July/August 2000

interact

The Journal of Component-Based Development & Integration

CBDi forum

RIP Software Components?

What is the difference between a web service and a component? Will services supersede components?

in this issue:

Microsoft .NET, What, Why and When?	4
Are you being served?	9
Why Automate the Business Process?	14
Product Overview - Compuware's UNIFACE Eight	21
Product Overview - Makeover or Brain Transplant? Compuware's XPEDITER/DevEnterprise	25
Product Overview - Reviewer for Select Enterprise	29

The Journal of the Component-Based Development and Integration (CBD*i*) Forum

As the industry turns its attention to web services, stimulated by the Microsoft .NET announcements, we need to rapidly establish clarity surrounding the new concepts and how they relate to current practices. So what is a web service? What is the difference between a web service and a component?

You may be forgiven for thinking that software components are yesterday's news. The focus of today's vendor marketing budgets and consequent media frenzy is web-services, XML, SOAP and business process automation. Normally this wouldn't concern us greatly as this is mostly marketing froth, and hype is a necessary evil to make the world go round. However one of the side effects of excessive marketing is misunderstanding and we have noticed that in the rush to report on the new ideas, accuracy takes a lower priority. Just at the point where we have managed to educate the industry at large into understanding software components, so the game changes to web-services and everyone runs around creating confusion over the new concepts. The level of confusion and misunderstanding is actually breathtaking. For example:

- One of the most widely respected industry analyst groups reports that Microsoft's .NET platform replaces COM.
- Journalists are repeating this misunderstanding that web services replace components.
- A major European journal comments "the .NET strategy requires 100% buy in into Microsoft", ignoring the open nature of web service architectures.
- Despite the hype, almost all media commentators and analysts are indicating that web-services won't become reality for years, ignoring the obvious examples that are in widespread usage today.

And in time-honored fashion we are now waiting for the entire software industry to carry out a "global replace" on their marketing materials and claim that they are "the leading providers of web services". So what is a web service? What is the difference between a web service and a component?

continues...

'RIP Software Components?' continued...

Let's start with a simple definition - *a web service provides a location independent business or technical service in a manner that the consumer of the service does not need to be aware of the implementation of the component providing the service*. For experienced component designers and builders this is simply elementary. A component consists of an interface, an implementation and an executable. The interface specification provides the external perspective and a quality component will ensure that the component consumer can make use of the service provided by the component simply by being aware of the interface details. The web service adds contractual, security and transaction management capabilities that enable the service to be executed across the web.

For those that are confused specifically by the difference between COM and .NET, COM is the Microsoft programming model that delivers separation of interface and implementation (typed components). The .NET framework delivers additional component services enhancing COM+ and is designed to make COM development easier. The .NET framework automates the low level plumbing including reference counting, interface description and registration plus memory management and security. But under the covers the COM component is there and, with a couple of caveats, existing COM components will plug and play in the web services environment. We examine and comment on the .NET framework in this month's report Microsoft's .NET, What, Why and When[1]. Perhaps some of the confusion arises from the positioning around DCOM. Whilst DCOM might still be useful (performant and easy to use) in homogeneous NT environments, its place in the heterogeneous world is clearly now superceded.

For those that may be confused about openness and or dependence on any one vendor, the major advance that web services bring is the ability to use a service independent of the implementation. The only requirements are that you are using open standards in the interconnectivity layers (XML, SOAP and SCL - the new SOAP Contract Language). Naturally every vendor will work hard to convince its customers that its implementation platform is the superior way to deliver services, but consumers of services operate independently of the implementation. No rocket science here, just basic component thinking that we figured out ten years ago, but now brought to the mass market as the vendors roll out the infrastructure and the application services.

A good way to think about this is to consider there are three roles involved in service consumption. Figure 1 - Web Service Relationships, simply illustrates that the capability is delivered by a component builder, implemented and deployed by a component implementer and consumed by a service consumer.

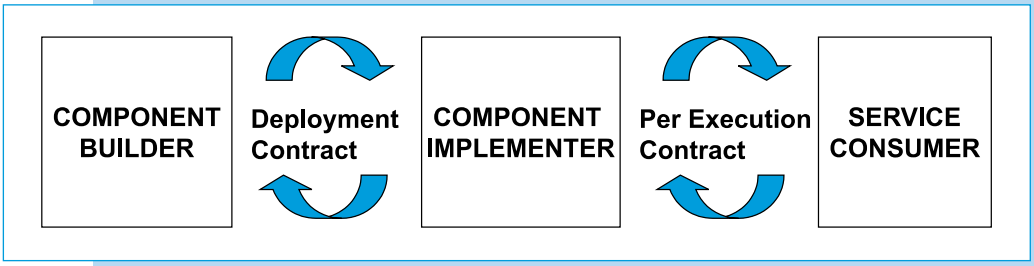

Figure 1: Web Service Relationships

What is interesting and important is to consider the very different contractual relationships between the three parties. The contract between the builder and implementer is conventional, covering licensing for reuse generally on the basis of known numbers of servers or clients. The contract between the implementer and the consumer is however focused on a per execution perspective, with execution and access rights and data protection. Note these perspectives are equally common to a web service that provides a business service such as "check credit" and a service that provides a software capability such as "use email and personal organizer services".

A key characteristic of web services will be the degree of flexibility and choice that a specific service provides. If an ERP package provider offers a monolithic package as a set of web-services the benefits being provided are confined to the delegation of operational responsibilities. This is typical of the current trend to ASP based services. So whilst the consumer should not need to be aware of the implementation details to use an individual service, it is absolutely critical they are aware of the dependencies between the provided services. There are several types of dependency that the consumer should be informed about. First the

© cbdiforum.com Limited, July/August 2000

existence of integrity units, which will potentially facilitate the speedy upgrade and replacement of functionality, in other words a measurement of the relative extent to which the services are provided from a component based environment as opposed to a monolithic environment. For example is it mandatory that the services ADD CUSTOMER and ADD ORDER are provided by the same set of components? Whilst the implementation is transparent to the web service, of course it is not irrelevant when it comes to change. It is therefore important to understand the dependencies and to ensure that these are implemented in interfaces and not hard wired. Figure 2 illustrates the options between fine and coarse grain components and emphasizes the need to ensure that dependencies are implemented through the interface, which of course is why we still need interface models and component architectures underlying our usage of web services. Second the compliance with a set of application standards, which might be implemented as a component based framework that enable the services to closely collaborate. Third the compliance with a set of semantic standards which might be implemented as a group of services or a service bus. For example the service(s) are compliant with a set of standards that might apply to an organization, supply chain, an industry group etc that allow the services to loosely coexist, exchange information and collaborate.

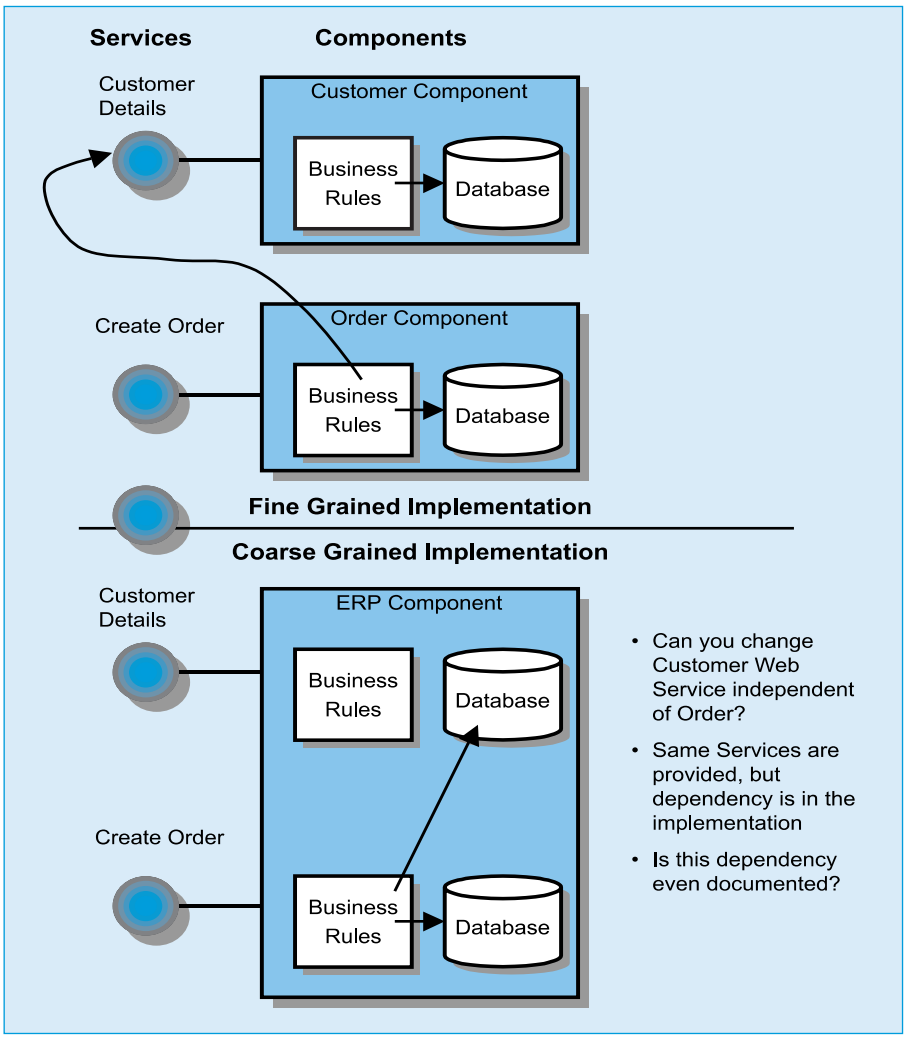

Figure 2: Service Implementation Options

Another key characteristic of the web service will be the way in which the service can be integrated. In general we have been advising for some time that integration should be implemented as a separate layer, and certainly independent of the business logic application layers. With web services we will recommend that maximum flexibility will be achieved when the consumer implements the integration, which is then independent of the implementation layers. We will anticipate that many early initiatives to claim support for web services will not deliver optimum flexibility in this area and consumers will need to consider very carefully the constraints that this may impose.

Effective deployment and usage of web services requires all the infrastructure and architecture necessary for components plus the additional layers that deliver the increased separation and security. The good news is that if you have already embraced component thinking and practices from either the provider or consumer perspective, moving to web-services is going to be relatively straightforward. In this bumper July/August edition of INTERACT we focus on the closely related topics of web services and business process automation with reports on both the technology and business drivers. Lawrence reports on .NET, What, Why and When plus a report on the business background to business process automation. Richard takes a look at how the service based economy will develop and who the dominant players will be providing the business services.

David Sprott *david.sprott@cbdiforum.com*

1. Microsoft's .NET, What, Why and When, INTERACT July/August 2000

© cbdiforum.com Limited, July/August 2000

Microsoft .NET, What, Why and When?

You have probably noticed Microsoft's recent big announcement on their .NET strategy. If this passed you by, visit www.microsoft.com/net first, as I don't intend to repeat the basics in terms of products and features. Rather I will focus on those aspects I think more relevant for some CBDi Forum analysis.

The .NET Vision

A quick recap on the vision is however useful. Mainly to add some clarity, as the announcements seems to have been widely misunderstood. In fact we observe huge confusion in the media generally about what .NET really is. For some it is Microsoft's attack on the Java world. For others it is a five-year vision of remote service provision, for others it is replacing COM with XML based services.

Many aspects of .NET are not really that new in terms of how vendors, commentators, even science fiction writers, have envisioned what capabilities would be available if we had a fully wired world.

What Microsoft has done is to pull together the various strands of this and explain how they are delivering a platform that enables it. The vision is less revolutionary than Microsoft have claimed. But that's no bad thing - ultimately most business and development managers are risk averse and would be unlikely to invest in something too far removed from their existing comfort zone. Core aspects of the vision are:

- Everything is connected via the web. Everything can be connected thanks to transparency between the layers, via technology like XML and SOAP.
- The web becomes a source of services, not just data. Services can support the individual, run a business, or be assembled into more complex applications.
- Services, whether provided by the platform, by packaged software, or business applications, could be hosted on any layer, in-house or by 3rd parties, and their host changed transparently to the consumer of the service, as illustrated in figure 1.

There are three core areas to the products that Microsoft will deliver as part of the. NET vision:

- **.NET Execution platform** - The .NET enabled versions of the operating systems and other core components of the platform provide the .NET technology infrastructure.
- **.NET Development Tools** - a wide variety of .NET enabled tools to build components and the web services they provide, and to assemble applications from them.
- **.NET Services** - Ranging from Microsoft software such as Office delivered in a .NET edition and available over the web as a service, through 'building block' services such as calendar or identity, to Microsoft supplying web-based business services of a complete application such as CRM.

Endorses CBDI approach

The CBDi Forum has good reason to be supportive of Microsoft's new .NET vision. Not because we are Microsoft zealots, rather because .NET is very supportive of the CBDi vision we have painted over the last couple of years.

It is unfortunate perhaps that each time Microsoft enhances their vision of distributed computing that they choose to re-label it. This gets interpreted as requiring a whole generation of new software before the vision is anything other than vapourware. Sure the .NET versions of just about every piece of Microsoft software that they announced will improve the ability to deliver on the .NET vision. However, the reality is that many aspects of the vision can be implemented using today's technology, and importantly there is nothing to stop users getting onboard and delivering .NET style applications, or at least preparing themselves now, for .NET is as much about adopting the right mindset and approach as it is about future releases of software.

As we have often commented about CBD, it is much more important to get the fundamentals of establishing a component architecture, putting the right project organization and funding in place, and ensuring everyone involved has the right mindset, than it is making a technology selection for the implementation stage (which keeps evolving anyway). Success is measured on how well you meet the needs of the business, not on how many features of the latest technology you use.

© cbdiforum.com Limited, July/August 2000

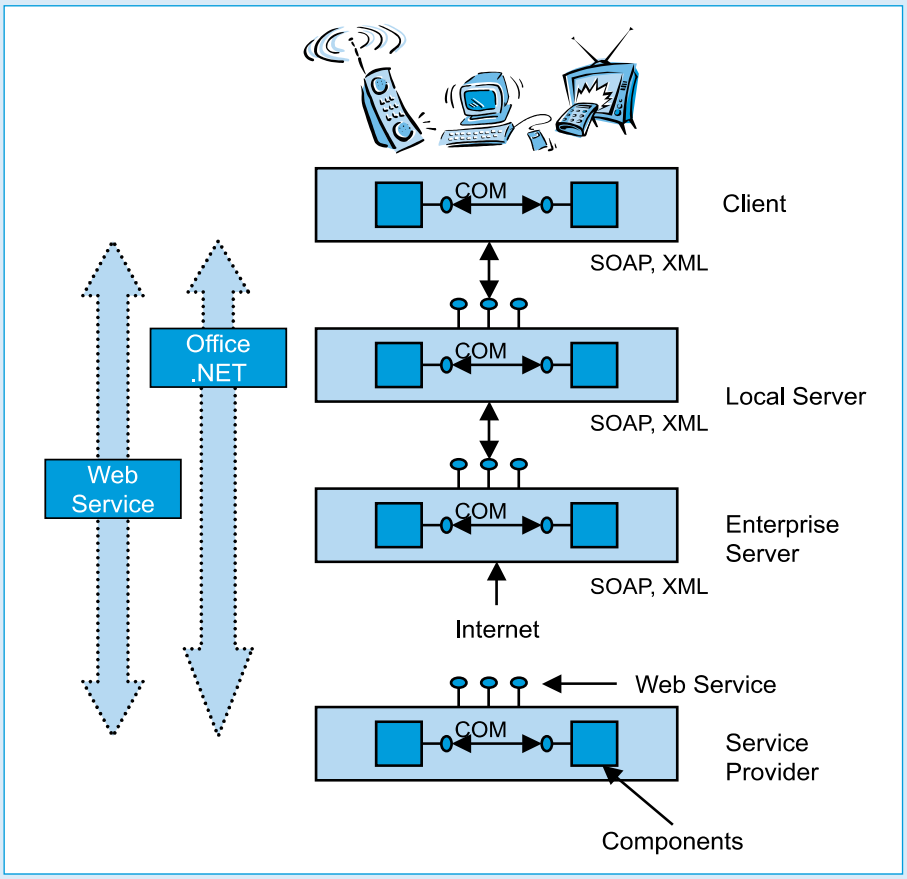

Figure 1: .NET Services Move Across Tiers

A key challenge for Microsoft (and other technology vendors) will be to ensure that developers are building component architectures that deliver lasting value to the business (in return for their investment), not just using the technology because its there.

Of course, getting the last drop of performance out of the platform may require you to use many of the latest features. And .NET will make those features more productive and easier to use. Just don't lose sight of the need to support your business!

Service Based Development

The cornerstone to success with .NET is going to be the adoption of a service based development approach.[1] Microsoft's .NET vision is a wide range of business and technical services that are provided across the Internet - what they term Web Services - and are assembled into a custom applications. This will require developers to focus much more on the services provided by a component than on its implementation. With Web Services there is no software for the consumer to implement, as the service provider has already done that work.

The availability of a complete range of Web Services to meet all needs will no doubt take time, and as such the majority of development activity will still be focused on building custom implementations, or bringing purchased components in house. We believe it is critical that development projects focus first on service provision and the agreement of 'contracts' between applications and the components they consume, and they ensure that flexibility is enabled by not building links directly to internals of the current physical implementations. I.e. by using interfaces correctly.

Microsoft .NET means that the assembly of applications will be based on schemas and contracts, not on than IDL or SQL or other technology specific APIs, and I believe this more complete availability of the CBD approach will finally help developers understand that the proper use of interfaces is not just about calling subroutines!

However, there are several requirements for making service based development work that must be implemented within the technology layer. How do you find the services, or what precisely they do and the contract for using them? How do you implement the contract, and ensure adherence to it?

SOAP

On the technology side, it is increasingly clear that SOAP (Simple Object Access Protocol) and further developments of it will be a key part of enabling the .NET vision.[2] The good thing about SOAP is that as it is just a protocol, it does not require any new technology, ORB or API. Microsoft has committed to supporting SOAP in COM, and so have several CORBA vendors. As such, SOAP is a very attractive solution to the need to integrate services running on a range of heterogeneous technology.

As well as using SOAP to enable the connection of services, Microsoft is proposing the SOAP Contract Language (SCL), an XML format for describing contracts between the service and its consumer, and Service Discovery Language.[3]

We predicted last year that a more loosely coupled approach to application assembly/integration based on

continues...

© cbdiforum.com Limited, July/August 2000

'Microsoft .NET, What, Why and When?' continued...

XML-RPC (from which SOAP evolved) could gain favour over more complex component technologies such as DCOM, especially if taken up by Microsoft.[4] What is surprising is how quickly this idea has now taken hold, but also that it has been endorsed by IBM, HP, Sun and others. Though SOAP doesn't stop any COM+ vs EJB vs CORBA technology wars, as it still requires one of these to implement the applications it connects, it will substantially reduce arguments about the need to standardise on one or the other in order to achieve interoperability.

Web Services

Web Services might be best thought of as two strategies for Microsoft. On one hand it is the delivery of technologies that enable developers to implement interfaces to their components as Web Services, to make them easier to access and assemble into applications. On the other, it is the delivery of Microsoft products as Web Services themselves.

Microsoft's intentions regarding its own role as a Web Service provider are not fully clear. Initially, they will offer a range of 'building block' services that can be implemented in house, but also invoked across the web as services. Today this features a range of business generic and technical services covering subjects like Identity, Personalization, Calendar, Notification and Messaging.

I would expect this list to grow. Whilst one would expect Microsoft to provide a range of technical and generic Web Services, it is interesting that they also plan to introduce more business oriented services, for example adding Web Services to their bCentral.com web site that is aimed at small businesses, by including CRM and product catalog functions.

It is very important to realise that the intention of many of these services is that not only is the functionality of the service executed transparently 'somewhere' on the web, but that also the information and data you entrust to it is also stored 'somewhere' on the web too. Many organizations will doubtless be wary of this for some time due to security and access concerns.

It is our belief that the provision of Web Services will become increasingly prevalent, though this will require a stepwise improvement in security etc, for developers and business alike to be convinced. Microsoft deny they have any plans to host applications, but any expansion on the above list will ultimately bring them into direct competition with ASP and traditional package vendors.

It is Web Services and the SOAP technology that has led some observers to claim that COM is replaced. I believe the two are complementary, at least for some time as illustrated in Figure 1. COM will still be used to tightly couple together fine-grained components that reside close together. However, using DCOM to connect distributed components is bound to give way to SOAP, where a looser coupling is desirable, and necessary when heterogeneous technologies are concerned.

Putting aside issues of who should supply them, from a technical perspective Microsoft's SOAP toolkit makes it easy to build and consume Web Services. In fact, such is the ease that it reinforces in my mind the need to think carefully about what is being published and why, because the how is too simple.

I have visions of developers delivering a plethora of Web Services simply because they can, without much regard to the business process that ensures they are executed in the correct manner to ensure business integrity. Whilst CBD encourages the separation of building and publishing services from their consumption, the danger is that Microsoft's new technologies make it so simple that applications are assembled too fast, RAD style without regard to the complete business process.

Subscription

Microsoft believes that the services they and 3rd parties provide will be paid for on a subscription basis (where charges are applicable). As well as the aforementioned Web Services, Microsoft will also provide subscription-based usage of Office.NET, the next generation of their Office suite.

The industry has talked for some time about subscription based approach to software acquisition, but this has so far been a coarse grained approach such as ASP's hosting ERP or similar packages. Microsoft's .NET vision takes us a step closer to subscribing to finer-grained services, and on perhaps a more ad-hoc basis than entering into monthly contracts.

However, for this vision to succeed there needs to be parallel improvements in security, authentication, the micro-billing systems and ways of monitoring service level agreements.

Orchestration - Workflow Driven Application Integration and Assembly

The one new piece of software that Microsoft did preview at the same time as the .NET launch was BizTalk Orchestration. This adds business process

© cbdiforum.com Limited, June 2000

automation, or workflow to BizTalk Server 2000. It takes the form of a Visio based development environment for the process definition and assembly, which generates an XML schema that is executed by a new 'state engine' running in BizTalk Server.

This is the most business-oriented piece of development software I have seen so far from Microsoft. The intention is that business users, already familiar with using Visio or similar products to draw process diagrams can now simply add to those diagrams the links to the actual applications/components that provide the required services and generate the workflow that automates the process. This fits in well with the CBD concept of supplier/consumer split of development, as it separates the assembly process out from that of delivering the components, and takes a step towards enabling business users to assemble applications without recourse to professional developers.

Orchestration is quite capable of integrating services from COM, MSMQ, SOAP, and includes a wide range of adaptors for other packages and technologies, building as it does on BizTalk Server's existing capabilities in this area.

Again the scope and role of Orchestration fits well in our CBDi approach. See my recent report 'Application Connection for E-Business'[5] that discusses the need to both assemble applications from components via the tight coupling of component technologies, and also integrate them in a wide-area workflow with more loosely coupled approaches. Orchestration also fits in well with our recommendation to isolate the

continues...

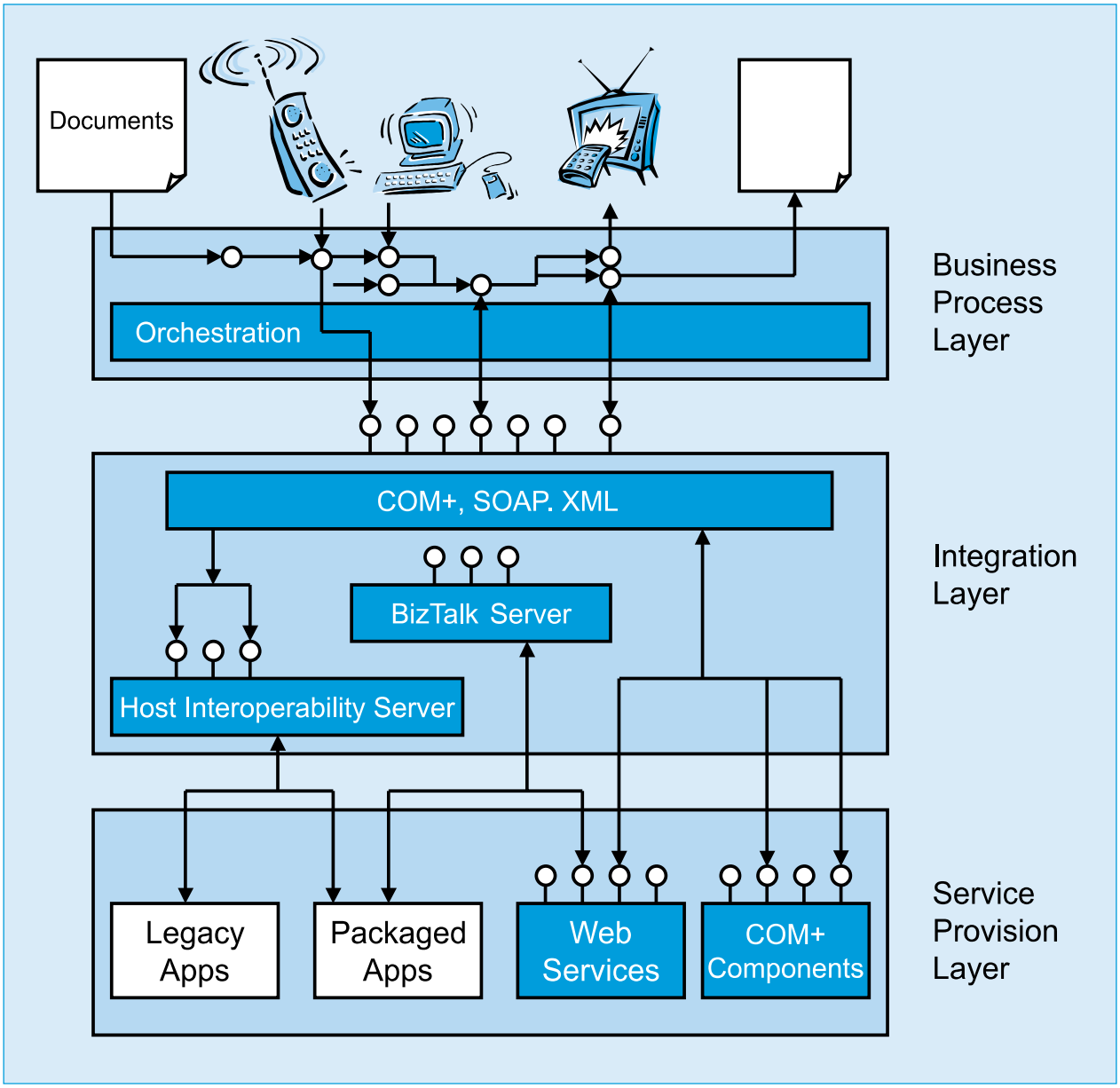

Figure 2: Mapping DNA and .NET to CBDi Application Layers

© cbdiforum.com Limited, June 2000

'Microsoft .NET, What, Why and When?' continued...

workflow/business process as a separate layer independent of presentation, business or data objects, as it focuses attention on just defining the business process in isolation from the actual implementation of the other layers.[6]

C# (sharp)

I am not so encouraged by C#, their new object-oriented programming language. C# is a development of C and C++ that adds Java-like capabilities and platform independence, together with ease of use more akin to Visual Basic than C programming. It also has a Common Language Runtime engine that can support multiple development languages and potentially operating system independence if Microsoft, (or Microsoft's partners) provide an implementation on other platforms.

However, I question just how many different programming languages and variations we need? I am sure that developers in Microsoft environment will find the new features useful and hence it will gain a following, but I think they and Microsoft will be missing the point if they believe C# is what is going to make it easy for businesses to build a new generation of web applications. For the record, I must say I felt pretty much the same about Java, which was over hyped as a solution to all IT's problems.

For me, Orchestration seems closer to the business need of delivering applications as rapidly as possible. The CBD process does of course enable developers to use the appropriate tools for each role, and as such a language such as C# clearly has its place. But Microsoft need to be careful about placing too much emphasis on C#, as it will enable their detractors to drag them down into a C# vs. Java debate, and overshadow the rest of the otherwise sound .NET strategy. If the .NET vision is successful, we will have far fewer developers writing code from scratch anyway.

Conclusions

As can be seen in Figure 2, Microsoft .NET maps comprehensively to our CBDi application layers. Microsoft .NET provides a strong vision for how e-business applications will be delivered in future. The fact that some of Microsoft's rivals are also labelling it a 'me too' strategy only re-enforces the strength of the vision, irrespective of whether their claims are valid or not. The fact is, that e-business is forcing vendors down a convergence path and therefore it is not surprising that their strategies should appear similar as there aren't too many options when you take an objective view of what is required, at least for the foreseeable future. However, their implementations are allowed to differ and this is where competition will occur, not by convincing users to have a different vision.

Given earlier industry debates on how much functionality should be in the client layer it is interesting to observe that Microsoft have now seemingly fully embraced the network computing vision. What is key however is that Microsoft is enabling a transition strategy that allows users to move functionality transparently between the client/server/web layers more easily than before. This was something that was missing from competitors' thin client strategies that required users to take a more rigid choice.

I do think .NET has been misunderstood by some, e.g. described by at least one commentator as a requiring Microsoft to achieve the same level of domination on the server as it has at the desktop for it to work. I see entirely the reverse. Yes, Microsoft would like everyone to take their implementation of the vision, but capabilities such as SOAP, XML, Web Services, the BizTalk Server, the Host Interoperability Server are recognition by Microsoft that it lives in a heterogeneous world where interoperability is more desirable than standardisation on a single technology. Sure it might be a Trojan horse, but that is for Microsoft's competitors to worry about. I.e., they must deliver an equal or better implementation of the e-business vision than Microsoft to survive.

As I said earlier there is no need to wait for .NET specific software before getting on track. The key to success will be adoption of the approach, not a specific release of the technology. A shift towards programming applications with greater emphasis on messages, events and more business-oriented 'orchestration', rather than objects and methods may require re-education of many developers. Thinking services is not just using APIs. As such, organizations should seek out training in CBD methods and approach, not just .NET technologies.

Lawrence Wilkes *lawrence.wilkes@cbdiforum.com*

1. Service Based Development, INTERACT, December 1999
2. SOAP Specification - *http://msdn.microsoft.com/xml/general/soapspec.asp*
3. Download draft specification at *http://msdn.microsoft.com/net/scl.zip*
4. Application Connection, INTERACT, September 1999.
5. Application Connection for E-business, CBDi Forum Report, June 2000
6. Four Tier Architecture - CBD in the Windows DNA Environment, CBDi Forum Report, June 2000.

© cbdiforum.com Limited, July/August 2000

Are you being served?

Who will be the leading players in the service-based economy?

In this month's editorial, David asks whether the ASPs will survive (perhaps under new labels) or whether they will disappear as the new players offer something more attractive. Who will be the dominant service providers: computer manufacturers, software platform providers, ERP vendors or telecoms companies? In this report, Richard explores these questions in a little more detail.

The Growth of Software Rental - Back to the Bureaux?

Over the past year or so, there have been many attempts, with differing degrees of success, to create a market for Application Service Provision (ASP). These attempts have been triggered by various considerations.

On the supply side, several suppliers of large expensive software packages, such as ERP, have found it increasingly difficult to find new customers prepared to undergo large and expensive package implementations. These suppliers recognized that if they could make their products available in the form of outsourced application services, this would substantially expand the number of potential customers. Assuming reasonable levels of customer retention, this approach would hopefully also provide a much longer and consistent revenue stream.

And when the package suppliers themselves are not willing to enter the ASP market, third parties are clearly ready to take up the opportunity, provided that they can negotiate mutually beneficial licensing or OEM arrangements with the software copyright owners.

On the demand side, many purchasing organizations are willing to pay a significantly larger cost over the expected lifetime of product use, rather than incur the bulk of the costs as an initial investment. This approach seems to shift much of the risk onto the supplier, especially if an organization can discontinue using the service at relatively short notice.

Case Study: Fidessa

The Fidessa application package is used by banks for equity trading. Major investment banks typically install the software at their own site, but smaller banks can rent the software over a network, thus saving a large initial investment.

Royalblue, the maker of Fidessa as well as other financial and helpdesk software, has increased sales by 42% in the past six months, mainly as a result of the trend towards software rental, which has helped it tap into new markets.

It is natural to compare ASP with the computer bureau services that proliferated in the 1960s and 1970s. Bearing in mind the performance difficulties experienced with many of these bureaux, some analysts have issued warnings about the capacity of the new ASPs and their ability to deliver required service levels. In fact, although it's clearly important to specify, negotiate and monitor service levels, there is no reason to suppose that an external service provider would be any less powerful or reliable than an inhouse solution. Hardware is surely no longer the major cost factor, and the ASPs are often closer to the Internet hubs than their customers. And for web-based services, where demand levels may fluctuate enormously, a specialist service provider may be better able to procure extra capacity at short notice.

(This highlights one of the key differences between applications and services. Services make use of shared resource that remain in the ownership and control of the provider, not the user. Therefore ownership protection and capacity planning need to be considered carefully when making service engineering decisions - indeed, these considerations can affect the order in which the engineering decisions should be made.)

A much more serious inhibitor for the ASP market has been the clumsy footprint of many of the services, whose scope - if not the actual software - is typically based on the legacy of an existing application package. Many user organizations have experienced difficulties with wide area integration, even with the support of the EAI vendors, and this is one of the reasons why ASPs have so far failed to dominate the market. (It's not the only reason, as we shall see later.)

In some sectors of the software market, however, service provision has been extremely successful, especially where the service has been tightly scoped. Even if the

continues...

© cbdiforum.com Limited, July/August 2000

'Are you being served?' continued...

ASPs manage to reinvent themselves, the flexibility and choice they can offer may remain markedly inferior to anyone who can offer independent sets of services.

Case Study: SCM Dialtone

SCMdialtone is an Internet-based software service that exposes business components for real-time supply chain management.

Behind SCMdialtone is a collection of business objects that do everything from managing customers and item part numbers to calculating (in real time) the production and procurement plans that will replenish stock based on the day's sales. These business objects are invoked on the service host using the simple object access protocol (SOAP).

Central to the SCMdialtone business model is the notion of transaction-based billing ("microbilling"). The service is designed to support very high volumes of low-cost transactions, via a network of franchised datacentres.

SCMdialtone is targeted at vertical B2B hubs and system integrators, as well as end-user organizations.

Back Office

A number of financial institutions have been looking at Back Office as a potential ASP or business service. Either they might outsource Back Office to someone else, or they might provide Back Office as a service to other finance companies. However, it turns out that Back Office is not a meaningful service in its own right, but may be a collection of loosely coupled services, provided by multiple companies.

If we look at this more closely, we find that for many organizations in the finance sector the interface between "front office" and "back office" dates back to a time before widespread computing. The back office was established to manage vast quantities of paper-based transactions and information that could not be processed "in real time". This front/back interface was then replicated in the design of computer systems: front offices were provided with fast information displays to support trading and suchlike functions, while back offices were provided with efficient information capture and control suites to support administrative functions such as billing. This interface is strongly embedded, not only in the IT architectures, but also in the organizational structures and cultures of these companies, and is extremely difficult to change.

In these organizations, people often talk about "The Back Office" as if this term had the same meaning everywhere, in all financial organizations, equally for paper-based and electronically mediated processing. Back Office is apparently the name of a universal service, and our archetypal finance company might consider sharing a single Back Office between several Front Offices, outsourcing Back Office to a generic Back Office service provider, or becoming a Back Office provider to other companies.

But as soon as you look at the detail, the interface between Front Office and Back Office turns out to vary significantly from place to place. It's certainly possible to share a single Back Office between two Front Offices, or conversely to distribute the support for a single Front Office between two Back Offices, but it's by no means a trivial exercise.

Thus instead of regarding Back Office as the name of a service provided to Front Offices across a well-defined interface, it may be more appropriate to regard it as the umbrella name for a loosely packaged set of services, which may be assembled from different sources.

Telecoms traditionally maintained a similar division - the Back Office was split off from the technologically important domain of delivering connectivity, and included such lesser functions as support, administration, billing and management systems. However, since the customer relationship management sits with the Back Office functions, the emphasis has now been reversed, and it is often the Back Office systems that are now regarded as strategic. Companies such as Telcordia (formerly Bellcore) provide a range of business services (aka components) in this area.

B2B Exchanges

Early B2B initiatives were often based on naïve assumptions. Ambitious start-ups claiming that they could save billions in procurement costs for big corporates, taking a large cut of the savings for themselves. Of course, the big corporates quickly realized that they could do this for themselves, and were not going to hand over the savings to third parties. Many of the early initiatives have been dominated by the big corporates, although there remains a role for independent third parties to provide well-focused B2B services, based on specific technical and business capabilities.

Enter a new player: the regulator. B2B marketplaces may sometimes increase competition, cutting transaction

© cbdiforum.com Limited, July/August 2000

costs, reducing barriers to entry, and leading to increased choice, reduced prices and better inventory management. But when dominated by a cartel of big players, it can be an anti-competitive force. US regulators are reviewing several new ventures, notably Covisint, the auto exchange formed by DaimlerChrysler, Ford and General Motors, supported by Oracle and CommerceOne. Architectural and technological issues - such as global integration and market transparency - will overlap with regulatory issues, and this may create additional roles for smart and independent service providers.

New Roles

In general, technological progress typically results in an increase in the number of separate players, and a creation of new niche roles. This is particularly evident in the telecommunications sector, thanks to a combination of deregulation and other factors to which we shall return below. Another widely studied example of role proliferation is the introduction of electronic tax filing in the United States (see box).

Case Study: Electronic Tax Filing

Since 1990, US taxpayers have been able to submit their tax returns electronically.

The introduction of electronic tax filing in the US has resulted in a considerable increase in the number of separately identified roles within the system - some necessary for the electronic system to work at all, and some new commercial and professional opportunities generated by the system.

Prior roles	• tax payer • return preparer • mail carrier • banking service • Inland Revenue Service • retailer
Required new roles	• electronic filer • communications network provider • software vendor
Additional opportunities	• information broker • consumer credit provider • tax planner • investment banker

Of course, a proliferation of roles doesn't necessarily mean a proliferation in the number of independent players within a market, as a large organization can take a strategic decision to play multiple roles, or to enter strategic alliances with partners playing complementary roles. However, even when many roles are played by a single organization, there is still a strong business argument as well as increasing technical support for configuring this organization as a federation of explicitly collaborating service providers, rather than as an old-fashioned tightly coupled monolith.

Business Services

A common reaction to the concept of web service based architectures is to be deeply concerned about operational integrity. It is primarily for this reason that perhaps the majority of media commentators are advising that web services will not become pervasive for three to five years. However this perspective is based on an extrapolation of current practice. In the future, outsourced services are more likely to be business services or software services, but not application services. Software services will rapidly become pervasive in the same timeframe as the current PC model of computing transitions to the web based model, and we rent our usage of whatever personal productivity software we choose to use. Business services will be strongly favored over application services because this places the commercial costs and risks together with the operational responsibility and overcomes the operational issues.

Suppose I represent an insurance company, and I use a component-based service from another company to help me perform the underwriting. Don't I need to know the algorithm that the other company is using? Suppose that the algorithm is based on factors that I don't believe in, such as astrology? Suppose that the algorithm neglects factors that I believe to be important, such as genetics or genomics? Am I not accepting a huge risk by allowing another company to define an algorithm that is central to my business?

There are three main attitudes to this risk. One attitude, commonly found among civil servants, lawyers and software engineers, is to break encapsulation, crawl all over the algorithm in advance, and spend months testing the algorithm across a large database of test cases. If and when the algorithm is finally accepted and installed, such people will insist on proper authorization (with extensive retesting) before the smallest detail of the algorithm can be changed. The second attitude is denial: impatient businessmen and politicians simply ignore the warnings and delays of the first group.

continues...

© cbdiforum.com Limited, July/August 2000

'Are you being served?' continued...

There is a third approach: which is to use the forces of competition as a quality control mechanism. Instead of insisting that we find and maintain a single perfect algorithm, or kidding ourselves that we've already achieved this, we deliberately build a system that sets up several algorithms for competitive field-testing, a system that is sufficiently robust to withstand failure of any one algorithm.

The most direct mechanism is a straight commercial one. If the company operating the underwriting algorithm also bears all or some of the underwriting risk, then its commercial success should be directly linked to the "correctness" of the algorithm.

Where this kind of direct mechanism is not available, then we're looking for feedback mechanisms that simulate this, as closely as possible. Just as the survival of the company using these underwriting services may depend on having access to several competing services, so the survival of the underwriting services themselves may depend on being used by several different insurance companies, with different customer profiles and success criteria. (This reduces the risk that all the customers for your service disappear at the same time, and gives you a better chance to fix problems.)

The bottom line is that I'm buying an underwriting service rather than an underwriting calculation service, a business service rather than an application service. ASPs will have to rebrand themselves yet again, and present themselves as genuine business process outsourcers.

This relates to my own contention (to be explored in my forthcoming book on the Component-Based Business) that we should focus on business components that deliver business services, rather than merely information or application services.

Case Study: Alterna Technologies

Alterna Technologies is a Canadian company providing corporate treasury services via a hosted virtual shared service centre in Dublin. Bulk payments are formatted on behalf of customers, and sent to one or more banks for processing.

Benefits to customers include the ability to use the web to access information about their transactions. The service also offers a degree of independence from regular banking providers.

Role of Telecommunications Companies

While much of the attention about component-based development has focused upon the activity of traditional IT companies, a fair share of the original thinking stems from the telecommunications world. The telecoms companies have done a lot of the groundwork for the service-based economy, and may soon be taking the leadership of CBDi away from the triumvirate of Microsoft, IBM and Sun.

Many years ago, some computer visionaries preached the convergence of computing and telecoms, to form a composite subject to be called IT (information technology). For a long time, however, the computer side of the equation dominated the field, IT became synonymous with computing, and the telecoms side was forgotten. (More recently, some purists have tried to reassert this convergence through the acronym ICT - information and communication technology - but this has failed to take hold except in some academic and government circles.)

Once upon a time, computer entrepreneurs grew rich developing spin-offs from HP or crumbs from IBM's table. And, for a while, it seemed as if every new idea in the IT world could be traced back to some work at Xerox Labs.

But let's look at what was happening behind the scenes during the 1990s. The World Wide Web appeared, apparently from nowhere. Concepts such as distribution and federation are widely used, in business processes as well as technology. CORBA, while now relegated to a niche function and bypassed by other products and proprietary standards, nonetheless triggered a significant mindset change in distributed software engineering. These weren't invented by Microsoft or Sun, nor even IBM and Xerox. It was in many instances it was the telecoms companies that funded this work. Now it's Bell Labs (part of Lucent) that everyone seems to wants to steal ideas from.

Policy Management

As an example of how telecoms is leading other industries in managing an enterprise as a network of services, let's look at policy management. Telecoms uses software intelligence in the form of a policy server to manage network services using rules. Policy servers understand the rules by which the network manager describes the behavior of the network, the topology of the network, and the behavior of the devices that

© cbdiforum.com Limited, July/August 2000

constitute the network. Using business-related information from the corporate information database, the policy server then configures the devices in the network so that the desired behavior can be implemented.

Much of this network management is currently done at a fairly low technical level, below the level of the application. Not surprisingly, there are products specializing in telecoms-specific domains, such as Lucent's DEFINITY software that implement policies for telephony. However, telecoms companies are now turning their attention to higher application and business process levels: Lucent also has a product called CentreVu CRM that implements policies for customer relationship management, and has declared an interest in using policies to manage an enterprise's e-business applications.

Strategic Considerations

Business processes are traditionally understood as value chains; the process is decomposed into a series of steps, each of which adds some value to the whole. Some methods of business process analysis go slightly further than this, and divide processes into value-adding and enabling. Many early outsourcing initiatives focused on the so-called enabling processes; however, modern business process automation typically also considers outsourcing of the value-adding steps of the process.

This approach works reasonably well in slow-moving businesses, where the value chain doesn't change very much. For many industry sectors, even the conversion of a business process into an e-business process may not radically affect the way the value chain is understood. An Internet bank receives deposits and makes payments, charges or pays interest, pretty much like a traditional bank. Even when the process goes wrong, it still goes wrong in recognizable ways.

In these businesses, the likely services will correspond to the smallest meaningful steps in the value chain - you may wish to regard these as business objects or elementary processes, depending on your background. The likely service providers will be those companies that control the relevant resources and can deploy the necessary capabilities most cost-effectively across the largest domain.

However, in high technology and high service industries, including telecoms, the basis of competition is rather different. We have already seen how the strategic emphasis in telecoms has been reversed: from connectivity to customer relationship management. Service providers are not competing to provide specific services within a fixed and well-understood business process, but competing to exert design control over the way the business process itself is configured. This leads to a new approach to business strategy, based on value ladders instead of value chains, which is being pioneered by Boxer Research Limited. Boxer identifies three kinds of service offering, as shown in the following table.

Three kinds of service offering

The **c-type** service offering supplies 'off the shelf' services. The supplier has no involvement in the decomposition of the client's needs. The supplier's concern is focused on its own operational effectiveness with respect to those services for which it has identified a ' market'. What is sold is the ability to supply a service.

The **K-type** service offering is configured to deliver services in a form that is appropriate to the demand situation in which it believes its customer to be. The client defines the need in the form of a demand, but the supplier decomposes that demand - the problem - into a set of (better) satisfiable service specifications. What is 'sold' is the ability to decompose the problem into a form which renders it (more) tractable.

The **P-type** service offering, which collaborates with a potential service customer in the specification of the latter's need as a demand, before (jointly) decomposing it into the form of services which itself or others can supply. What is 'sold' is the ability to articulate need in the form a demand for services which are believed to be satisfiable, in other words to define a problem.

Source: Boxer Research Limited

We started with the question: Who will be the dominant service providers: computer manufacturers, ERP vendors or telecoms companies?

The telcos started exploring the notion of the service-based economy long before anybody else, and are further advanced in many areas than their IT rivals, giving them a considerable advantage.

The telcos have a long way to go, of course, to consolidate and exploit this advantage. They could still throw it all away, by failing to disseminate the ideas and paradigm shifts through their own organizations, by allowing technological considerations to dominate over business ones, or by failing to discriminate genuine opportunity. But at present the tide is running in their favour.

continues...

© cbdiforum.com Limited, July/August 2000

'Are you being served?' continued...

Summary

We continue to predict an expansion of services and service providers. The indicators are that the market is now ready for service providers, both large and small, from a range of backgrounds, to make a lot of money providing services. At the same time, procuring organizations should be able to save a lot of money. The critical success factors for both providers and procurers are the scoping and integration of these services. These factors will continue to deserve our attention as the market develops.

It is high probability that value adding business services will become a major market and the dominant providers will be those organizations that can outsource the commercial responsibility. Increasingly the platform providers and software products organizations will perform the role of providing the components, infrastructures and frameworks and the deployed application and business service will be managed by vertical specialists such as the telecos and banks.

> *"The dominant providers will be those organizations that can outsource the commercial responsibility. The business service will be managed by vertical specialists such as telecos and the banks"*

Richard Veryard *richard.veryard@cbdiforum.com*

Richard's book on the Component-Based Business will be published in November by Springer Verlag.

Discuss this report on the CBDi Forum Discussion Forum Components in Business

Why Automate the Business Process?

It looks like a workflow-based approach is set to take a more central role in application delivery. After many years on the sidelines, like components, it seems that workflow's time has finally come. Why is this? In this backgrounder report we examine the business circumstances and problems that commonly provide a strong case for workflow like functionality.

My Call Centre Nightmares

Like me, I expect you spend more and more time as a consumer talking on the phone to the companies you buy products and services from. Whatever the reasons, such as closures of branch networks, call centres now take care of a large proportion of our person-to-person interactions with many companies.

Many application delivery projects of late have been focused on providing the call centre operative with access to a wide range of systems. Typical scenarios are putting a single UI across many diverse back-end systems, often unifying the business process that runs through them. EAI products are commonly used, linking together CRM, ERP and legacy applications.

However, in my experience this has not resolved the frustration I often find in dealing with call centres. When they only have to follow the script, no doubt the availability of all the different systems to the operator pays great dividends in terms of productivity. The trouble is, often my only reason for calling them is because the script has failed. "Where is my shipment? Why have you charged me this? This is not what I asked for. Etc, etc," seem common opening gambits.

© cbdiforum.com Limited, July/August 2000

The worst answer you can get is “we will have to phone you back”, as you know they rarely will, and it inevitably leads you into follow-on conversations that are mainly taken up trying to ascertain who you were dealing with the previous time. Having been recently asked, “who did you talk to”, at the end of the conversation I asked the operator for his name so I would be prepared next time. “Sorry, we are not allowed to give out our names” was the reply! Or try explaining that you have no idea what department you spoke to as all you did was call their free phone number and respond to the automated options.

Applications that have been delivered by traditional approaches, either built in-house or acquired packages, record the basic information about a transaction (the order, customer, invoice, type of data) but typically record little about the process by which it was made. I.e. who did what and when, or what task is next, and who should perform it?

Workflow Management Systems (WMS) can provide a ready built solution to this by not only defining the process, but also recording the current state of any instance, so we get the who and when, not just the what.

WMS have also historically evolved from document management requirements. However, in this report I will focus more on the need to support business process automation.

Wide Area Workflow

I feel some sympathy for the call centre operator though, as of course they have little to do with the actual execution of the end-to end business process they have initiated and are your interface to. I know things are about to take a turn for the worse when the call centre tells me to phone another number - a good indication that they don’t actually have sight of the whole process. I recently played ping-pong with the call centre and the logistics department of a major UK PLC. Whilst I simply tried to find out the status of an overdue shipment, they argued with each other over the validity of the order number I had. It wasn’t that my order number was just wrong, but according to the logistics department it was entirely the wrong format and could not be one of theirs, but the call centre was adamant it was correct. And so I called each in turn trying to find out the status of my order. The logistics department could only look at their shipment system, so without a valid order number they were unable to look me up as customer and trace it from that end. It turned out I had been given an internal dispatch note number, not an external customer order number, though I don’t know why they didn’t spot this earlier and tie the two together. Put it down to lack of operator experience.

In this case the call centre and logistics department were in two different cities, a not unusual situation, and their switchboard was seemingly unable to patch me from one to the other forcing me to call each of them in turn. And with greater outsourcing of logistics, they could easily have been two different business entities. As such, organizations need to be able to record and have sight of the who, what and when of the whole business process throughout not just in their own departments, but also of their suppliers and partners. The term Wide Area Workflow seems an apt description for this requirement.[1]

The whole e-business scenario will clearly increase the need for this. However, it is not just the application of messaging or workflow technology that will resolve this requirement, but more the need for the various members of the virtual supply chain to work together to develop a collaborative business process. Again many current e-business technology solutions seem focused on delivery of the information that needs to be exchanged between partners, but not on managing the state of the whole business transaction that is now being performed by potentially many people in many organizations.

The Workflow Management Coalition (WfMC), a group of vendors and users,[2] has produced a number of standards including interoperability between workflow systems to enable them to manage a business process that spans different organisations. This enables the scenario shown at the top of Figure 3.

However, for reasons I have explained in a related report last year[3] (e.g. duplicated rather than shared information), I believe this is sub-optimal. The technology is now being put in place to enable the more collaborative approach shown at the bottom of Figure 3. The web services concept together with SOAP will enable organizations to make information available to all participants in a shared business process without having to duplicate it.[4] And with workflow vendors planning support for SOAP, then there is no reason why participants should not share their workflow too.

continues...

© cbdiforum.com Limited, July/August 2000

'Why Automate the Business Process?' continued...

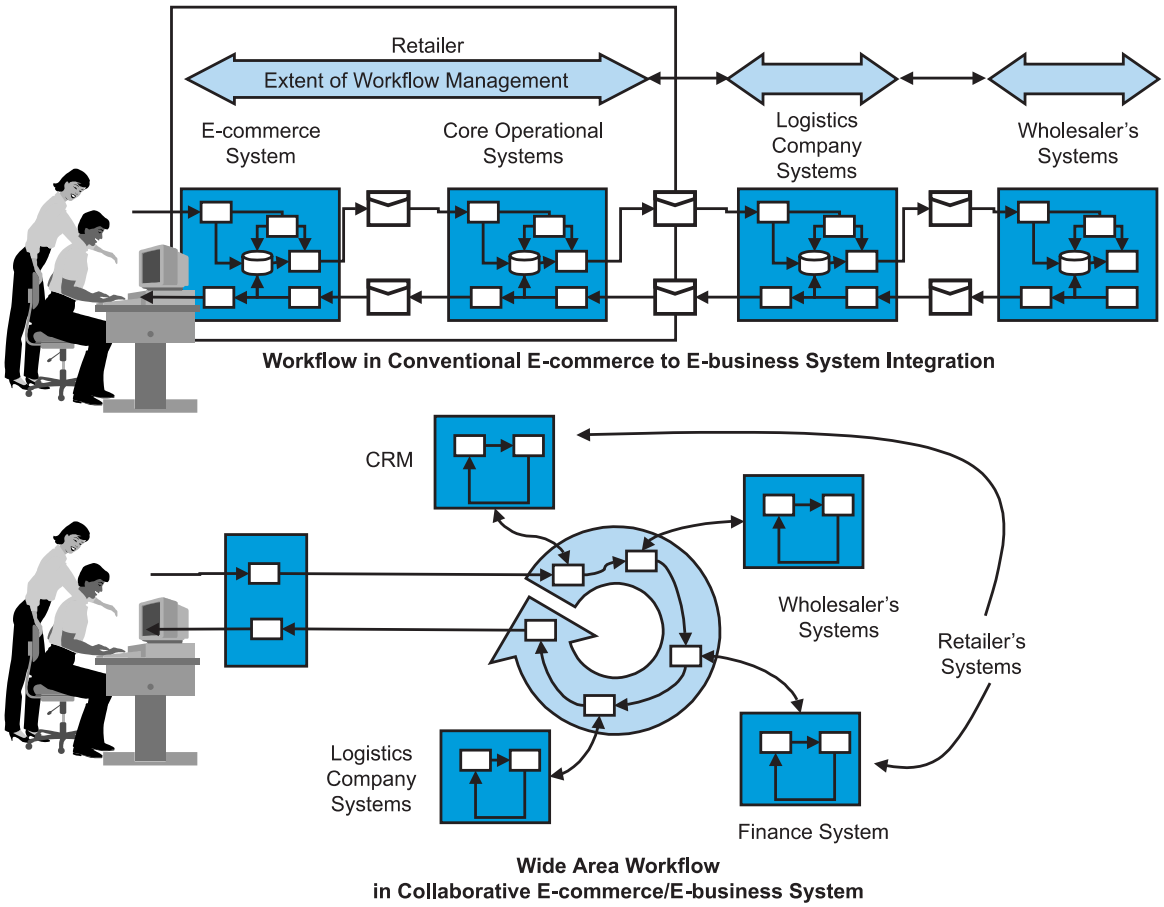

Figure 3: Wide Area Workflow

This would give all participants full access to the status of any particular task (access rights permitting). So the retailer would not just know that the logistics company had been send the dispatch note, but also who's work list it was on, when it has been picked and packed, and by whom, making any potential problem resolution easier.

The main issue here would be one of organizational politics though, not technology. Who owns and manages the workflow? Should a single instance be shared or should each organization retain its own workflow engine? Either way, the growth of e-business trading hubs and communities will increase the need for participants to consider not just how they exchange information but how they collaborate in their business processes, as the two are intertwined.

Wide Area Business Transactions

What naturally follows in this discussion is the need to address transaction management across a number of different participants. OLTP techniques have been the mainstay of ensuring data integrity in business applications for some years. Various incarnations of two-phase commit allow distributed transactions to be managed, ensuring that changes in information across multiple databases can be synchronized and rolled back or recovered should an error occur in one. However, the scope of these transactions only reflects one step of the whole business process. The scope of the end-to end business process is commonly referred to as a business transaction, or long transaction. E.g. do you delete the order if a customer refuses delivery and you delete the delivery note? The rules for such business transactions are complex and span a much greater period of time than the transaction integrity of OLTP systems are designed to cope with.

This is compounded when there are many participants in the business transaction. The travel reservation is given as a common example, where flight, hotel, and rental car reservations need to be negotiated with three different companies and if one is unsuccessful, then the others should not be made. Given that one might lose

© cbdiforum.com Limited, July/August 2000

an available 'slot', it is typical to make a reservation for one part, and then go back and cancel it should another be unavailable. Therefore, a series of compensating transactions need to be made to roll back the business process.

Given that the whole business transaction is undertaken by many different applications, undoing it can typically require a good deal of manual processing. This might be a good advert for integrated ERP solutions, however this does not solve the requirements of multiple participants in e-business.

One of the benefits of workflow systems is that by containing knowledge of the whole business process, they can automatically issue compensating transactions to roll back a business transaction. Also, by being aware of the current state of the business transaction, they can roll it back from any stage.

Of course, this might not be as simple as just deleting the prior steps. E.g. you do not disassemble a car because the customer refused delivery on inspection. This example would require you to undo those individual transactions that associated that instance of the car with that specific customer, but not the ones that requisitioned the additional hi-fi the customer had specified. And then kick off an additional process to find a new customer.

Similarly with the travel reservation. If the hotel you try is fully booked, you don't necessarily automatically cancel the flight. You try other hotels until all alternatives are exhausted, and you do not book the flight, cancel the flight, book the flight, cancel the flight, as you try each combination. As such, you need to be careful about simply 'rubber banding' the steps of the business process as something that can be undone as a unit, and double check that any compensating transactions are truly just a reversal of the original.

This I believe is one of the key challenges for workflow systems. Just how comprehensively are business users willing to document their processes, particularly when they have complex exceptions? Two problems are:

- Having seen a lack of desire on their part to convey the complete specification of the system to application developers in the past, why will business users now reveal all to the workflow system?
- Even if they have the desire, they are unable to express the complexity of the system in a form that will result in an effective workflow system (that's why we still have programmers)

This will depend on the ultimate user of the system, and the desire to fully automate the process in order to achieve the benefits of Zero Latency, Straight Through

continues...

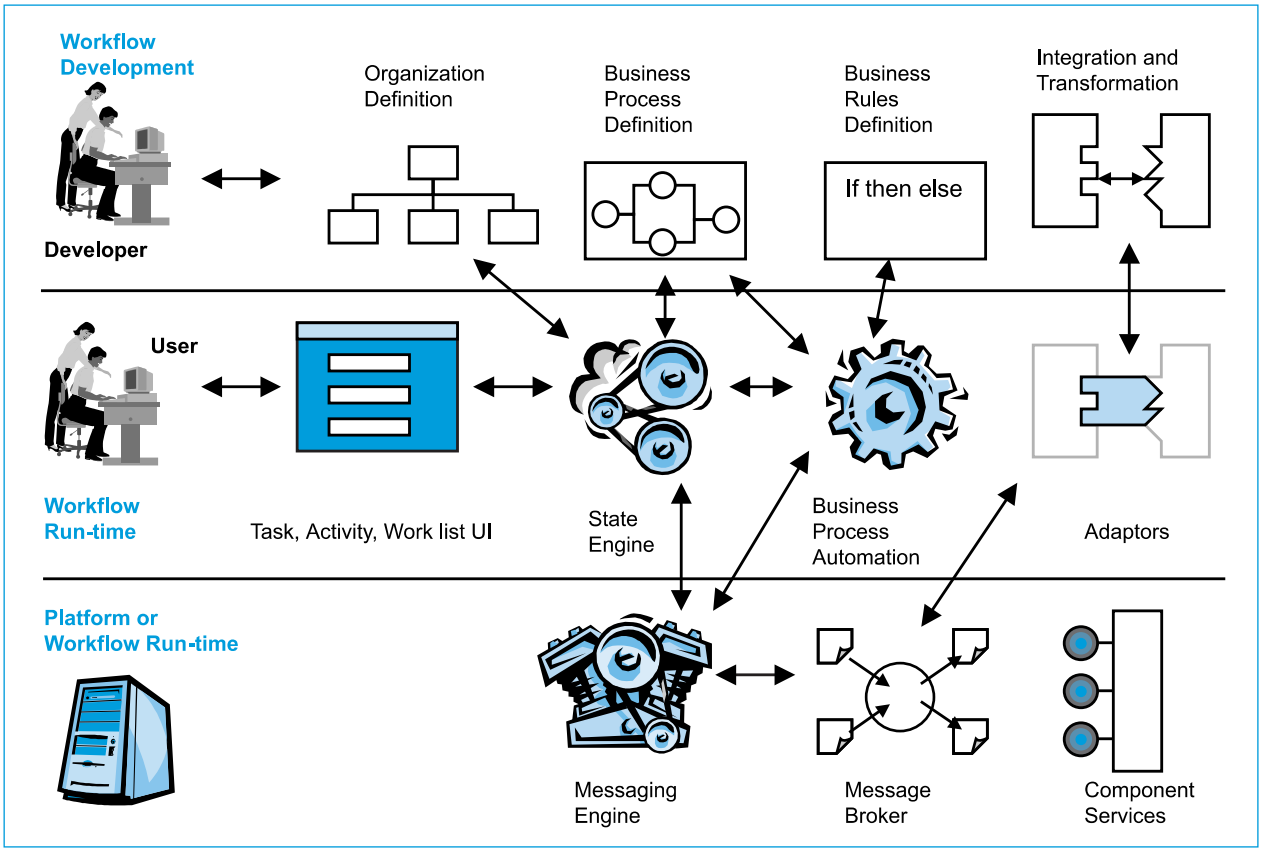

Figure 4: Workflow Components (sic)

© cbdiforum.com Limited, July/August 2000

	Workflow Management Systems	Application Development Tools	EAI Tools	Assembly Tools
Business Process Definition	Yes	Yes	Yes. Business Process is the common metaphor used in describing when integration activities will occur	Yes
Business Rules Definition - Declarative, scripted or programmed	Yes	Yes. Will tend to provide the greatest flexibility (via programming)	Yes	Yes
Organization Definition - Organisational structure, plus actual HR data	Yes	Not usually. Only if built into the application	Not usually	No
Integration and Transformation - defining interfaces, message exchanges, mapping data transformations	Yes. Might be via bundled EAI	Might be via bundled EAI capability, otherwise programmed	Yes. Core competency of EAI	No. Only via external tools
UI Definition - To support Task management, etc.	Yes. Sometimes generated automatically	Yes	Not usually	No
State Engine - Managing state of transaction. Generating tasks, work lists, etc. Managing rollback/recovery	Yes	Not usually. Only if built into the application	Only more monitoring	No. Targets platform
BPA Engine - Sequencing, decisions etc of business process	Yes	4GL type approaches will have some engine for executing business logic. Otherwise compiled into application	Yes	No. Targets platform
EAI-type Adaptors - for different packages, platforms etc.	Yes	Becoming more common to be supplied as part of development tools		Yes

© cbdiforum.com Limited, July/August 2000

	Workflow Management Systems	Application Development Tools	EAI Tools	Assembly Tools
Monitoring - measurement of efficiency in execution of business process	Yes	No	No	No
Messaging Engine - Message delivery	Yes. Might be supplied as part of the Workflow System or by inclusion of messaging system	Supported but not usually included. Expected as part of platform or external messaging system	Yes. Might be supplied as part of the EAI engine or by inclusion of messaging system	No. Expected as part of platform
Message Broker - Sequencing, routing, transformation of messages	Yes Might be supplied as part of the Workflow System or part of the platform	Not usually. Expected as part of platform Might be part of 4GL engine	Yes	Not usually. Expected as part of platform
Transactional - Ensures transaction integrity of workflow. E.g. backs out transactions where message could not be delivered.	Not always. Might be part of workflow or messaging engine	Enables transactions to be defined. Expected to use OLTP	Not always	Enables transactions to be defined. Expected to use features of targeted platform
Long Transactions	Yes	Not unless programmed	Not unless programmed	Yes
Document Management	Common where background of workflow is document centric	No	No	No

Table 1: Workflow Functionality Comparison

Processing (STP) and other such buzzwords. If the system is destined to become self-service with the customer executing the process themselves, or execute as automatically as possible, then it needs to be pretty watertight. See my previous report on e-commerce integration problems3. Clearly any expectations of efficiency due to automated STP will be compromised if manual intervention is constantly sought.

Ultimately a balanced view needs to be taken between how much development effort is expended on achieving automation vs. an acceptable amount of manual intervention.

Fat or Thin Workflow Management?

Having discussed some of the current reasons for workflow management we should consider how it should be implemented.

Why not just build an application? Obvious reasons are time, cost and skills. Whilst a custom application could provide the same function in terms of business process automation, workflow engines can provide a lot more functionality besides. As discussed earlier custom applications don't tend to deal with the state of the

continues...

© cbdiforum.com Limited, July/August 2000

'Why Automate the Business Process?' continued...

business transaction, managing work lists, etc. A WMS is like any other tool. It provides extra functionality you would find difficult to build from scratch, and works at a higher level of abstraction that makes their use far more productive than custom development. Typical functionality of a modern WMS is illustrated in Figure 4.

However, now that a business process driven approach is becoming more popular, several categories of software tools are overlapping in providing some measure of ready built functionality in support of workflow and business process automation requirements. In addition, the engines that underpin WMS could also be used to support other application and are often sold stand alone, or increasingly as part of the operating infrastructure.

The functionality that is required to support workflow and business process automation is examined in Table 1, together with comments on whether this is likely to be included as part of a WMS, application development tools, EAI tools, or emerging assembly tools.

This raises the question of just how much functionality should the WMS contain. As requirements have grown then so has the footprint of the average WMS. This will then lead it into overlap with other tools, and perhaps competition. So the decision becomes not just one of what WMS should be acquired, but a wider debate on whether to use a WMS or an application development tool instead.

The latest version of Compuware's UNIFACE product for example,[5] includes comprehensive support for business process automation with WfMC standards compliance, and supports organizational requirements. As such this blurs the boundaries for application development tools outlined in Table 1.

Then there is the question of just how comprehensive should the support be for the different applications, platforms, message formats, etc, that will be required to be integrated to support Wide Area Workflow requirements. Should the WMS include Enterprise Application Integration (EAI) like facilities for transformation and integration, or just support 'standard' interfaces and delegate it to a EAI tool proper. This is confused by the fact that many EAI tools themselves include business process automation and workflow capabilities, though lack the support for task management and/or organization.

On the other hand, Staffware's Enterprise Objects[6], enables the functionality of this widely used WMS to be consumed as COM, Java or CORBA services so it can be embedded within a custom application.

Meanwhile, Microsoft's BizTalk Server[7] will provide what it terms Orchestration. At first sight this looks primarily like a business process driven assembly tool. In comparison to many WMS this is has very thin functionality. It targets a State Engine running in the BizTalk Server, and as the services of this are accessible then you could build task management and organizational capabilities on top. I would expect to see 3rd parties doing this, if Microsoft does not. Orchestration is best thought of as using business process automation as a way of driving the many different services provided by the Microsoft platform, and other applications available to it, rather than a traditional WMS.

So, should you select a 'fat' or 'thin' solution? Of course it is not that easy, and requires well rehearsed considerations of integrated solutions vs. rolling your own from components. Platform considerations might also feature highly. As you would expect for example, IBM's MQSeries Workflow layers well on top of MQSeries as a messaging engine, and in integrating with the WebSphere platform.

Whilst high-end WMS will ensure their different functions are fully integrated, organizations need to be careful that they are not implementing a multitude of state, BPA and messaging engines as a result of acquiring multiple 'solutions' to various requirements that actually have common infrastructure needs.

We will examine the capabilities of WMS and alternative solutions in future product overviews.

Lawrence Wilkes *lawrence.wilkes@cbdiforum.com*

1. Wide Area Workflow Management: Creating Partnerships For The 21st Century, Gerold Riempp. Springer, 1998. ISBN: 3540762434
2. *http://www.aiim.org/wfmc/*
3. Potential Integration Failure Points in E-Commerce Applications, INTERACT, June 1999
4. Simple Object Access Protocol. - *http://www.w3.org/TR/SOAP/*
5. Reviewed in INTERACT, July/August 2000.
6. *http://www.staffware.com/home/products/seo.htm*
7. *http://www.microsoft.com/biztalkserver/techres/whitepaper_orch.htm*

© cbdiforum.com Limited, July/August 2000

CBDi forum

productoverview

Compuware's UNIFACE Eight

Widely used and respected, UNIFACE reinvents itself and makes a serious play for greater market share by emphasizing the virtues of an open, integrated product line in support of B2B implementation

The Independent Product Overview

CBDi Forum Product Overviews provide accurate and objective reports on products and services supporting component-based development, application integration and legacy rejuvenation.

From the leading information resource for advanced application delivery.

The Component-Based Development and Integration (CBD*i*) Forum

Over the past 18 months we have strongly promoted the architectural concept of an "application" layer comprising of process, event and sequence management, reusing "services" delivered by the component server layers. There are strong arguments supporting this approach, not least separation of concerns allowing the business process to be modified independently of the business logic servers, which can be delivered in a highly independent fashion from disparate sources.

Not surprisingly we observe the entire industry moving to embrace this approach, with huge emphasis on the notion of independent service provision. At the same time we observe the increase in demand for workflow or business process automation support, clearly driven by the urgent requirement to implement the process changes caused by Internet projects and to manage the ongoing adaptation in business models. As organizations implement B2B business models, the requirement for process automation becomes even greater as the correct and timely execution and management of processes becomes a contractual matter between partners.

So what has this got to do with UNIFACE you ask? For those unfamiliar with the product, UNIFACE has its origins as a 4GL programming and deployment environment designed to allow rapid development of enterprise class applications independent of target platform and DBMS. The deployment environment provides a very wide range of platform options including OS/390, AS/400 and various types of Unix including Linux, and Windows 2000. This integrated development environment is widely used by many of the world's larger companies and package builders, that value the integrated, multi-platform, technology independent development

continues...

© cbdiforum.com Limited, July/August 2000

'Compuware's UNIFACE Eight' continued...

environment but generally do not expect the technology to be absolutely leading edge. However we are pleased to report that for the past few years Compuware's UNIFACE product development center has been aggressively pursuing a component-based strategy and they and their customers are clearly benefiting from this strategy.

We visited the product development center last February and at that time commented favourably on their Assembly Workbench,[1] a clever concept that integrates components from disparate sources by importing and managing the interface signature. This was a key step that elevated UNIFACE into an open integration product. At the same time the Compuware services organization showed us the investments they were making in defining a component-based method that has become the foundation for their product development and customer guidance. Today the UNIFACE product line has expanded dramatically beyond its 4GL origins to become an open business process automation environment which addresses the key issue for its large enterprise and package developer customers, how to move rapidly to implement B2B relationships.

Function	Design and Integration	Deployment
Business Process Automation	Business Process Modelling Business Manager Workbench	Business Process Server
Integration	UNIFACE Integration Workbench UNIFACE and XPEDITER/DevEnterprise (Legacy Renewal)	UNIFACE Application Server and UNIFACE Router
Design and Construction	UNIFACE Developer UNIFACE Web Developer	UNIFACE Router UNIFACE Application Server UNIFACE JTi (Java Thin client interface)

Table 1: UNIFACE Eight Product Components

Since we last reported on UNIFACE 7.2.04 in March 1999 there have been two releases: UNIFACE 7.2.05 and 7.2.06. These releases focused on improving Internet enablement for e-Commerce, XML support and legacy integration. The latest release, UNIFACE Eight, now going into Beta status will commence shipping commercially before the end of this year and we will comment on the composite status of the product at the release Eight level.

Business Process Automation

Compuware illustrate a typical customer requirement, where a supply chain involving orders, shipping and supply is co-ordinated between several organizations and workflow and information requirements routinely cross organizational boundaries. The new business process automation module, named UNIFACE Meridian has been developed in-house by Compuware to support a highly flexible, cross-organization process management environment. The new product provides graphical business process modelling and design capabilities plus a business process engine. The product is designed as an application oriented product, as opposed to a document architecture, and provides a straightforward approach that is intended to enable end users to define events, dependencies and business rules-like conditions. The approach is based on modelling an organization in the real world, for example security is implemented by definition of roles and responsibilities of the actors involved. Application components are simply invoked by requesting services and changes in a business process can be effected extremely rapidly at run time by business managers.

The business automation product is a separate product component with a published API set based on WfMC standards where appropriate. Compuware indicate that the product has been architected such that in principle interoperability with other workflow products could be implemented in future, and the component could be offered as a separate product, but no decisions on these matters have yet been made.

Legacy Integration

For many of Compuware's customers integrating legacy applications is an essential pre-requisite for Internet projects. UNIFACE 7.2.05 included a Java thin client

© cbdiforum.com Limited, July/August 2000

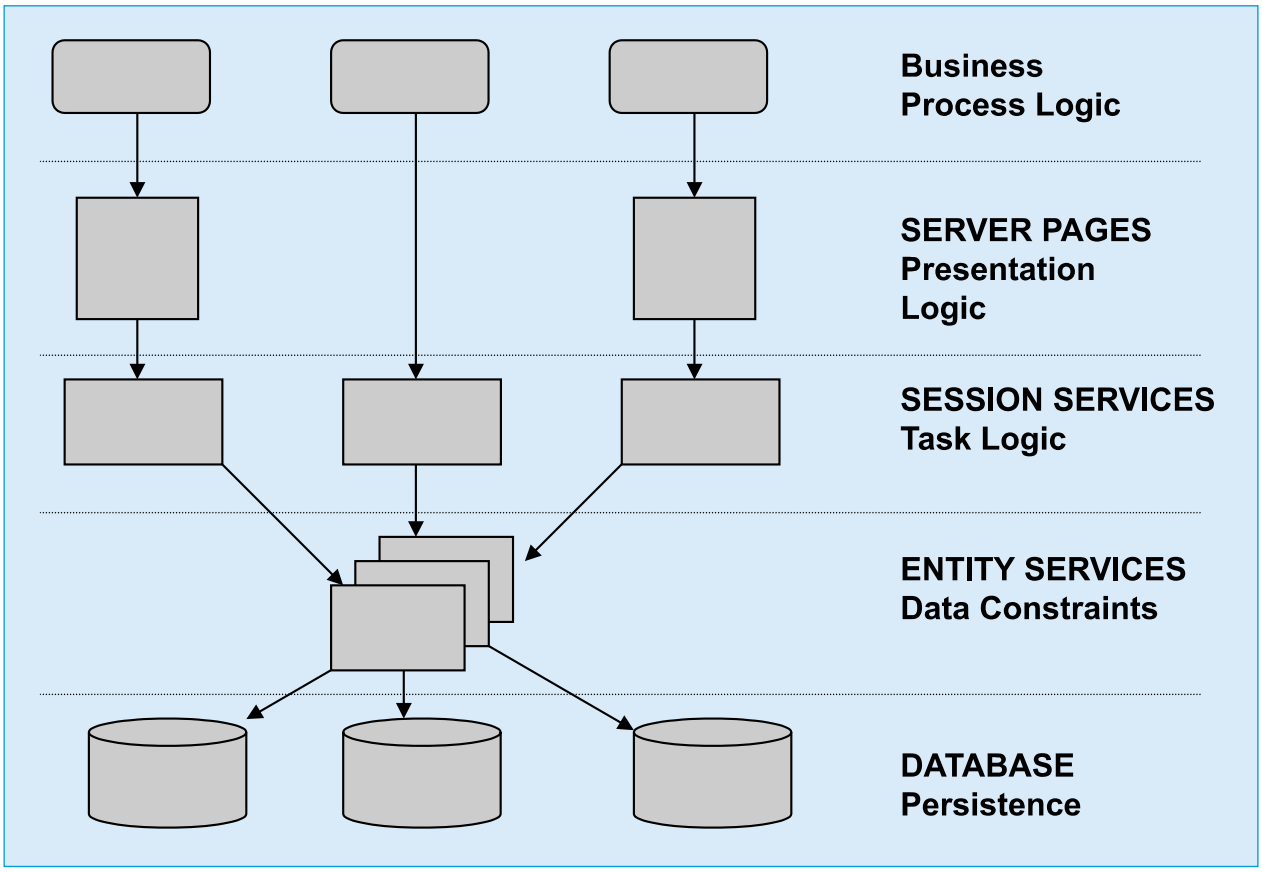

Figure 1: UNIFACE Eight Application Architecture

capability referred to as UNIFACE JTi, enabling existing UNIFACE Seven client/server applications to be deployed as web applications with no code changes required. Customers can choose between self-installable Java applets or a browser plug-in to ensure no loss of UI functionality.

UNIFACE 7.2.06 introduced integration of UNIFACE with XPEDITER, a code coverage product. When a CICS or IMS/DC program is started XPEDITER traces the program path showing the coverage graphically. Whilst this is also useful for documentation purposes, the primary benefit that this delivers is the capability to establish a clear split between presentation logic and application logic by introducing a COMMAREA that can automatically be imported as a signature into the UNIFACE component Integration Workbench and the OS/390 transaction reused as a component, without the overheads of screen scraping. For more details see the separate CBDi Forum report on XPEDITER/ DevEnterprise.[2]

To use the business process automation and integration capabilities fully it is essential that customers' applications be componentized. Recognizing this Compuware have established an approach to assist their customers to upgrade their existing UNIFACE applications. Projects are run by the Compuware services organization, which executes a process that first establishes a test harness, then componentizes the application and then retests before delivering the upgraded application back to the customer.

Design, Construction and Integration

In UNIFACE 7.2.04 Compuware introduced close collaboration with the HTML authoring tool Dreamweaver. This arrangement provides a level of integration between the application development and web design environments, including IDE integration and tag and template reuse.

In release Eight Compuware have introduced a significant change in the internal structure of the application that elevates the component concepts into first order constructs as session and entity services. Previously these were implemented using naming conventions with no enforcement. Compuware product managers explain that whilst you can still override the new architecture on a tactical basis, for example in order to upgrade an application without restructuring, once this migration has been completed a system level switch can be set to enforce the design approach in future. The new development approach is illustrated in Figure 1.

continues...

© cbdiforum.com Limited, July/August 2000

'Compuware's UNIFACE Eight' continued...

We have already mentioned the Assembly Workbench and are pleased to see Compuware renaming this the Integration Workbench. Whilst this might seem like a trivial point this emphasizes the open integration aspects of the latest product and the dramatic change in footprint that has been implemented. The Integration Workbench now includes the ability to import XML DTDs. The Component Construction Workbench allows the use of DTD's to create new session services and also to create new DTDs from the same. The same facility exists for MQSeries. We are also pleased to report that XML can be used at any level of the architecture creating the highest levels of open flexibility that the CBDi Forum have been strongly recommending, providing the option to implement a logical seven tier architecture. Also important is the availability of support for SOAP.

> *"UNIFACE has clearly been successfully reengineered into a modern component-based development and work flow-oriented integration environment"*

The earlier UNIFACE run time architecture has now been substantially revised and simplified to allow a) significantly enhanced scalability and b) open integration with other architectures. There is a new UNIFACE Router that provides load balancing and switching in conjunction with UNIFACE and other application servers plus HTTP servers. The previous universal request broker functions have been distributed between the new server products. The net result is the capability to act as an integrator of a wide range of technologies including CORBA, COM, EJB, IMS/DC, CICS, and MQSeries.

Summary

At this stage we must conclude that the age of the "enterprise application development product" must now be declared over. UNIFACE's conventional competitors have long faded away and UNIFACE itself has clearly been successfully reengineered into a modern component-based development and work flow-oriented integration environment. This represents a genuinely new choice for application developers and integrators that now have the option to choose best of breed, or (and this is really the key) an integrated environment that can effectively collaborate with disparate best of breed environments.

In many respects this choice between integrated or best of breed has always been a matter of philosophy. The integrated environment will never keep bang up to date with the very latest technologies, nor give you the low level control needed perhaps for the minority of applications, nor provide maximum functionality within each of its product components. However for most everyday applications the integrated product provides rapid delivery with a lower skills requirement that is easier to manage. The virtues of working at a model-based level are also well documented allowing the user to target and or integrate multiple technologies either concurrently or over time. We expect existing customers to be pleased with UNIFACE Eight. Other organizations that have heterogeneous development and target environments perhaps including the mainframe, that need to implement a managed component-based environment for B2B should consider UNIFACE Eight as a means to reduce the complexity of integrating the disparate products and developing the skills that are inherent in a native, Java based best of breed strategy.

David Sprott *david.sprott@cbdiforum.com*

1. CBDi Forum Product Overview, March 1999, UNIFACE Component Assembly Environment
2. CBDi Forum Product Overview, July/August 2000, Makeover or Brain Transplant? How Compuware XPEDITER/DevEnterprise could help your applications live longer!

© cbdiforum.com Limited, July/August 2000

CBDi forum

productoverview

Makeover or Brain Transplant?

How Compuware XPEDITER/DevEnterprise could help your applications live longer!

The reuse of existing applications to reduce the time to deliver support for new e-business requirements would be an obvious approach for traditional bricks and mortar organizations to take. Though their inventory of legacy applications will have recently received a good dose of maintenance to get past the Y2K hurdle, there is still a significant effort required to enable them for the web. Compuware's XPEDITER products have been widely used to test, debug and correct code in Y2K projects, and before that. Their latest incarnation now enables the restructuring and reuse of that code.

The Independent Product Overview

CBDi Forum Product Overviews provide accurate and objective reports on products and services supporting component-based development, application integration and legacy rejuvenation.

From the leading information resource for advanced application delivery.

The Component-Based Development and Integration (CBD*i*) Forum

Makeover or Brain Transplant? Which should you choose?

Legacy applications are used to receiving a makeover. So-called screen scraping was common in the transition to client/server, and has found a new lease of life in rendering the interface to those same applications to work in a web browser.

> *"behind that new appearance it is still the same old difficult to maintain application"*

There is nothing wrong with this simple approach as an expedient way of getting applications on the web. However, behind that new appearance it is still the same old difficult to maintain application. Whilst it might simplify network deployment to use a browser instead of a Windows client, I believe that new requirements will mean that the applicability of such efforts will be short-lived, or simply not viable at all.

continues...

© cbdiforum.com Limited, July/August 2000

'Makeover or Brain Transplant? continued...

Reasons for this include:

1. **New presentation devices.** Already WAP phones, interactive digital TV, PDAs, etc. figure alongside PC based web browsers in terms of the devices many organizations must support. How many makeovers must they give to support changing UI requirements? Or course, use of XML for the interface will greatly assist here to remove device dependence.

2. **New business processes.** More difficult to support via a makeover are the changes in business process that e-business brings. Often the business process that is built into the monolithic legacy application fights against the new requirements, forcing a sequence of activities that is no longer relevant to the new process.

3. **Self-service.** A makeover was appropriate when the main focus was on a technology-driven transition to client/server, and putting a PC on every workers desk. The operator was still the same employee who was familiar with the company's processes.In many situations the full extent of that process was in fact only in the employees head, and not in the applications. So how would a simple makeover support self-service by customers or external partners? It is likely a good deal of work would be required to make it usable, not from a technical perspective, but from a business process viewpoint.

4. **Business Integration.** More often the requirements of the new e-business process span multiple applications and extends beyond the organization. As such, existing applications do not need a new UI, but need to be integrated into a new application supporting a much broader business process.

> *"XPEDITER/ DevEnterprise provides valuable tools to help understand what the system does and identify the potential components, and provides automation for their extraction and wrapping"*

Given this scenario a brain transplant would appear more suitable than a makeover. What is required is to take the business logic, the brains of the application, and reuse it in some new form. Unfortunately, brain transplants are extremely complicated in comparison. However, the analogy is appropriate as it is not just a new look that is required but a whole new set of limbs, a skeleton and nervous system if the core of the application is to truly get a new lease of life, not just appear so. To compound matters, the same brain is required to service the needs of several different bodies, i.e. different business processes reusing the same core business objects.

Componentization of the legacy systems to reuse their core business logic therefore seems appropriate. However, until recently this has not been a trivial exercise, nor it must be said, top of many organizations agenda as they got through Y2K and have been happy to leave legacy applications 'as is' where possible.

Legacy componentization involves similar challenges to Y2K including lack of documentation, spaghetti code, and lack of specialist resources for both technical and business understanding issues. This is further complicated by the need to manage the transition between old and new environments, such that the componentized business logic can continue to support the existing legacy application it was created from, as well as new applications.

Enter XPEDITER/DevEnterprise

Fortunately, development tool technology has now moved on to support this requirement. Compuware's XPEDITER/DevEnterprise is promoted primarily for Y2K and Euro conversion type projects, where the analysis of code structures and the ability to 'trace' how data flows through a system and hence catch all the code that needs amending is highly useful when those systems are less than ideally documented. However, it also provides a 'code splitting' capability that allows the analyzed structure to be re-engineered into discrete components.

XPEDITER/DevEnterprise provides valuable tools to help understand what the system does and identify the potential components, and provides automation for their extraction and wrapping. The following outlines the main activities supprted by XPEDITER/DevEnterprise.

Application Analysis. First, the whole system can be analyzed to understand the relationships between modules. This could be a CICS region for example. This builds the relationships between the transactions, programs, subroutines, maps etc.

Program Analysis. More details of the structure of individual programs is then built by analyzing the

© cbdiforum.com Limited, July/August 2000

source code. This isolates perform groups and helps separate out I/O from business logic.

These steps, which can be run on the mainframe, populate a repository that underpins XPEDITER/DevEnterprise, that in turn feeds a Windows-based client that enables the application and program structures, and the various data flows to be examined in an easy to use GUI environment. The repository makes it easier to take a system wide view, which is important as file access and associated data rules are commonly duplicated across programs, as is business logic.

XPEDITER/DevEnterprise shows its heritage by focusing its capabilities on the IBM mainframe environment, though after all this is where the majority of legacy assets remain. CICS, IMS and batch (via JCL) COBOL applications can be analyzed at the application level, with PL/1 and Assembler specific information provided to the level of programs, copybooks, jobnames and stepnames, though only COBOL can be analyzed at the program level and automatically restructured into components. However, any ANSI II COBOL can be treated this way.

Segment Component. Once Program Analysis is complete the next stage is to identify the code that is to be made into a separate component. Once identified, the actual code splitting is an automatic process. Not only does it create the component, wrapping up the isolated code with an interface, it also inserts the necessary calls to that interface into the original program so that it still functions. There are several scenarios where code splitting could be very useful as illustrated in Figure 1.

1. **Split Application.** A straightforward application of this in terms of web-enablement is to take existing 3270-based applications and separate out the screen I/O and navigation from the actual business processing, enabling it for example to be invoked directly from a new application running in a web server. Both the existing 3270 application and the new web application would use the same new interface. This is more of a face-lift than a simple make over, as it clearly requires some invasive surgery on the application. At the other end of the program, the same approach of separating out I/O could be used to create data components, or perhaps help manage the transition to a new DBMS. XPEDITER/DevEnterprise can analyze the usage of DB2, IMS databases, and identify calls to IDMS and Adabas.

continues...

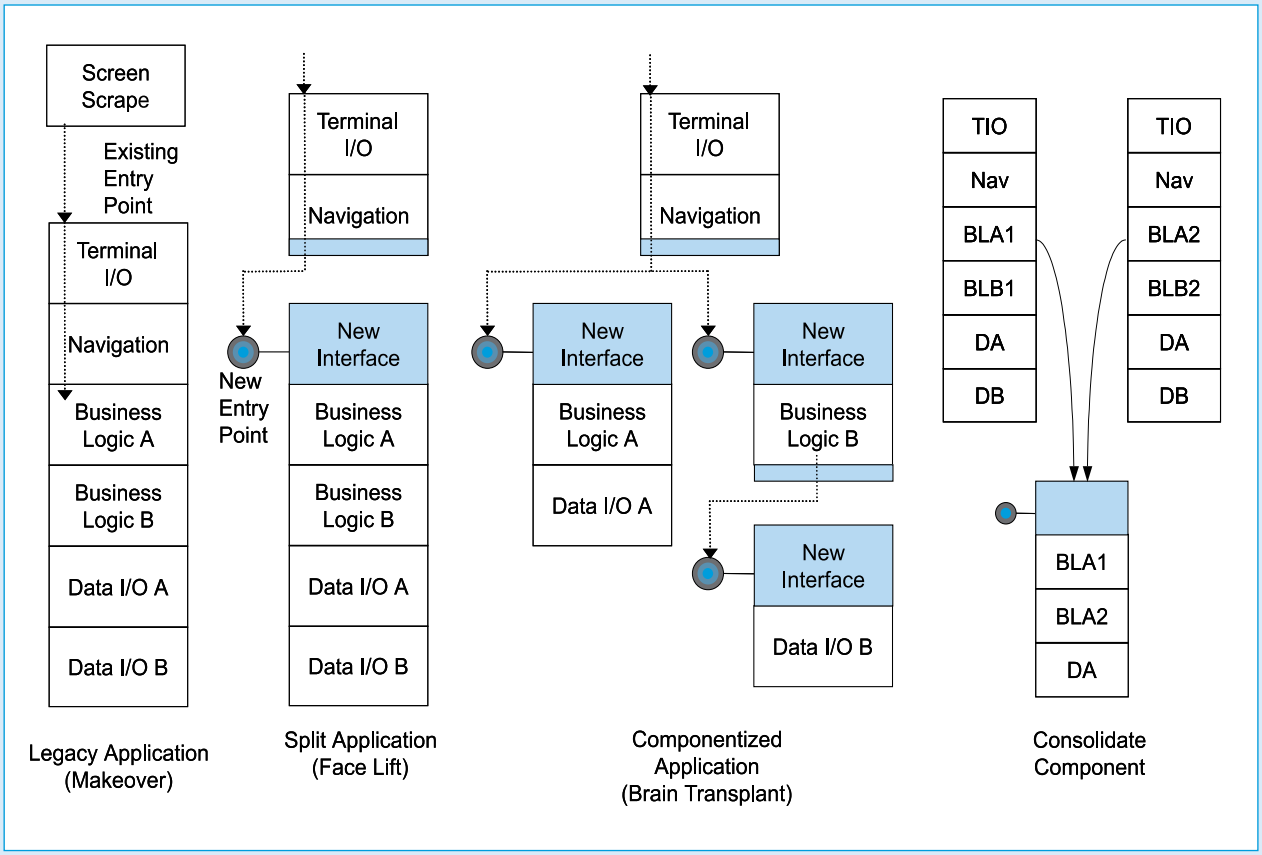

Figure 1: Wrapping, Splitting and Componentization

© cbdiforum.com Limited, July/August 2000

'Makeover or Brain Transplant? continued...

2. **Componentize Application.** The next step would be to group and separate different areas of business and data logic into discrete components. Business logic and Data access could be componentized into separate layers too.

3. **Consolidate Components.** A common objective for componentization is to remove duplicated code from many programs and isolate it into a single component, or to consolidate different pieces of code that is associated with the same business object but is fragmented to many applications. XPEDITER/DevEnterprise's data analysis tools and wizards support the detective work required to trace how data is used across the whole system and where duplication or fragmentation might occur. Once identified, the code splitting process can be performed but this time on multiple programs, so that they all invoke the same component. However, if different but related code is required to be consolidated into a single component (as illustrated in figure 1) then XPEDITER/DevEnterprise's automation capabilities need to be complemented with manual intervention.

The Bigger Picture

A key advantage of XPEDITER/DevEnterprise is that it is integrated into the wider range of testing tools that Compuware provides. As well as being a natural fit with the rest of the XPEDITER range, it also can be managed from QADirector, Compuware's testing 'consol', and work with other tools from the QA suite.

An obvious partner would be XPEDITER/Code Coverage. This is used to monitor a system to see which parts are actually executed, and can identify this down to the individual COBOL, PL/I or Assembler statement level. Not only does this support the analysis stage in helping to identify which parts of the system are related, but can also be used as part of a regression test to measure whether the same statements in the segregated components are still being executed as they were when they were part of the monolith. As you would expect, these two tools work seamlessly together.

XPEDITER/DevEnterprise has wider usage than just componentization too. Euro conversion, and other significant maintenance tasks would obviously benefit from a clearer understanding of the systems involved.

Conclusions

The reuse of code from legacy applications has been something of a holy grail for some. However, looking at the effort involved it is not surprising that to date, most organizations have chosen to leave the application as is, or simply rewrite it. Compuware's XPEDITER/DevEnterprise changes this equation. Components can be identified and extracted in an easy to use tool environment dedicated to that task rather than scanning endless listings, then cutting and pasting what you hope is the associated code.

There are some limitations. The ability to only code split COBOL is one, though this does make it suitable for the majority of applications. Another is that the segregated components still remain as mainframe COBOL applications, though this should not be an issue for those users who place emphasis on faster, easier access to existing code to support new business requirements above that of porting the code to a new environment.

XPEDITER/DevEnterprise is not a silver bullet. You cannot tip legacy code in one end and pour components out the other at the touch of a button, and such is the complexity (and mess) of many legacy applications it is doubtful such a solution will ever exist. What it does do is simplify the task, and importantly ensures the accuracy and completeness of the work, especially when used together with other XPEDITER tools.

More information can be found at *http://www.compuware.com/products/iad/*

Lawrence Wilkes is CTO and Principal Analyst of the CBDi Forum and can be contacted via email at *lawrence.wilkes@cbdiforum.com*

© cbdiforum.com Limited, July/August 2000

productoverview

Reviewer for Select Enterprise

IT projects are often like cobblers children; we are the last to get new shoes. We report on an interesting process improvement tool from Princeton Softech that is specifically designed to automate and improve core project management tasks.

It is not a new idea that it is possible to automate many of the more routine tasks of application development. The existence of a comprehensive metadata based repository enables the intelligent analysis and usage of the work in progress to assist in for example test data generation, progress reporting and quality assurance. One of the most difficult issues for project leaders, analysts and designers is determining when the modeling activity is "complete".

There is the ever-present dichotomy between incomplete understanding and analysis paralysis. Project leaders and managers live or die by their understanding of project status, the combination of completeness, correctness and consistency. Over time I have been aware of several and have personally used one tool that analyses the repository based data and provides reports on the assessment of completeness, correctness and consistency. These tools have varied in usefulness depending on the quality of the underlying metamodel and the sophistication of the analysis. Of course today we have UML as a common backplane to many application development environments and this provides a uniform, standards based set of deliverables which creates a good basis for analysis.

I am pleased to report that Princeton Softech has recently announced the availability of an add-on product called Reviewer for Select Enterprise that aims to provide project analysis services to project managers from the underlying repository data. Many will be aware of Princeton Softech's Select Enterprise, a sophisticated UML based modeling,

continues...

The Independent Product Overview

CBDi Forum Product Overviews provide accurate and objective reports on products and services supporting component-based development, application integration and legacy rejuvenation.

From the leading information resource for advanced application delivery.

The Component-Based Development and Integration (CBD*i*) Forum

© cbdiforum.com Limited, July/August 2000

'Reviewer for Select Enterprise' continued...

	Style	Completeness	Standards	Correctness	Report	Componentization
Analysis	●	●	●	●	●	●
Architecture					●	●
High level Design	●	●	●	●	●	●
Low level Design	●	●	●	●	●	●
Testing	●	●			●	
Performance					●	●

Table 1: Reviewer Concepts

design and build environment which provides explicit support for component based development and assembly. Reviewer for Select Enterprise is an add-on product that provides continuous assessment of project status.

The Reviewer concept is based on a set of quality and completeness criteria that are organized into a number of levels as shown in Table 1 and which are used in an automated assessment process. You might consider this analogous to a spell checker for application development.

Each category has a set of criteria for each of the qualitative assessment categories. For example in the case of the Analysis category, style addresses matters such as avoiding the use of multiple inheritance, completeness covers mandatory fields, data types and implementation of abstract classes. Standards covers the content of diagrams. In the case of Architecture, Report addresses for example the identification of operations that cross architectural boundaries and the reporting of dependencies between packages. In this category, componentization covers the integrity of inter package operations.

> *"The Reviewer approach formalizes this evolutionary (team building) process in a manner that encourages the formalization of decisions with easy to use documentation facilities."*

The Reviewer product is installed very rapidly and simply adds two menu options to the Select Enterprise Tools drop down menu - Reviewer and Standards. The Reviewer menu option allows the user to define a wide range of criteria for completeness, standards, correctness and componentization. There are some 75 options that may be checked. For example at the simple level declarations of matters such as definitions of mandatory fields, items and options to be used on diagrams and matters of style and notation. And at the practical guidance level matters such as policies on use of inheritance, use of abstract operations, inter-package operations and treatment of operations that cross architectural boundaries. The Review process is run on request, and for my relatively simple model (one Use Case, One Class Diagram, three Process Threads) returned the results in nine seconds. The reports provide summary counts of object types and status in total and by author and package. Then detailed reports provide blow-by-blow details of exceptions. Reports can be customized as necessary.

The Standards menu option provides the project team or organization with guidance on policies, practices and patterns. The facility also allows you to document locally agreed standards and to define examples to assist team members. The product comes initially with a comprehensive set of project standards and this facility allows the team to refine and modify the standards to suit the specific circumstances. This is actually really useful because as we all know, as a team comes together, getting agreement on standards and policies is not something that happens overnight, but instead evolves over some considerable time, with major decisions often being taken in formal reviews. Consequently the Reviewer approach formalizes this evolutionary process in a manner that encourages the formalization of decisions with easy to use documentation facilities. As might be expected there is also a Web Assistant that will give access to a considerable body of material that is updated on a dynamic basis by Princeton Softech.

© cbdiforum.com Limited, July/August 2000

My experience of this type of tool in the past has been rather mixed, and I think this was due to three factors. First the assessment criteria (in my experience) were pretty elementary. For example a strong emphasis on completeness and correctness was in evidence. Second the reporting process was just that. It was a tool for project managers to find out what the teams were doing and to complement the project management process. Whilst the reports were used effectively at review time, the separation between reporting and analysis and design tools did not encourage the project team members to utilize the assessment facility. Third the facility was not really useful to expert level analyst/designers. Where the Reviewer concept works well is that it is not just an assessment process, but also an amalgam of guidance and assessment that will be equally useful to designers and developers on an ongoing basis as well as project managers. The feedback loop works. I was also reasonably impressed with the level of insight in the qualitative assessment criteria. I would hope and expect that over time these would be extensively enhanced both by project teams and Princeton Softech which would in fact encourage the experts to capture their expertise as patterns that can be more easily adopted by others.

"Using Reviewer review processes will generally be much more productive because models should immediately adhere to basic standards and basic completeness checks should be cleared out the way allowing reviews to rapidly focus down on more productive issues."

Princeton Softech also report that they expect the product to provide a "prompt option" where the analyst/designer can use the facility in the same way that we use spell checkers today on a dynamic basis. I actually think that a better analogy might be the grammar checker, and if you are like me, I tend to either ignore or turn this off, except when I know that I need it. But it is a useful tool that experts and novices alike will value. Princeton Softech and one of their customers are reporting some 10 to 20% improvement in project productivity using Reviewer. I would expect that this improvement comes from the contribution that Reviewer makes to the overall process. The net effect of using Reviewer should be that review processes will generally be much more productive because models should immediately adhere to basic standards and basic completeness checks should be cleared out of the way allowing reviews to rapidly focus down on more productive issues.

Reviewer is an add-on product for Select Enterprise. The product has been developed for Princeton Softech by an independent organization, Development Process. Princeton Softech indicate that Reviewer may be made available for other modeling environments soon, although they comment (perhaps a little tongue in cheek) that of course not all modeling and development environments have the same depth of metamodel as Select Enterprise and therefore the value to the process will be proportional. Whilst we fully agree on the relative comprehensiveness of repository contents, we do think that the major benefits of Reviewer will accrue primarily from overall process improvement and that Princeton Softech have delivered an important add-on to the archetypal development and delivery process that all organizations should take a look at.

David Sprott *david.sprott@cbdiforum.com*

Further information at: *www.princetonsoftech.com*

© cbdiforum.com Limited, July/August 2000

www.cbdiforum.com

The CBDi Forum

www.cbdiforum.com

The CBDi Forum is the world leader in analytical information and experience sharing on the topic of component based development and integration. The Forum provides comprehensive services to IT middle management (architects, strategists, business analysts, development managers, infrastructure managers and lead developers), which allows them to make effective decisions. The Forum has 3000 subscribers and members world-wide.

The CBDi Forum was formed in late 1997. The Forum concept is founded on the conviction that the industry is undergoing a major (ten year) change in practices and process, enabled by technology that will lead to radical reengineering of software products, markets and processes. Our mission is to be the world leading analytical information resource on component based application delivery and integration by providing deep insight into practices, processes, technologies and markets and to ensure widespread understanding and credibility of CBD concepts and practice.

The INTERACT Journal is published monthly and subscription is included in Silver and Gold CBD*i* Forum membership.

If you have any questions or comments on the reports in this and other Journals, or suggestions for future topics and experience you would like to share, Contact the Forum Analysts:

David Sprott
Email: david.sprott@cbdiforum.com
Tel: +44 (0) 1483 571151
Fax: +44 (0) 1483 440860

Lawrence Wilkes
Email: lawrence.wilkes@cbdiforum.com
Tel: +44 (0) 1344 776196
Fax: +44 (0) 1344 776185

If you would like further information before deciding to subscribe to the CBDi Forum please contact us by telephone on +44 1483 578450 or by email at info@cbdiforum.com

cbdiforum.com Limited, PO Box 536, Guildford, Surrey, GU4 7BX, England

Products & Services Vendors

Make sure your organization is represented in the new Forum directories and that your entries are up to date. Use the on-line submission form to submit/modify your directory entry:

Legacy Rejuvenation Market Directory:
www.cbdiforum.com

Component Market Directory at
www.componentsoftware.net

The Component Software Directory from the organisers of the CBDi Forum

IMPORTANT NOTICE: The information available in this publication is given in good faith and is believed to be reliable. cbdiforum.com Limited expressly excludes any representation or warranty (express or implied) about the suitability of materials in this publication for your purposes and excludes to the fullest extent possible any liability in contract, tort or howsoever for implementation of, or reliance upon, the information contained in this publication.