A management platform for commercial Web Services

J R Hill

Web Services offer the promise of allowing BT, along with its customers, partners and suppliers, to be able to assemble distributed applications that span multiple systems and organisations. The commercial drivers for this are cost reduction, new revenue opportunities and a better service through the use of automated end-to-end business processes. To achieve these benefits, BT will require a fully managed internal Web Services infrastructure.

At the same time, creating a managed platform for our customers' Web Services deployments provides us with a new business opportunity. Early experience suggests that 20% of the effort in creating a commercial Web Services application is expended in defining the commercial opportunity, the service interfaces and server implementations. The remaining 80% is spent on ensuring that the operational and deployment environment is adequate to support business-grade service levels. Managing that 80% of the problem for our customers is a business opportunity for BT.

This paper describes the requirements for a management platform for Web Services and then goes on to describe some of the value added business services that this platform can support.

1. Web Services management — an introduction

Web Services management is the process of bringing the practical implementation of a solution based on the Web Services model under effective control.

Web Services are intended to be loosely coupled, distributed, and reusable. These characteristics confer significant benefits in terms of flexibility and efficient use of technical resources. However, the flexibility can quickly turn into creeping complexity if the services are not deployed as part of an enterprise-wide architecture and also brought under management at an early stage. At the time of writing, management platforms for commercial Web Services can achieve this using non-standard approaches and proprietary technology.

As Web Services evolve, it is expected that standards, or at least standardised approaches, will be developed to solve these problems.

In order to understand the management issues, it is necessary to have some technical understanding of what a Web Service is, how it is constructed, deployed and used. There are many detailed sources of information on this subject but it is convenient to summarise the essential details here.

- Commercial contract

  There is no absolute requirement for a commercial contract to be associated with a Web Service but, at a business level, a Web Service almost certainly represents a contract to provide some service through a technical interface. It is either explicit or implicit that the service will be provided with certain performance characteristics and terms and conditions will apply to its use. Often the use of the service will be restricted to identified and authenticated individuals, groups, applications or servers. There may be business contractual issues related either to the performance of the service, its availability, or the transactions that are supported.

- Interface description

  It should be possible to write a client application for a Web Service without needing to know the way in which the service is implemented. The interface definition of a Web Service provides information about the location and specification of the actual operational service. This interface description is expected to be written using the Web Services Description Language (WSDL) and may be published as a file by the service provider in a searchable registry conforming to the UDDI (universal description, discovery and integration) specification. Anyone wishing to use the service must retrieve the WSDL file from the registry and can use this as a specification to create a client application. A published WSDL file can be thought of as a 'technical contract' between the provider and consumer of the service that it will be available at the given location and provide the functions that have been specified.

- Service presentation

  The service presentation is the logical location at which the service is available. This instance must appear at the location given in the WSDL and provide the published functions. It is important to understand that the service presentation is a logical rather than a physical entity. This is analogous to Web page addresses or URLs where the physical location of the service is entirely hidden from the browser.

- Service implementation

  At a deployment level, the Web Service is presented to the service consumer as an interface available over the network. This interface has to be supported by an actual implementation of the core service functionality. This is normally provided by a software application although a hardware solution could also be used. Multiple instances of an implementation can provide the same service, as long as calls to the service presentation can be balanced across them.

- SOAP message

  Web Services messages are analogous to posted letters. They have an envelope with addressing instructions and a body that contains the information 'payload'. A Web Services message is written in extensible mark-up language (XML). The structure of the envelope and body must conform to the rules laid out in the simple object access protocol (SOAP) specification. The layout of the body, including attribute names and data types must conform to the interface specification of the specific service called. At run time, a client application calls a Web Service by sending a SOAP request to the destination address for the service and a SOAP response message is returned. The values of the attributes will of course differ from invocation to invocation.

Web Services are intended to be loosely coupled and distributed over networks. They expose a set of open interfaces that are potentially available to all clients that have a network path to them. Although technically the client and service are loosely coupled, for all non-trivial applications there remains a strong or even critical business interdependency between the client/server pair. This forms the core of the commercial Web Services management challenge. How can the technical benefits of reuse, interoperability and loose coupling be achieved between federated systems, while at the same time supporting the business imperatives of high security, availability, accountability, resilience, and now agility?

2. The need for an enterprise architecture

In the past, business and systems architectures tended to be based on monolithic stacks built more or less independently round departments or lines of business. While these delivered productivity gains and cost savings, they tended to produce duplication and were also very hard to maintain.

Two of the key business drivers for the widespread adoption of Web Services are the efficient use of resources and the need for increased business agility [1]. These business drivers are leading to a growing trend towards a new type of flexible business and technical 'service-oriented' architecture.

A service-oriented architecture (SOA) is where the end-to-end processes that enable the business are defined in terms of a managed set of reusable services. Within an SOA, a business redefines its resources as a set of services that can be flexibly combined to meet its business objectives [2]. Web Services are a key enabler [3] for an SOA, providing standards for quicker and cheaper integration. Industry standards are also being proposed for end-to-end processes including electronic business processes and electronic business documents.

By adopting electronic business standards, organisations have the opportunity to create faster, more accurate and less expensive automated processes. These can be extended to include customers, partners and suppliers, resulting in stronger and more effective business relationships. Gartner [4] predicts that, by 2007, SOA will be the emerging mainstream software architecture.

The two paradigms of SOA and Web Services are synergistic and it is argued that the full benefits from Web Services can only be realised if they are adopted within a service-oriented architectural framework that can exploit and manage the resulting flexibility and reuse.

The SOA can be regarded as four layers that separate presentation, process, service and resources (see Fig 1). Common standards and infrastructure provide interoperability, co-ordination and reliability.

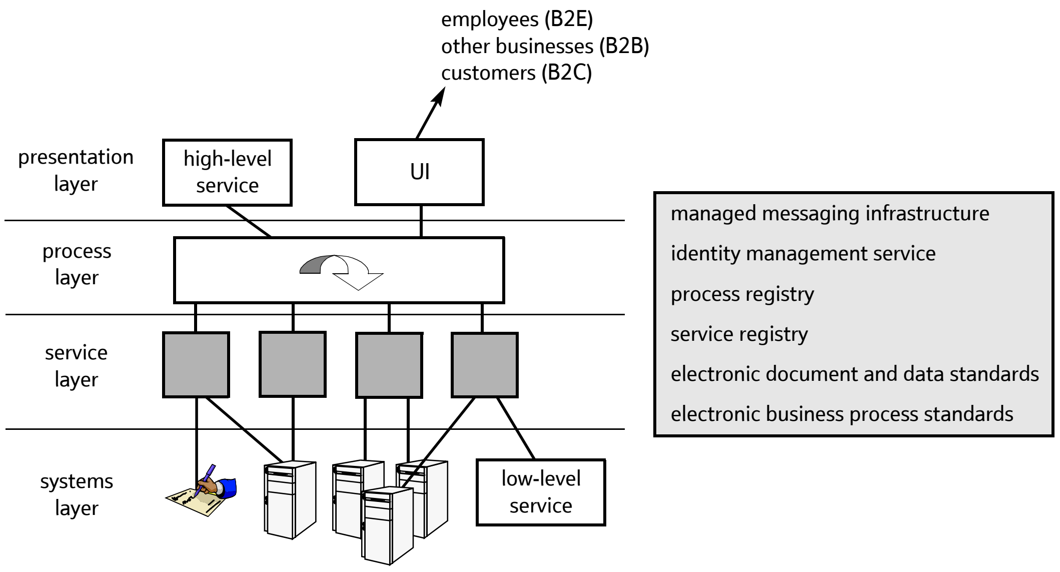

Fig 1 A simple service-oriented architecture.

- Presentation layer

  At the top of the stack, the presentation layer is responsible for managing the dialogue with end users. This could be via a Web browser interface or other user-friendly interface. It could also be expressed as a high-level service that the user integrates into their chosen application. This allows the architecture to be recursive with high-level services being combined from lower level ones. The separation of presentation from business process allows the same business activities to be provided though a number of communications channels and devices.

- Process layer

  The process layer provides 'glue' that ensures that a series of services are executed in a particular order, according to business rules, to achieve a business aim. The separation of processes from the supporting services increases business agility since processes can be created, modified and withdrawn without needing to redefine the supporting services.

- Service layer

  The service layer contains a set of reusable business functions. A service will be made available through an interface conforming to standards. High-level services provide functions such as ordering a broadband circuit, arranging an engineer's visit, reconfiguring a firewall, or requesting a bill. Low-level services providing finer grained functions are also possible and would be found at deeper levels within the architecture. A managed set of services encourages reuse of similar functionality across the business, leading to cost reduction and improved synergy between business areas.

- Systems layer

  Services can be implemented in a variety of ways, ranging from manual tasks performed by people through to automated systems, or a hybrid of the two. The system layer contains any physical implementation required by the services in the service layer. Because the service layer hides the system layer, the implementation can change as long as the external service remains unaffected. The hiding or 'encapsulation' of systems by services provides flexibility to upgrade, consolidate or outsource internal infrastructure without having any impact on end-to-end business processes.

- Common infrastructure

  Configuration management across the business is achieved through the use of enterprise-wide registries and directories that manage information on services, processes and user identities. Effective interoperability is achieved by adopting common standards for electronic business processes and data. Operational reliability is enhanced through the use of a common managed messaging infrastructure.

Management of the architecture is a challenge that should not be underestimated. Although there are technical issues that need to be addressed, the primary challenges are cultural and organisational. The SOA makes extensive use of common infrastructure and is dependent on a willingness to share resources. This requires funding and accounting practices that encourage reuse and prevent slightly larger initial development costs for reusable assets being fully loaded on to the originating project.

The technical architecture needs to be managed to ensure that the subset of standards used to access and exchange information between services is agreed. This includes interface and messaging standards and also a common set of business and operational models, processes, documents, and schemata, with tightly defined structure and meaning. These need to cover at least the minimum set of standards to ensure interoperability within an organisation and needs to be regulated through an effective governance framework.

Fundamental to the success of the SOA is a service registry, which should be centrally managed. The registry will be the first point of reference for any team seeking a service that satisfies their business objectives. In addition to a service registry, there may also be a need for the architecture to contain a process registry. Using these, proposition and product managers, working with the customer service community, can define automated business processes running across services that represent new offerings or mass-customised solutions.

Although the vision of the SOA is radical, implementation can be evolutionary. Once a governance structure has been put in place, implementation should be phased to provide specific business benefits for which a business case has been made. This is made possible because of the inherent flexibility provided by the layering of the SOA.

3. Management of service creation and reuse

The adoption of Web Services tactically as a simple integration technique can probably be managed comfortably within existing application development and deployment practices. At this level, Web Services merely represent another integration technology with a new set of rules for exposing interfaces and making remote calls.

Web Services applications have similar design and deployment rules to those used to deliver Web pages. They should ideally be stateless and deployments can include redundancy and load-balancing techniques to achieve scale and resilience. Technical delivery of individual Web Services and client applications is exceptionally well supported in recent versions of most deployment platforms. Microsoft, BEA, Sun Microsystems, IBM, Novell and most other major software platform vendors provide Web Services development environments and support Web Services deployment.

In addition to the creation of entirely new Web Services, there is strong vendor and third-party support for the exposure of legacy systems and technology as Web Services. The systematic exposure of legacy systems in BT as Web Services is described by Calladine [5].

One of the most frequently quoted benefits of Web Services is the potential for reuse, i.e. to be able to create a service that can be reused by many different client applications. Platforms for commercial Web Services will need to provide some structure to service creation, change and withdrawal. Vendors such as Infravio [6] and Actional [7] are now developing products that can perform configuration management of Web Services and provide support for the development life cycle.

The design of even relatively simple distributed applications can be challenging and this is compounded when existing services that are being called are not under the designer's influence. This places a responsibility on designers and implementors of new services within an organisation to ensure that the services that they release are fit for reuse. At a minimum this requires that services be properly tested, including specific tests for vendor platform interoperability. The opportunities for reuse will be increased further if design and deployment is managed as a co-ordinated process leading to defined sets of complementary services.

Interoperability problems between different vendor platforms can be a problem despite the existence of standards and a high degree of platform vendor co-operation. These arise from subtly different interpretations of standards, implementation of unstable proposed standards, and also from the use of options within standards. For example, WS-Security is a specification that defines a general-purpose mechanism for associating security tokens with messages [8]. However, no specific type of security token is required by WS-Security and it is entirely possible for client and Web Services to comply with the specification but be unable to interoperate at a working level owing to the use of incompatible security tokens.

This type of issue is being managed as an industry-wide initiative by the Web Services Interoperability organisation (WS-I) [9]. Formed in February 2002 it aims to promote a consistent and reliable inter-operability between Web Services across platforms, applications and programming languages. It provides guidance, recommended practices and supporting resources such as test tools for developing interoperable Web Services.

Ideally, a business will attempt to factor out reusable functionality into a manageable number of services and ensure these are designed and deployed in a co-ordinated way and to an appropriate quality and scale. In reality, it is likely that a more *ad hoc* approach will be adopted. In either scenario, a commercial Web Services platform must support the creation of properly qualified and tested services and maintain a registry of available services for designers to reuse.

For Web Services, the accepted way to publish and discover reusable services is via a UDDI registry. Although UDDI provides a technical mechanism for managing the definitions of available Web Services, it is insufficient in isolation and would need to be employed within a more complete delivery management framework that provides:

- a clearly expressed and consistently enforced management commitment to reuse,
- a clearly defined policy for the systematic reuse of Web Services,
- a set of Web Services presentations,
- a registry containing information about the Web Services available,
- a set of processes for the acquisition of new Web Services,
- a set of processes for the change management of existing Web Services,
- a set of processes which maximise the use and resulting benefit from the repository.

It is rather surprising to find that, although there is widespread support for Web Services creation and deployment, there has only been limited support available for a service-oriented development methodology that promotes and supports systematic reuse. Methodology for service-oriented development of applications [10] (SODA) has been proposed, but it is still relatively immature.

BT has proposed and developed platforms that support application development making use of reusable commercial Web Services. The managed Web Services gateway platform [11], developed by BT, was created to allow operators to offer wireless and fixed network capabilities to a mass market of developers in a managed environment. Examples of such services are shown in Fig 2 and can include capabilities covered in specifications such as Parlay X [12, 13], which covers location-based services, messaging, payment, and simple call control. By providing a service exposure engine, a developer's portal and an operator's portal, the platform enables the commercial provision of services, self-service developer account management, and management of the accounts and services by the platform operator. For example, operators can monitor their gateway and service performance, developers' service usage, activate and deactivate individual applications, developers, or services, and set limits on platform load.

In addition to supporting feature-rich services deployed in the platform, the whole management and control infrastructure can be simply reused to export existing or new Web Services that are not currently supported by an appropriate framework. Tools are provided to simplify this process.

4. Management of the operational environment

Web Services operational management can conveniently be partitioned at the deployed Web Services interface. The operational management of the application deployment platforms that implement the interfaces is essentially the same as for other Internet server applications. The physical environment, operating system, application server and running application clearly need to be brought under effective control to ensure uninterrupted service but the procedures required to do this are well understood.

It is probable that Web Services will increasingly be used to extract performance data from application platforms that support the business and to provide interfaces on these systems to control their behaviour. In this situation, it is suggested best practice to group and clearly separate user Web Services from management Web Services.

The distributed nature of Web Services deployment, together with the interrelationships and dependencies between the components, creates an additional requirement to perform operational and configuration management of the assembly of services.

The long-term ideal would be for a management solution that is both standardised and federated so that the participants in the assembly can be self-configuring and self-assembling. At present this is not a viable option and an overarching operational framework is required. This must support a large number of run-time requirements including:

- implementation of security policy and provision of security services,
- client and Web Service identification, authentication and authorisation,
- protecting messages from corruption, interception and modification,

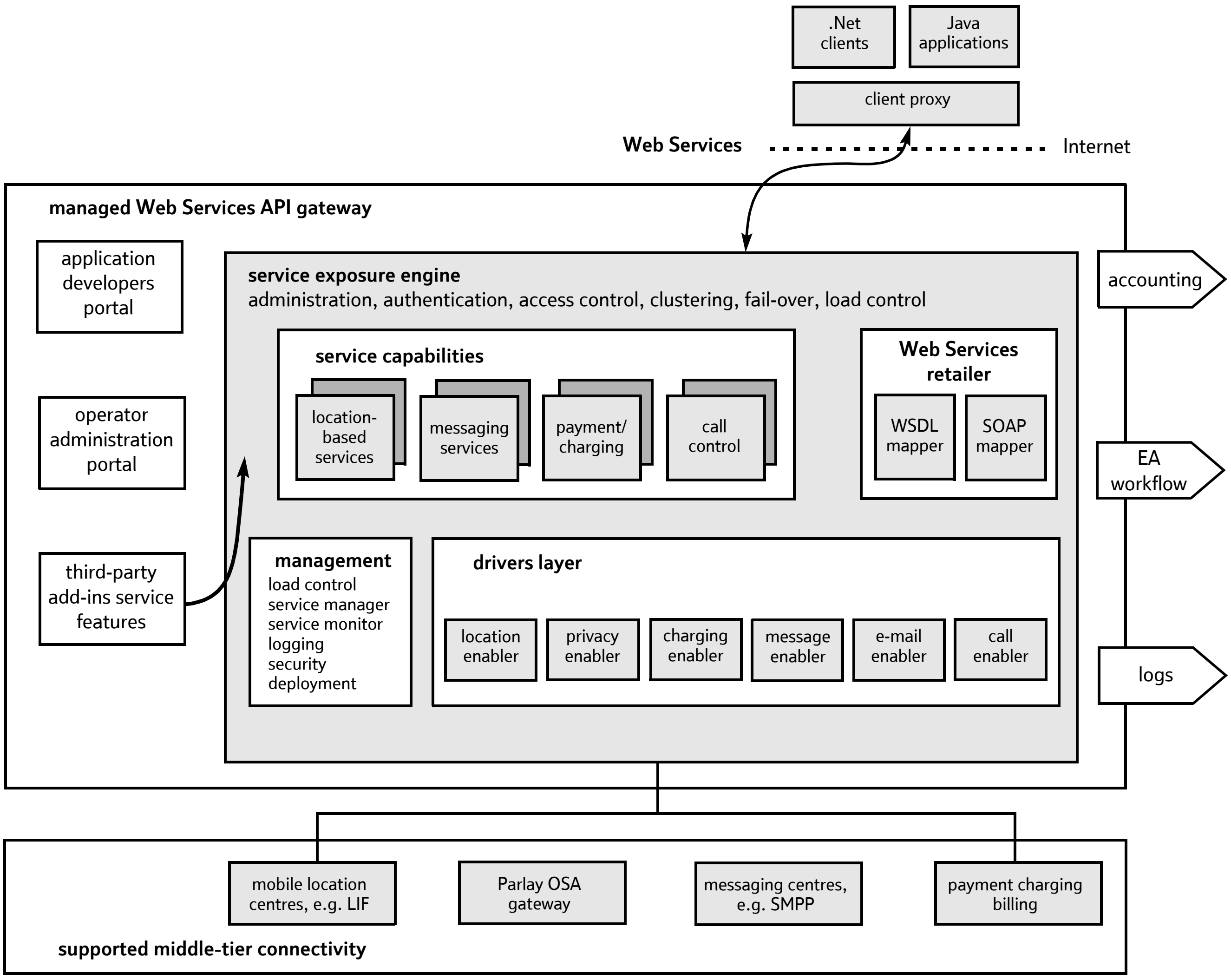

Fig 2 The use of a Web Services gateway to provide a commercial application development platform.

- support for reliable delivery of messages and re-routing or redirection of services,
- priority-based queuing and routing for SLA enforcement,
- monitoring the availability and accessibility of Web Services for potential use,
- monitoring of running Web Services performance and throughput,
- notification of failures and critical events,
- logging of activity and creation and analysis of audit trails,
- starting and stopping access to services for maintenance,
- live upgrade to running services and management of versions and relocation.

All the above features need to be provided in such a way as to be transparent to Web Services clients and providers. Failure to ensure standards-compliant presentation of services that have passed through the environment will create an unacceptable dependency between it and the services that it manages. The design and deployment of infrastructure to provide these would be prohibitively slow and expensive on a per-Web Service basis. One response is to introduce a consistent management platform into the design that can provide these features as an overlay on top of services that just provide core functionality.

One way to provide this overlay is to design and deploy a gateway that wraps simple internal Web Services and provides security and other management services. This is the approach adopted in the managed Web Services gateway platform described above. An alternative approach is to design and deploy the required features within a distributed Web Services management layer (Fig 3).

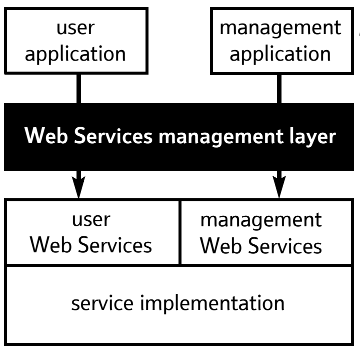

Fig 3 Inserting a management layer between Web Services and client applications.

This approach at first seems counter-intuitive. Why insert a management layer between client and server in what is fundamentally a peer-to-peer computing architecture? A convenient precedent is the Internet itself. IP packets are generally sent between a source address and destination address over what seems to be an entirely passive network. In reality, there is a great deal of packet inspection, prioritisation, routing and translation in transit. This allows management and functionality that is not available or even expressible at the IP protocol layer to be transparently overlaid.

For Web Services, management visibility and control can be achieved by deploying a Web Services management layer (WSML) that intercepts and manipulates SOAP messages between a calling application and a Web Service. This can perform inspection and manipulation of the SOAP envelope or the body or both. Monitoring, analysis and control can then be performed in the management layer without any impact on or modification to the underlying services, some of which may not be under the direct operational control of the business.

Fortunately, it is possible to design a solution that combines the management functions described above with an implementation of current and proposed extensions to the SOAP specification. This allows a modular and highly flexible approach to satisfying both management and operational requirements.

Web Services standards are usually defined as extensions to the basic SOAP specification. These extensions are by design modular and independent of each other. This allows different functions to be plugged together to provide the functionality that is required for a specific usage scenario. For example, Web Services routing is independent from transaction support or security. This allows support for features to be provided at a later date or layered in on top of an un-extended SOAP message.

This design can be implemented as a logical pipeline of processing modules that can insert management features and implement SOAP extensions by intercepting messages and performing operations on them (Fig 4). Being modular, it is relatively easy to

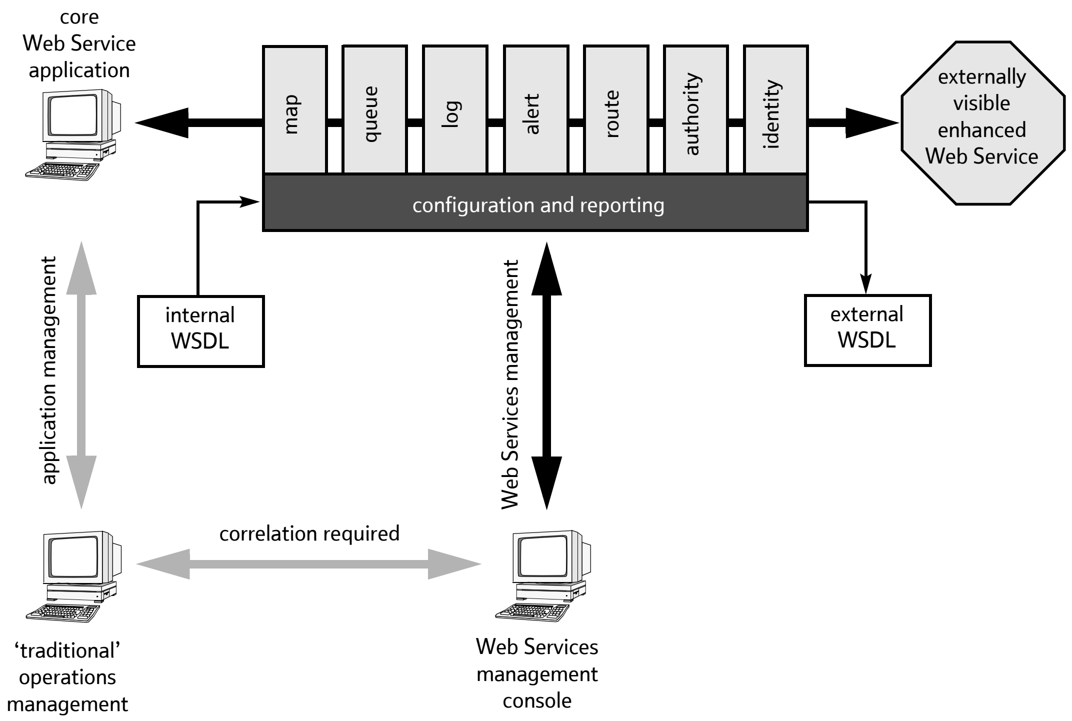
core Web Service application
map
queue
log
alert
route
authority
identity
externally visible enhanced Web Service
configuration and reporting
internal WSDL
external WSDL
application management
Web Services management
correlation required
'traditional' operations management
Web Services management console

Fig 4 A pipeline architecture for providing management and standards support for Web Services.

substitute proprietary functionality with standards-conformant features as these are agreed. This allows the WSML to:

- protect the implementation of the core service functionality,
- overlay Web Services standards extensions as they are agreed,
- map the internal core service WSDL on to that of an external presentation,
- outside the requirements for the standards, define business and security policy for access to the Web Service, enforce these policies and provide evidence of enforcement,
- maintain and demonstrate transactional integrity,
- define and assign transactional liability,
- define, enforce and demonstrate a service level agreement, potentially on a per-client basis,
- specify, monitor and present business and operational management information in support of business objectives,
- manage the operational deployment and life cycle of a Web Service and the underpinning core service implementation within the business level contract for service, detect problems, isolate and correct faults, deploy and manage redundant and resilient solutions.

This pipeline architecture can be implemented at a single point in the solution, which makes it logically equivalent to a gateway, or more commonly is distributed to form a routing network. The pipeline is actually closer to a ‘T’ pipe, with the cross of the T implementing the standards and business or operational rules. The stem of the T is a path for configuration of the pipeline and reporting of events and other management information via a logical ‘management console’. Often this management console is centralised and provides configuration, analysis and reporting. Correlation is required between the management of the messaging layer and the underlying operational applications and platforms.

Examples of products that are designed for distributed deployment with centralised management include:

- Flamenco Networks — Flamenco Network,
- Amberpoint — Web Services Management Foundation,
- Digital Evolution — Management Server.

The impact of managing the messaging layer is significant, extending from the deployment and operations issues presented above up to business level concerns. It allows the collection of raw data from across the enterprise, the refinement of that data into business intelligence and the presentation of the information to business managers. Business management is not passive and it is important that managers are able to use the knowledge that has been gained to control underlying resources.

By abstracting the implementation of capabilities as Web Services, the configuration, reconfiguration, monitoring, reporting and control can become highly standardised and automated. The abstraction of a capabilities ‘external contract’ from its ‘internal implementation’ allows business managers to control the infrastructure from a business perspective. The intelligence and control can be used to enable key business management functions that include:

- creating and maintaining customer relationships,
- monitoring and analysis of customer activity,
- creating commercial offerings,
- defining and enforcing service levels,
- measuring product performance,
- ensuring efficient use of resources and controlling costs.

By logging messages it is possible to collect statistical information about calls to specific Web Services. This can provide a business level view of customer activity and service usage and forms an audit trail for accounting or compliance purposes.

By inspecting the body of messages, it is possible to determine the calling application’s assumed role, identity and the context of the request. This can be used to help ensure that the platform implements business level policy. This approach can, for example, ensure that requests from priority customers, or transactions with high business value, are given priority over ones with lower business value. In the event of service platform degradation on failure or high load, ‘Gold’ customers are served before ‘Bronze’, and high value orders are given priority over non-revenue earning transactions.

Information from the messaging layer can be combined with information derived from other sources to create a Web Services-based ‘enterprise dashboard’ application that collects and displays high-level performance information from distributed information and presents this via a management console. Because the various information sources are abstracted as Web

Services and are loosely coupled, both from each other and the client management application, it is possible to quickly create product or role specific views on the same data. For example, one view for the product manager might show a product's performance in terms of on-line sales over time. By correlating this with Web site page impressions, the effectiveness of a marketing campaign can be assessed by the marketing manager. Another view that includes transaction times can help operations managers monitor and proactively recommend an increase or decrease to the server capacity and network priority assigned to the product.

5. Commercial opportunities for future Web Services platforms

There are a number of business models for commercial exploitation of Web Services. These include the direct supply of functionality via Web Services and third-party supply via the operation of broker environments [14]. In both situations access control, metering, service support and billing could be provided. Through its relationships with suppliers, and trust relationship with customers, BT is in a position to become a broker and to offer a library of components to enable quicker, cheaper and better solution development for Web Services deployment (see Fig 5).

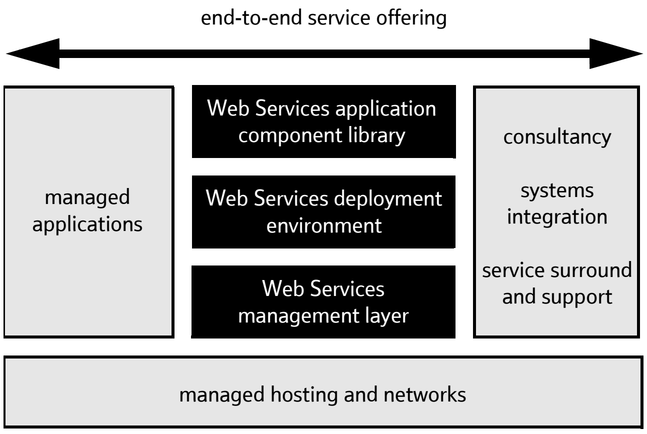
end-to-end service offering
managed applications
Web Services application component library
Web Services deployment environment
Web Services management layer
consultancy
systems integration
service surround and support
managed hosting and networks

Fig 5 Elements of the BT Web Services strategy.

For service providers, such as BT, there is an opportunity to support customer deployment of Web Services by the supply of managed infrastructure. Taking BT as a specific example, it is already recognised as a provider of secure networking to businesses of all sizes. It also provides consulting, professional services, IP networks, hosting, managed server applications and complete outsourced solutions. It is a natural extension of this portfolio to provide products and services to customers that want to benefit from using Web Services.

Early experience suggests that 20% of the effort in creating a commercial Web Services application is expended in defining the commercial opportunity, the service interfaces and server implementations. The remaining 80% is spent on ensuring that the operational and deployment environment is adequate to support business grade service levels [15]. Managing that 80% of the problem for our customers is a business opportunity for BT.

A managed Web Services infrastructure is commercially attractive to customers because it can be layered on top of existing service offerings such as networks and hosting and can reduce the risks associated with early adoption. This is especially true where customers wish to reap the full benefits of Web Services by extending their use from behind corporate firewalls and allowing their secure and reliable connection to the rest of the business community. There are two specific opportunities for managed infrastructure that have been identified, both of which are targeted at multi-site corporate customers in the UK and throughout Europe.

5.1 *Managed deployment environments*

The Web Services Deployment Environment (WSDE) offers a managed environment for the delivery of Web Services solutions via leading application software vendors. Where traditional hosting services typically cover only hardware and operating system software, the WSDE extends the value-add to solutions providers and customers by additionally providing managed application software from leading Web Services vendors on top of that hosted platform, and including management up to and including the applications infrastructure. The WSDE offers both a pilot and production platform enabling customers to get Web Services up and running quickly at reduced cost and risk.

Using experience gained from internal deployment, BT is developing a Web Services test and interoperability suite that will help to ensure that customers' deployed Web Services are truly standards based and available for use in all common application development and deployment environments.

5.2 *Managed Web Services networks*

Addressing the management opportunity described above, BT has launched the Web Services Management Layer (WSML) product. WSML provides the capability to manage the provisioning, security, monitoring, problem identification, and reporting for Web Services transactions. It is ideal for use in a range of situations, including where customers expose Web Services to partners, or connect across disconnected sites (see Fig 6).

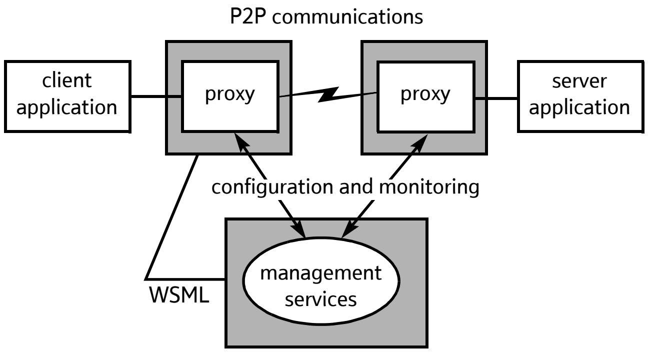

Fig 6 The BT Web Services Management Layer (WSML).

WSML takes raw Web Services created by organisations and creates assets that are ready for deployment. It is offered as a managed service and is architected to operate on a peer-to-peer basis, with facilitated management, similar to that described earlier in this paper. WSML is readily used to implement new standards, as opposed to having to implement them in the Web Services themselves, providing a ‘buffer’ that reduces the shock of their introduction to an enterprise.

The WSML platform manages the process required to provision a customer’s Web Service to their trading partners. It can also be used to manage Web Services delivery between an organisation’s own sites. It addresses general and ‘standard-gap’ issues of reliable delivery, security, provisioning, monitoring, reporting, metering and non-repudiation when using Web Services.

Once provisioned, the platform manages access to the Web Services and ensures the confidentiality of communications. Web Services activity and performance is monitored and recorded for the customer. Alerts are generated for the customer when the end-to-end performance of their Web Services falls out of specification. Self-service tools are provided to allow customers to manage their instance of the service and the peering relationships they have with other Web Services providers and consumers.

Technically, the platform is a managed application proxy designed to provide security, performance, monitoring, problem notification, and reporting services for Web Services communications between two points on a pre-existing TCP/IP network. To provide a managed service, the proxy must be deployed on both the requesting (client) and providing (server) end of a Web Services transaction. The client-side proxy intercepts communications from the client application and directs this to an appropriate server-side proxy. The server-side proxy accepts the communication and routes this to the correct server application for processing. The communication between the proxies is access controlled, encrypted for security, signed to ensure that it has not been altered during transit, and monitored to ensure timely delivery. The pairs of proxies that form the managed link perform these services transparently.

Although the proxies directly communicate with each other, the configuration information required to locate, identify and transact with other proxies is supplied by a centralised management service. The proxies monitor and record communications that pass through them and pass this data back to the management service. Thus communications services are peer-to-peer but management services are hub-and-spoke.

There are a number of advantages of this hybrid approach. For example, there is a reduced likelihood of forming a communications bottle-neck at a central hub. Also, it is possible for communications to be routed though a customer’s internal networks, reducing their need for Internet access bandwidth and the perceived risk of routing sensitive data on an external network. Once configured, proxies can operate independently in the event of a short-term communications failure to the management service.

The proxies are small applications that can be installed by the customer on a firewall, a dedicated server, or on the same hardware that hosts their client or server application. Alternatively, the proxy can be supplied pre-installed on a dedicated managed hardware appliance. This flexibility allows WSML to be positioned as managed software or managed hardware, the latter being potentially easier to incorporate into the customer’s security policy.

WSML can be deployed alongside data networks with a defined quality of service, e.g. using multi-protocol label switching (MPLS), to offer a combined communications infrastructure. This allows enhanced performance security and availability at the network layer with provisioning, routing, auditing and reporting at the Web Services application layer.

Notably, analysts have been supportive of BT’s approach to offering Web Services management, concurring that this capability aligns well to the role of a network provider. MCI have subsequently stated their intent to develop Web Services within the core of their network and it is believed they and other service providers will deliver similar management functionality to market over a period of time.

References

1 Boden T: ‘The grid enterprise — structuring the agile business of the future’, BT Technol J, 22, No 1, pp 107—117 (January 2004).

2 Wald L: 'Quick Service — Achieving Business Agility Through the Development of Service-Based Processes', The IT Journal, Q2/(2001) — http://www.hp.com/

3 Bloomberg J: 'How Web Service Management is the Key to the Service Oriented Architecture', zapthink report ZTR-WS106 (November 2002) — http://www.zapthink.com/

4 Fenn J, Natis Y, Sinur J and Linden A: 'The Integrated Enterprise from 2003 to 2012', Gartner Inc Research Report SPA-18-8139 (December 2002) — http://www.gartner.com/

5 Calladine J: 'Giving legs to the legacy — Web Services integration within the enterprise', BT Technol J, 22, No 1, pp 87—98 (January 2004).

6 Infravio — http://www.infravio.com

7 Actional — http://www.actional.com

8 Kearney P, Chapman J, Edwards N, Gifford M and He L: 'An overview of Web Services security', BT Technol J, 22, No 1, pp 27—42 (January 2004).

9 Web Services Interoperability — http://www.ws-i.org/

10 Bloomberg J: 'How Service-Oriented Development will Transform the Software Industry', zapthink report ZTI-WS101 (April 2002) — http://www.zapthink.com/

11 Yates M J: 'Enabling applications deployment on mobile networks', BT Technol J, 21, No 3, pp 134—140 (July 2003).

12 Lofthouse H, Yates M J and Stretch R: 'Parlay X Web Services', BT Technol J, 22, No 1, pp 81—86 (January 2004).

13 Parlay X Specification version 1.0, The Parlay Group 2003 — http://www.parlay.org /

14 Bilchev G et al: 'EURESCOM Project P1209', — http://www.eurescom.de/

15 Walker S: 'BT Web Services market research', Private communication (2003).

After gaining a BSc and PhD in Colour Chemistry from the University of Leeds, Julian Hill joined BT in 1984 to research the development and processing of materials for advanced electro-optic applications.

He has since held a wide variety of positions within the company, before taking on his current role at Adastral Park as a technology strategist within BT Global Services' Web Services team.