaboration work flow and application work flow to implement processes supported by publish and subscribe delivery to distribute data to the delivery channels. Into this infrastructure one then deploys the information repositories and business applications specific to each industry.

The most vital aspect of this is the process repository and the ability to create ad-hoc work flows, implement them and track them to completion without recourse to IT. However, as important, is the ability of an organization to be able to justify starting down this route by identifying a journey of evolution based on a number of discrete phases, each of which provides business benefit. In essence, as IT providers, it is imperative that all have to work out how to provide the capability rather than deliver the technology.

MIDDLEWARE**SPECTRA**
is published and distributed worldwide by:

USA and Canada:
Spectrum Reports, Inc.

Subscription Center
PO Box 32510,
Fridley, MN 55432, USA
Telephone: 763 502 8819
Fax: 763 571 8292

UK and Rest of the World:
Spectrum Reports Limited

Research and Editorial Office
St Swithun's Gate, Kingsgate Road
Winchester SO23 9QQ
England
Telephone: +44 1962 878333
Fax: +44 1962 878334

Subscription Centre
St Swithun's Gate
Kingsgate Road
Winchester SO23 9QQ
England
Telephone: +44 1962 878333
Fax: +44 1962 878334

Email and Internet

Email:
spectrum@middlewarespectra.com

World Wide Web:
www.middlewarespectra.com

ISSN 1356-9570

[incorporating FINANCIAL MIDDLEWARESPECTRA ISSN 1460-7220]