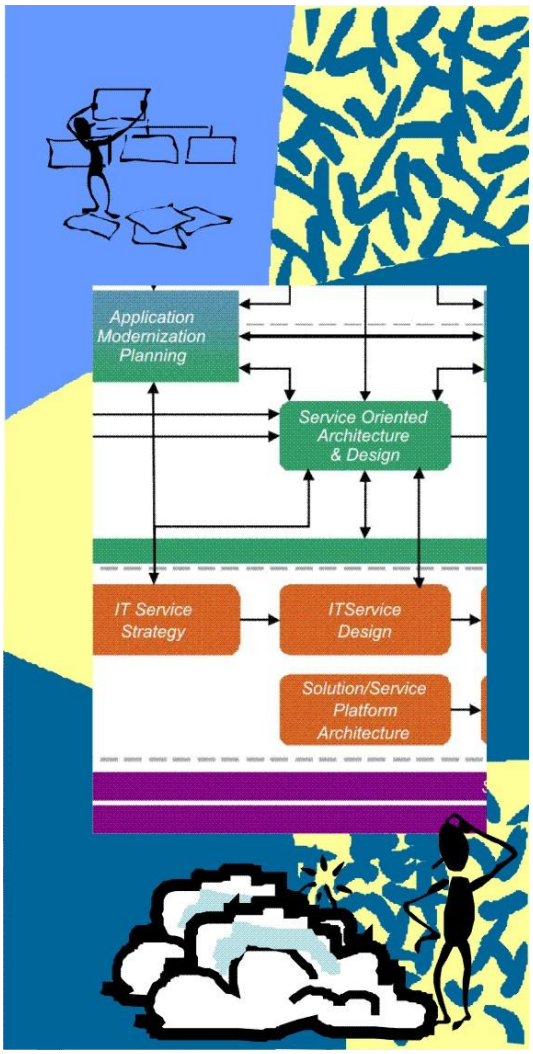
CBDI Journal



Independent Guidance *for* Service Architecture and Engineering

Best Practice Report

Service Portfolio Planning and Architecture for Cloud Services

In this report, we show how the CBDI-SAE approach can be used and extended to architect for Cloud Services. We extend our current guidance with new and refined classification systems, diagrams, policy types and techniques designed to promote visibility and good governance over Service Portfolio Planning activities and Cloud Services provisioning.

By Lawrence Wilkes

Originally published in the April 2010 edition of the CBDI Journal

Service Portfolio Planning and Architecture for Cloud Services

Cloud Computing is concerned with deployment, but introduction of Cloud Services cannot be a purely technical deployment matter. There are numerous considerations that may impact on all the Stakeholder Views. In this report, we show how the CBDI-SAE approach can be used and extended to architect for Cloud Services. We extend our current guidance with new and refined classification systems, diagrams, policy types and techniques designed to promote visibility and good governance over Service Portfolio Planning activities and Cloud Services provisioning.

By Lawrence Wilkes

Introduction

The interest in Cloud Computing and the provision of Cloud Services has grown significantly. However, terms such as "Software as a Service" or "Public Cloud" can still be a bit misleading and vague. What exactly is the type of capability being offered by a Service? Who is actually providing it? After all, aren't all Services provided by software?

In CBDI-SAE we have defined a structured approach to identifying Services and in particular classifying them into different types with relevant policies and rules guiding the development of the Service Architecture. Moreover, our Service Portfolio Planning (SPP) process guides the traceability between the architecture Views from Business, Specification, through Implementation, to Deployment, providing a business centric architecture on which to make decisions about provisioning the Services in the portfolio, and their implementation.

In this report we set out to establish how our SPP and Service Architecture approach provides a framework for better understanding of the role of Cloud Services and decision making for their use.

Cloud Services

What are Cloud Services? Simply put, they are Services provided where:

- Their physical location is transparent to the Service consumer.
- The computing infrastructure is provided on a shared basis to derive economies of scale, or to improve agility, scalability or reliability.

The core concept of a Service is that it hides from the Service Consumer the complexity of how the capability is implemented by the Service Provider. Hence the focus in SOA is primarily on *what* capability the Service provides, and not on *how* it is provided.

The Service can virtualize the capability. With appropriate technology it can even be transparent to the Service Consumer *where* the resources are located or *whose* resources are used. By removing the constraints of tightly coupled resources, Service Providers and Service Consumers have greater agility to use alternative resources as long as the Service Specification continues to be met.

The concept that Services are provided 'somewhere in the cloud' has always been central to our vision of SOA and we often used the cloud metaphor to illustrate this. Figure 1 for example was published back in 2001[1]. Even prior to that some 15 years ago in our early CBD research we presented the notion of application solutions assembled from a 'cloud of services' provided by software components implemented in different technologies, though the notions of the infrastructure provided by a public cloud were not developed then.

At the time, there was a fair amount of skepticism from any audience we presented this to as to whether this would ever become reality, or at least widespread. Cloud computing is only now becoming mainstream because the industry and its customers have by and large achieved a basic maturity in SOA that enables this more sophisticated architecture.

Whilst the premise of cloud computing and the use of services provided by external entities is now more widely accepted, many of the concerns raised then still remain – including security, risk, and worry of dependency on external entities.

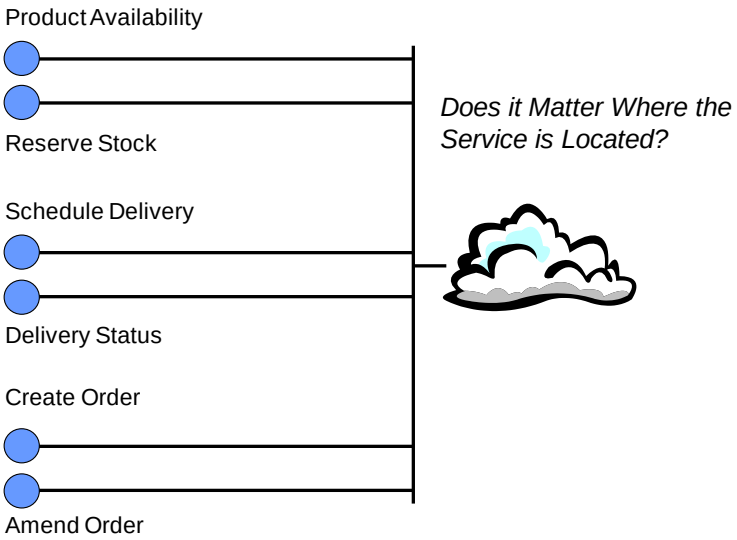


Figure 1 – Does it Matter Where the Service is Located?

Cloud Service Classification

There is a high level of convergence throughout the IT industry and its customers around Cloud Service classification as follows:

- Software as a Service. The capability provided to the consumer is to use the provider's applications running on a cloud infrastructure.
- Infrastructure as a Service. The capability provided to the consumer is to provision processing, storage, networks, and other fundamental computing resources where the consumer is able to deploy and run arbitrary software, which can include operating systems and applications
- Platform as a Service. The capability provided to the consumer is to deploy onto the cloud infrastructure consumer-created or acquired applications created using programming languages and tools supported by the provider.

Meanwhile the deployment or availability of the services may be described as

- Public Cloud. The cloud infrastructure is made available to the general public or a large industry group and is owned by an organization selling cloud services.
- Private Cloud. The cloud infrastructure is operated solely for a single organization. It may be managed by the organization or a third party, and may exist on-premises or off-premises.
- Community Cloud. The cloud infrastructure is shared by several organizations and supports a specific community that has shared concerns (e.g., mission, security requirements, policy, or compliance considerations). It may be managed by the organizations or a third party and may exist on-premises or off-premises.
- Hybrid Cloud. The cloud infrastructure is a composition of two or more clouds (private, community, or public) that remain unique entities but are bound together by standardized or proprietary technology that enables data and application portability (e.g., cloud bursting for load-balancing between clouds).

The above are taken from the Cloud Computing guides published by NIST[2], and subsequently adopted by the Cloud Security Alliance[3], which we recommend reading.

Service Portfolio Planning and Cloud Services

Service Portfolio Planning (SPP) is an overarching process encompassing the identification of Services, arranging them and associated artifacts in the Service architecture views, and making planning and policy level decisions about their provision, deployment and usage based on business and IT strategy and requirements.

To some extent, the portfolio planning decisions as to whether to use a Cloud Service or capability should be little different to that for other Services. However, the introduction of Cloud Services will often be a deviation from current standard practices for many organizations, and consequently they may need to apply more rigor to their decision making processes and place greater consideration over factors that today may normally just be accepted as a given, or the de facto state in their organization.

Figure 2 – Service Portfolio Planning Decisions for Cloud

As illustrated in Figure 2, decisions to use Cloud Services or Cloud Deployment in the Service Architecture will be based upon a number of considerations of the Cloud option, or of the alternatives available, as further outlined in Table 1

Consideration	Cloud	Alt A	Alt B
Financial: What are the financial costs and benefits?			
• Cost benefit analysis			
• Initial Investment required			
• Ongoing costs			
• Charging and cash flow comparisons			
Agility: How is business or IT agility improved, or compromised?			
• Scalability to meet changes in demand			
• Transferability to alternatives			
• Portability between alternatives			
• Impact on maintainability			
Risk: What potential risks are involved in using, or not using it?			
• Risk analysis considering			
○ Reliability			
○ Security			
○ Back-up and recovery			
Capability: What capabilities are required, and who possesses them, or where?			
• Manageability, such as			
○ configurability			
○ the extent of auditing and logging of transactions, state changes, etc.			
• Internal or External capability, On- or off-premises			

Table 1 – Cloud Service Usage Decisions

As mentioned earlier, standard Cloud deployment types cover Public, Private and Community. Classifying the key artifacts in the architecture as Public, Private or Community should be done regardless of whether they are using Cloud Computing or not. It simply classifies the pattern of usage applied to the artifact.

Hence this usage pattern should be considered across all of the service architecture views, from business to deployment, not just the run-time deployment. For example, it is just as important to make decisions about public or private usage of business service provision in the Business Architecture, as it is about their provision in software at run-time, as policy may dictate that one should not overrule the other.

In the implementation and deployment view, the decisions regarding usage patterns may need to be decomposed further to map and classify each domain onto the various

available capabilities for implementation and deployment, such as compute, data storage or message delivery (data in-flight).

As part of the governance activities, we recommend organizations should therefore identify a set of policies that govern decisions in the matrix shown in Table 2, and further detailed in Table 3. This provides a check list, as to what are the organizations policies regarding the use of public, private or community usage of resources in each area. As well as security and usage policies, this matrix can also be used to set or make policy decisions based on economic factors and risks.

As part of this policy setting exercise, we recommend organizations identify their own set of usage patterns, and if necessary specialize the 'standard' set, and go beyond the basic public, private, community model.

Domain	Public	Private	Community
Architecture View			
Business			
Specification			
Implementation			
Deployment			
Infrastructure Capability			
Compute			
Data Storage			
Message Delivery			
Manageability			
Logging/Auditability			
Security			
Reliability			
…			

Table 2 – Usage Patterns Policy Matrix

Policies

There are clearly general security policies that need to be in place relating to issues such as access permissions, authentication, and the encryption and privacy of data

One might argue that there should be little difference with regard to the security of Public or Private cloud capabilities that are used, as there is no such thing as a totally secure private capability anyway. Everything should be treated as potentially at risk.

However, it is also true that in the case of an instance of public cloud capability the desired level of security may simply not be attainable because it lacks the necessary features, and hence its usage is not permitted.

Table 2 should have additional dimensions to reflect how the policy varies in accordance with the key policy subjects (CBDI-SAE Meta Model Types). For example, does the use of public or private cloud vary according to the business domain? E.g. the customer domain can use the public cloud, but accounts must use private.

Policy Types	Organizational Consideration	Policy Subject	Provision	Consumer
Business Architecture				
Sourcing Organization Consumer Organization	Which organizations can... provide and consume a business capability or service	Business Capability Business Service	Business Organization	Business Organization
Specification Architecture				
Sourcing Organization Consumer Organization	Which organizations can... provide and consume a specified service	Service Service Dependency	Service Provider Organization	Service Consumer Organization
Implementation Architecture				
Consumer Software	Which Automation Units can... consume a specified service	Service	Service	Automation Unit
Automation Unit Supplier	Which organization can... Provision the Automation Unit	Automation Unit	Automation Unit	Service
Deployment Architecture				
Compute Service Executer	Which organizations can... Host or execute the deployment artifacts at run-time	Deployment Node Endpoint	Hosting Organization	Service Provider Organization Service Consumer Organization
Data Storage	Which organizations can... Host databases at run-time	Deployment Node	Hosting Organization	
Message Delivery (data in-flight)	Which organizations can... Process messages at run-time	Message	Intermediary Organization	

Table 3 – Example Policies

A range of other example policies is shown in Table 3, covering sourcing and usage (consumer), and operational policies, and primarily specifying the organizational aspects in these examples, such as which organizations are permitted to source, consume, or host a Service.

	Host	Provider	Intermediary	Consumer
Logical or Business Participants	Hosting Organization	Service Provider Organization	Service Provisioner Organization	Service Consumer Organization
Physical Participants	Software and Service Hosting Platform	Service Providing Software	Service Broker	Service Consuming Software
Role	Contracts to host any other participants deployment artifacts	Contracts to provide the actual Service	Contracts to provide the Service - from a logical perspective	Contracts to use the Service
Provides	Physical Infrastructure Physical (run-time) Service	Logical Service	Logical Service	
Consumes			Logical Service	Logical Services
Organizational Context	Internal			
	External			
Location Context	On-Premises			
	Off-Premises			

Table 4 – Service 'Supply Chain' Participants

In this context, it is useful to understand the various roles played by organizations.

Traditionally we might think about the Service Consumer and Provider as the two most obvious roles. But of course it is more complex than that as detailed in Table 4.

Firstly we need to distinguish between the business participants – i.e. the organizations responsible for Service Provision and Consumption – and the physical participants (software and hardware) actually providing and consuming the services at run-time.

In many instances, the Service Provisioner in an organization may be acting as an intermediary. As far as the Service Consumer who is assembling a solution is concerned the Service Provisioner is the Service Provider. However, the provisioner may have contracted with an external business entity to actually provide the Service being consumed. As such the provisioner is just playing an intermediary role.

Moreover, any of these participants may be contracting with a hosting organization to host the deployment of their service or solution and associated artifacts.

Hence, whilst an organization may contract with a Service Provider organization to logically provide the services, the Service Provider may in turn contract with the hosting organization to physically provide the service at run-time.

In an organizational context, any of these participants can be either internal or external, depending on how you view them from your individual role within the service supply chain. Similarly, the location of any of the physical participants may be on or off-premises.

Service Architecture Views

In each of the Service Architecture views architects need to make a number of modeling considerations as shown in Table 5 with regard to how they add Cloud Services into the architecture.

View	Modeling Consideration
Business	For traceability, the associations between Cloud Services and objects in the Business Model should be shown, where relevant. For example in the case of Cloud Services classified as Core Business and Process Services that should have associations with Business Process, Business Type, or Business Service in the business model. A Cloud Service may introduce an industry standard business process; or a business process and semantics that are shared across a supply chain or information ecosystem.
Specification	The Cloud Service needs to be classified and placed accordingly in the Specification Architecture. See mapping in table 6 The contracts between the provider and consumer may need to be expanded to capture additional detail regarding the provision and consumption of Cloud Services, such as security classifications, charging mechanisms, and to record associations between different contract types (Service Specification, SLA, Commercial Terms, etc). At the same time, the level of detail in some areas of the contracts such as implementation instructions may decrease.
Implementation	If the Cloud Service is an implementation-only consideration, that is it is used to implement a Service in the Specification Architecture, but not visible in the Specification architecture itself, then it is only modeled as is part of the internal architecture of the Service. For example an Infrastructure or Utility Service that is used exclusively to implement a Core Business Service
Deployment	In a private cloud, the nodes hosting the service endpoints and the deployment artifacts would be modeled in the normal way, though little may be known about the deployment architecture of an external, public cloud service. However, the external behaviors are of interest – such as configuration management, operational management, (particularly where Services are provided as a mechanism to accomplish this), compliance with standards, and hence level of portability that enables a switch between providers

Table 5 – Modeling Considerations

Service Specification Architecture and Cloud Services

As we said earlier, the generalized terms used to classify Cloud Services can be a bit misleading and vague when it comes to producing a more exact model of the Service Architecture. What exactly is the type of capability being offered by a Service?

Hence it is better to classify what type of capabilities is provided by the Cloud Service according to CBDI-SAE Service Layer classifications, and place into the appropriate SAE layer. Table 6 illustrates how Cloud Service types might be mapped to CBDI-SAE Service Classifications.

Cloud Classification	CBDI-SAE Service Layers	Comments
Software as a Service (SaaS)	Fairly vague term, given all cloud services are software. So useful to decompose this a little more.	
• Application	Capability Service	Most often referring to a whole software application delivered 'as a Service' If the Application provided a single, coarse grained Service, then it might map to a CBDI-SAE Capability Service: *Designed to support a particular business capability (CBDI-SAE).* This offers a coarse-grained capability that may encapsulate many other business services (core business, process)
• Application	Process Service Core Business Service	If the application offers several distinct services, then it may be more appropriate to map it to a mix of core business and process services as appropriate
• Business 'Process'	Process Service	Some commentators propose the notion of 'Business Process as a Service'
• Business 'service'	Core Business Service	Where the service is managing some business type – such as customers, orders, products.
• Business 'function'	Utility Service	Many business functions provided 'as a Service' may fall into this category – providing useful utilities such as financial calculations.
Infrastructure as a Service (IaaS)	Infrastructure Service	*provides technical capabilities, rather than performs business logic (CBDI-SAE)* Most relevant to the implementation view For example, hosting, mediation, security, management, or registry Services. See our model of delivery life cycle and operational life cycle infrastructure[4].
Platform as a Service (PaaS)	Infrastructure Service	A coherent set of related infrastructure services designed to work together to provide a 'platform' on which to develop, test and deploy software and services, rather than individual infrastructure capabilities. As such, some of the Services provided may be as

Cloud Classification	CBDI-SAE Service Layers	Comments
		relevant to the delivery life cycle as to the operational life cycle From a CBDI-SAE perspective, same as IaaS. However, some might also consider the notion of a Business 'platform' - a coherent set of related business services, with tools for example to support modeling and executing business processes that use the business services in the platform.
n/a	Underlying Services	Many Cloud Services may end up being mapped as Underlying Services (see narrative)
n/a	Exclusive Service	Many Cloud Services, especially Infrastructure may end up being mapped as Exclusive Services – that is they are used exclusively within the implementation architecture of a Service, and not published for others to consume.

Table 6 – Mapping Cloud Service and CBDI-SAE Service Classifications

Architects may need to consider how many 'business services' are mapped on a case-by-case basis. Often the capability they provide may not conveniently map to CBDI-SAE Process, Capability, Core Business or Utility Services. They may be coarse-grained and contain a number of different capabilities through different operations.

Consequently, they may need to be classified as Underlying, and only made available to service consumers via higher layers that effectively wrap them and represent them as services that conform to the CBDI-SAE architectural approach.

Worked Example

Let's consider a worked example showing how different types of Cloud Services are used in the Service Architecture. Of course, there may be much more complexity to this in real life - this is just a simplified example.

In Figure 3, the Customer Management Service and Payment Service are both Cloud Services. Whilst there is no explicit indication of deployment approach in the Specification Architecture, we will show that this becomes an important consideration. Even though the Specification Architecture is, by definition, independent of technology and implementation approach, the choice of service and the specification details (templates) will necessarily vary for Cloud services.

The Customer Management Service is effectively an 'application' and promoted by the 3rd party provider as SaaS.

There is only one coarse-grained service that is used by the CRM and Sales Process Services. Hence it has been classified as a Capability Service. This has multiple-operations supporting different activities to manage information about customers.

The Tax Calculation Service is a simple one-function utility – promoted by its provider as a Web Service. As this Service will be used in several places across the organization's enterprise Service Architecture, it has been classified as a Utility Service. It isn't managing 'Tax' information, so it isn't a Core Business Services. It simply performs the calculation.

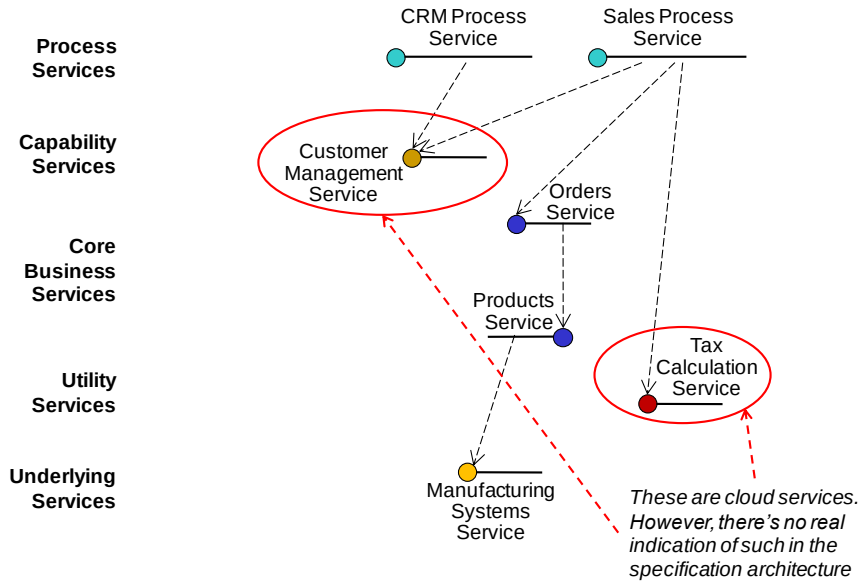


Figure 3 - Specification Architecture for Cloud – Coarse Grained Capability Service

The real impact here will be on the Service Specifications themselves, in that it will be inherited from the provider rather than specified in-house. Assuming that is, that the decision is made early on to outsource these without producing a detailed specification first.

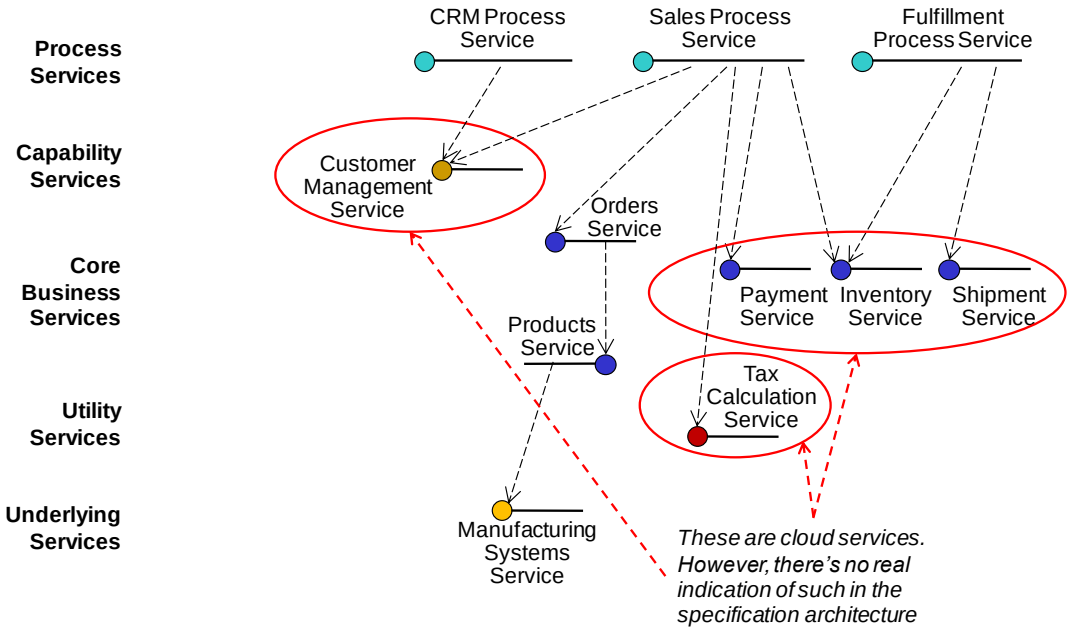


Figure 4 - Specification Architecture for Cloud – Individual Core Business Services

In Figure 4 however, the capabilities required to deal with payments, inventory and shipments are provided by Amazon Web Services[5] as individual services, and each one is directly consumed by the Process Services. It is clear now that these services map to the Business Types they are managing, and so they are classified as Core Business Services.

In Figure 5, we now show the classification of the Services according to the usage patterns that we outlined in Table 2. To enrich the example, we have added a new Material Requirements Planning (MRP) Process Service that has a dependency on a Manufacturing Inventory Service. This is used by participants at the manufacturing end of the supply chain to determine stock levels of raw materials and components used in the manufacturing process and to manage their replenishment.

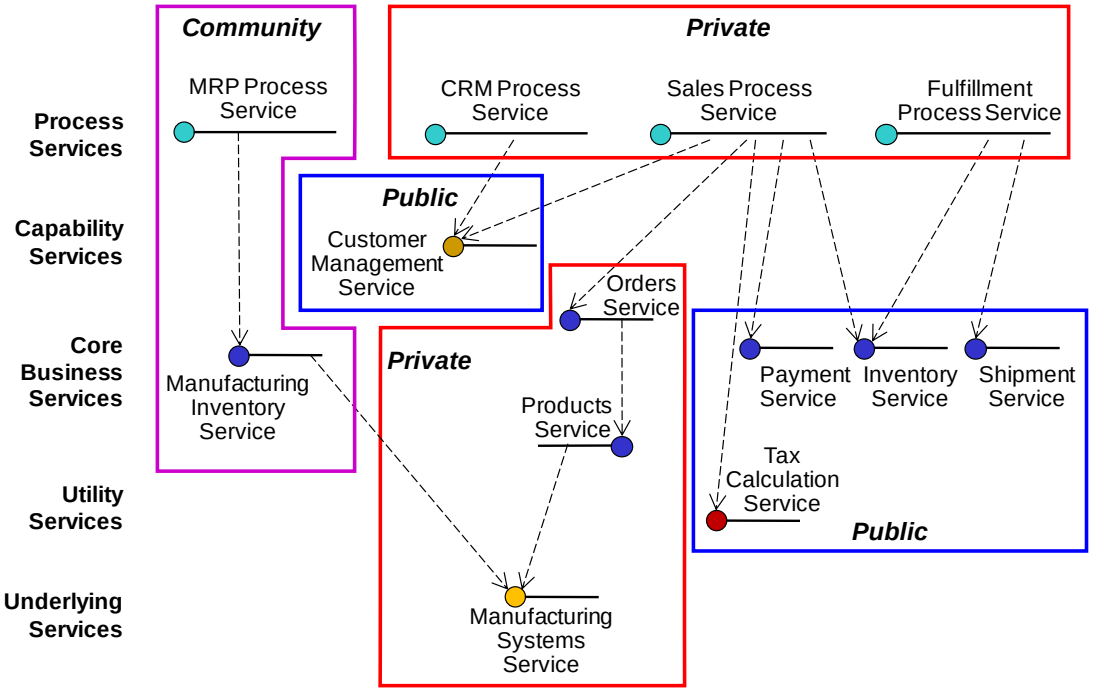


Figure 5 - Specification Architecture – Usage Pattern Classification

In this example, as there is a many-to-many relationship between suppliers and manufacturers in the same industry they collaborate as an ecosystem to more efficiently match demand. Consequently the Services provided are open to any trusted participant in the community, and hence classified as Community.

The 3rd party Cloud Services identified in Figures 3 and 4 are classified as Public, whilst the Orders, Products and Manufacturing Services are Private.

In this example, an early planning decision has been made to use these 3rd party Cloud Services. However, decisions in the Business Architecture should have already determined whether Services to support these Business Domains or Business Types are eligible for Public usage or not, and the service architects should not be overriding those decisions just because they have found a candidate Service. Or at least without seeking approval for deviation from policy first.

Service Specification and Cloud

With regard to detailing the Service Specifications of the individual Services then,

- First determine the appropriate Services classification
- Public or Community Business Services (Process or Core) acquired from 3rd parties still require a specification to be published to aid Service Consumers
- For Underlying Services (see rules for determining underlying[6]), decisions as to whether full specification needed depend on how it is going to be published
    - Is it going to be used exclusively in some part of the architecture? Then a full specification may not be needed
    - Is it going to be widely shared and available to multiple Service Consumers? Then a full specification is required
- For Infrastructure Services
    - If it is an Exclusive Service, used within the implementation architecture of a Services, then the instance doesn't require a full specification if it is not going to be published.
    - However, many infrastructure services may be provisioned in order that they are widely shared. So the 'type' needs a full specification, so that consumers know how to go about using it.
    - That is, if you build a customer data service using a generic cloud database service, then the specification of the customer data service doesn't need to be published, as it is used exclusively inside the Customer Service component.
    - Whereas, the generic cloud database service needs a full specification so anyone can use it.

Early Service Portfolio Planning Decisions are also key;

- Try to identify early in the service life cycle whether an identified Service is a good candidate for either cloud provision or cloud deployment.
- Don't over specify those that are identified as suitable for public or community provision - the available Services should already have a specification
- You may specify the Service sufficiently in order to do the first cut evaluation. How do you select candidate Services if you cannot compare required and provided specifications?
- You may still wish to add detail to that specification provided by a 3rd party for internal publication so that it can be consumed by solution assemblers for example – the documentation accompanying cloud service may not meet your specification policy
- If you decide to classify the Cloud Service as an Underlying Services, then you still need to specify the Core Business and Process Services that are consuming it and in turn making the capability available to solution assemblers.

Service Implementation Architecture and Cloud Services

In the Service Implementation Architecture we now model the implementation of the Services in the form of Automation Units. Using the example in Figure 4, as we now know the implementation is provided by Cloud Services, the Automation Units have been Stereotyped as «cloud» in Figure 6.

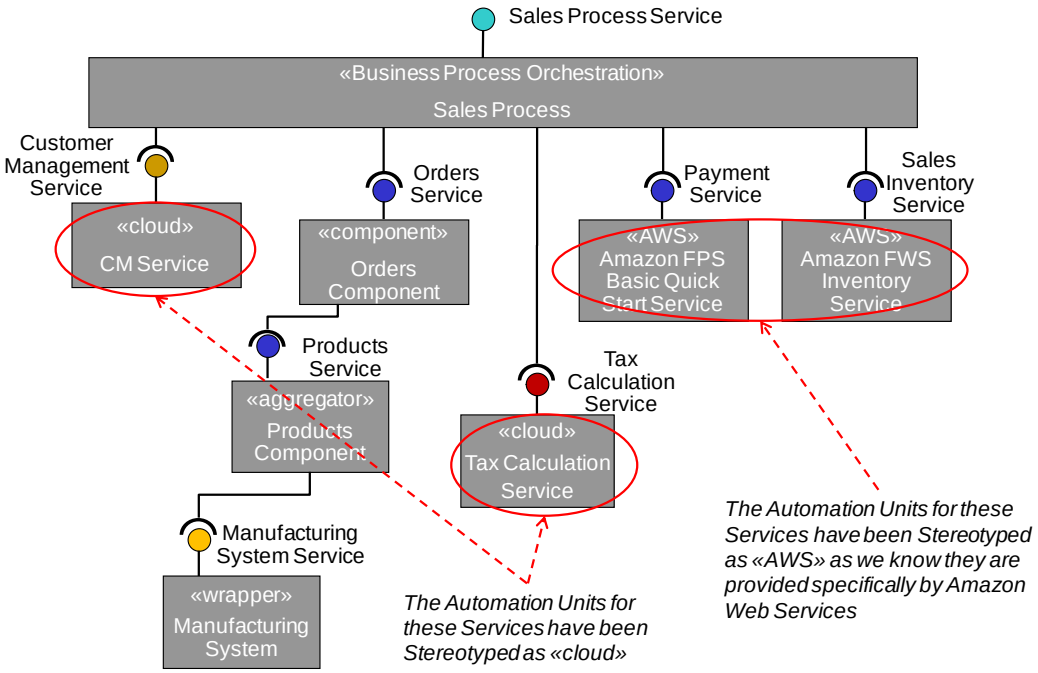


Figure 6 – Service Implementation Architecture Showing Cloud Implementation

In fact there is not much more the consuming organization can add to the Implementation Architecture about the implementation of these Services as the consuming organization has no knowledge of the actual implementation, nor need it have.

It is now readily apparent that these are cloud services. Of course, until cloud became the fashionable term, we might just have labeled them «external service» to denote that the implementation as far as the consuming organization is concerned is just a service, and the consuming organization has no knowledge of the provider's actual implementation itself.

As such, you could question whether it is actually necessary to stereotype them as «cloud» at all, but it may help convey some understanding and provide the linkage to the IT Service Design.

Equally, you may decide to be even more precise about the labeling and label them as Amazon Web Services (AWS), and explicitly name the particular Service used.

Internal Implementation Architecture

Figure 7 shows the Internal Implementation Architecture of the Orders Component. In this IaaS example, a relational database capability – Amazon RDS[7] - is provided as a Service by Amazon and is used to persist Orders.

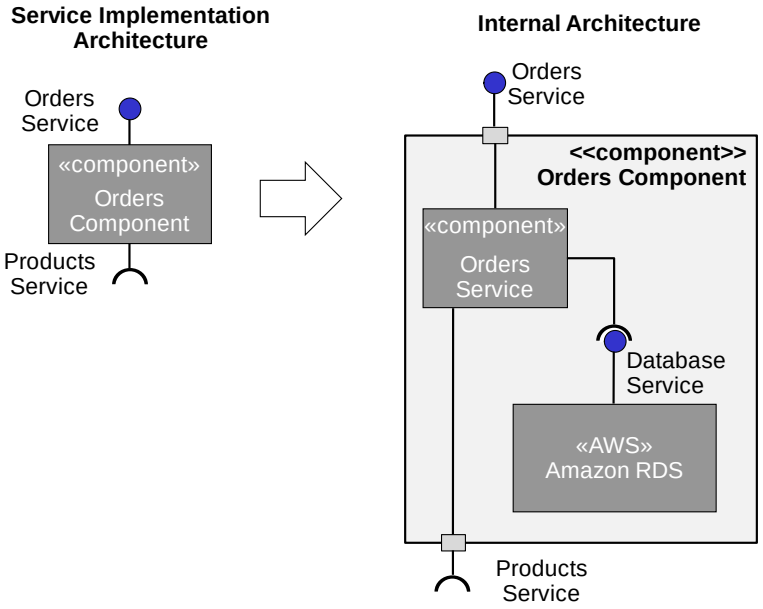


Figure 7 – Internal Implementation Architecture Showing use of Cloud Service

As this service is only part of the internal implementation of the Orders Service, then it is not necessarily modeled in the Service Specification or Implementation Architectures, only in the Internal Architecture. It would also be classified as an Exclusive Service - as you do not want others consuming it directly.

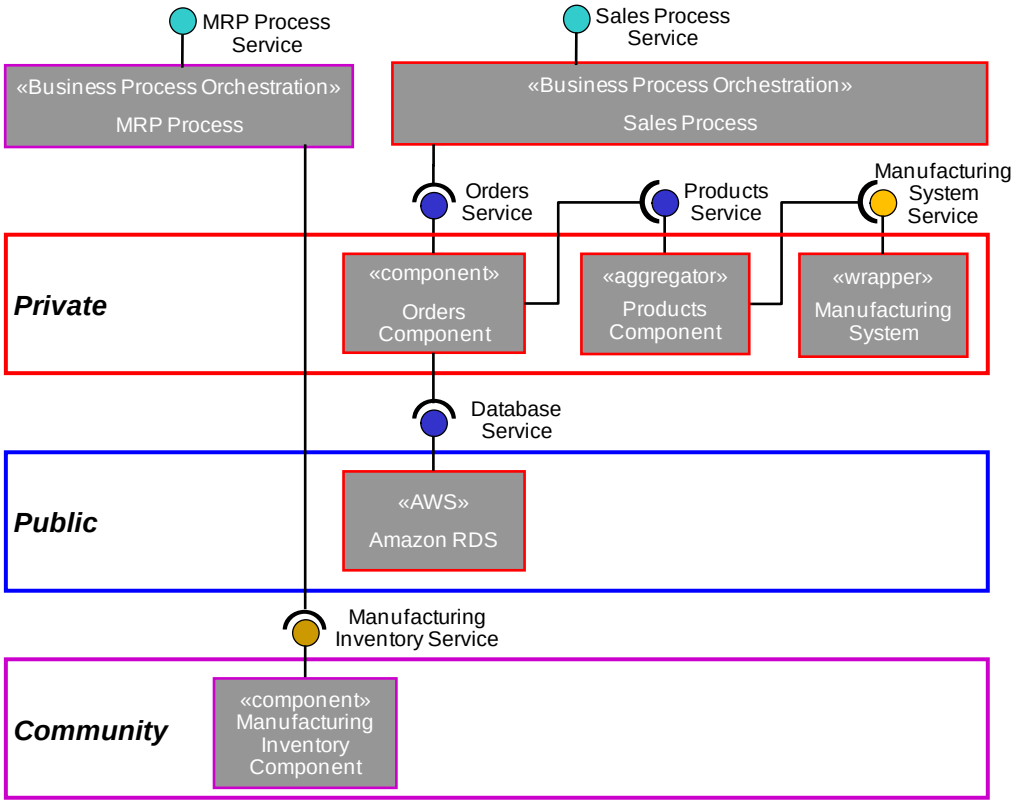


Figure 8 – Service Implementation Architecture Arranged into Usage Pattern 'Layers'

Figure 8 shows the classification of the Automation Units in terms of usage patterns

To simplify the diagram we have put aside the public services shown in the Service Specification, as we know their implementation is using the public cloud, and just show the provision of the Amazon RDS as Public.

We have arranged them into layers in Figure 8 as it is a useful way to communicate the concepts. The ability to produce such diagrams depends on the capabilities of the modeling tools used. The important thing however, is that this information is captured as meta data within the Implementation Architecture documentation.

Deployment Architecture and the Cloud

In the Deployment Architecture the distribution of the various implementation artifacts to the various nodes on the network is modeled.

Figure 9 shows a standard Deployment Architecture using UML 2.0 modeling notation (apart that is from the cloud icon…).

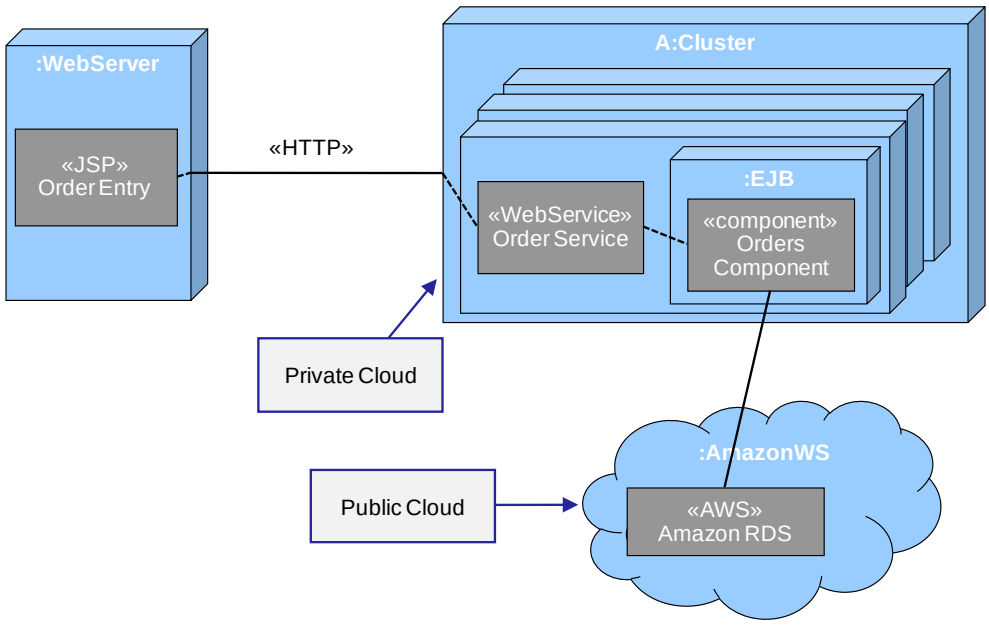


Figure 9 – Deployment Diagram Illustrating Cloud Node

To provide operational agility, increase performance and reduce costs, organizations may make use of server clusters, often referred to as virtualization, or in the context of this report the basis of a private cloud.

In this case, the actual instance of processor node will not be known in advance. The BPEL engine running the Sales Process will be deployed on-demand to any processor in the cluster, or scaled to many processors to enable load balancing.

It is then the cluster server's responsibility to pass the incoming order service requests on to the appropriate node that is currently able to execute the sales process.

Whilst the actual instance of processor node may not be known, the type of processor node or execution environment can still be specified to indicate compatibility. For example, all Enterprise Java Beans must be deployed to an EJB container.

Where the Public Cloud is used, then apart from the Service Endpoint little will actually be known about the deployment environment. However, architects can still add a node to the deployment diagram as a reference point for external services, and this can be named or stereotyped accordingly

In this example the Amazon RDS is shown as deployed to the Public Cloud

The concept of Cloud Deployment is obviously much more relevant at this stage and architectures start to consider which implementation artifacts should be deployed to Private, Public or Community Clouds.

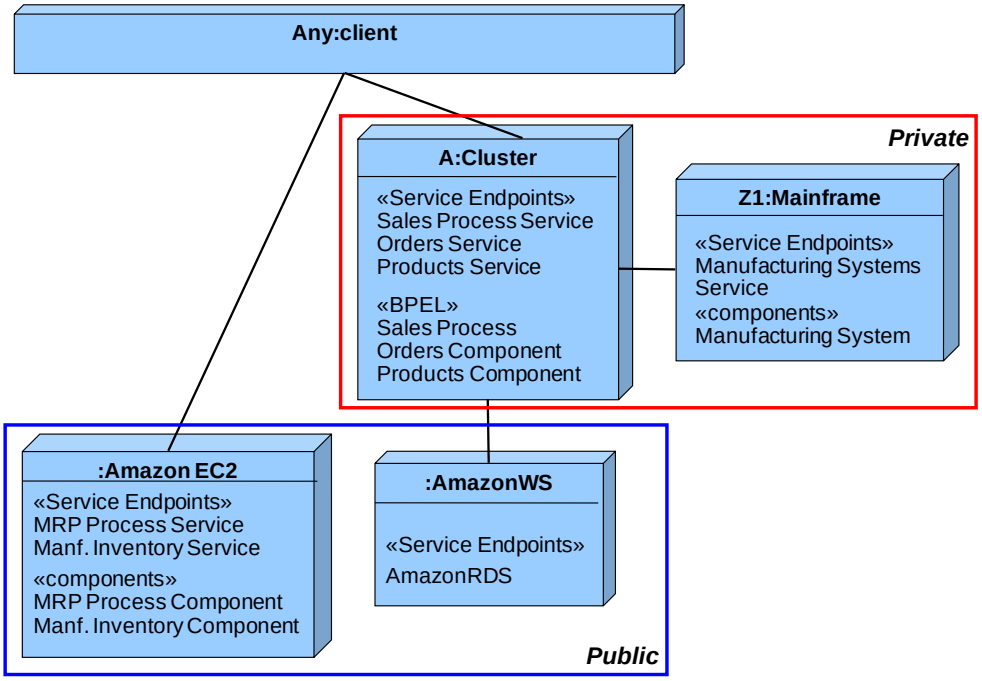


Figure 10 – Deployment Architecture Showing Cloud Deployment Type

At this point, the usage pattern classification of an artifact may be changed from the pattern used for the Specification and Implementation Architectures. This is not a breaking or deviation of policy, but is exactly what decisions in Table 2 are designed to permit.

That is, usage of a Service may be deemed as Private in the Specification Architecture, in that its usage is only internal, but it could nevertheless be deployed to the Public Cloud, providing the appropriate security levels can be achieved and that its usage remains private in the context of the consuming organization.

Also at this stage it may be more relevant to talk specifically about Cloud Deployment Types rather than usage patterns, and to apply them to the more detailed breakdown of infrastructure capability as shown in Table 2 – i.e. Compute, Data Storage and Message Delivery

Hence, figure 10 shows the MRP Process Service and the Manufacturing Inventory Service and their Automation Units deployed to the public cloud, in this instance using the Amazon Elastic Compute Cloud[8] (EC2). As long as the necessary security capabilities can be provided to ensure that usage is only by the trusted community, then the requirements of the usage pattern in the Specification and Implementation

architectures are still met. That is, the deployment may now be Public, but the usage is still classified as Community.

	Infrastructure Managed by	Infrastructure Owned by	Infrastructure Located	Accessible and Consumed by
Public	Third party provider	Third party provider	Off premise	Untrusted
Private	Enterprise or Third party provider	Enterprise or Third party provider	On premise Off premise	Trusted
Community	Enterprise or Third party provider	Enterprise or Third party provider	On premise Off premise	Trusted
Hybrid	Both Enterprise & third party provider	Both Enterprise & third party provider	Both on premise & off premise	Trusted & untrusted

Table 7 – Infrastructure Roles (based on CSA[3])

More detailed understanding of roles with regard to Infrastructure provision might be important here too. As table 7 illustrates, who actually owns or operates the infrastructure, or where it is located can vary within each Cloud Deployment Type. Ultimately it is the level of trust that determines what is actually public or private.

Trust in this case is the context of the organization producing the Service Architecture. That is, a capability is classified as public and untrusted by them because there are other users of the capability who are unknown and hence untrusted by the organization. However, they will in a different context be trusted by the Public Service provider, to the extent at least that the user has registered their account with them and pays for the service.

Improving Deployment Agility

The challenge with modeling deployment is that deployment diagrams are more typically used to define relatively static scenarios. However, a key goal with SOA of course is to provide a more dynamic, loosely coupled environment that is more responsive to change.

This implies that a static representation of where resources are deployed is inappropriate. You could get round this by labeling a node as :Cloud and showing all the relevant artifacts deployed to that without having to specify exactly where they are located, similar to that in Figures 9 and 10.

Alternatively, you could explicitly model a node acting as a Service Broker through which all Service Requests are dispatched to their correct endpoints.

Figure 11 depicts a Service Broker that can be driven 'intelligently' by policies and contracts, and by notification of changes in actual endpoints and deployments.

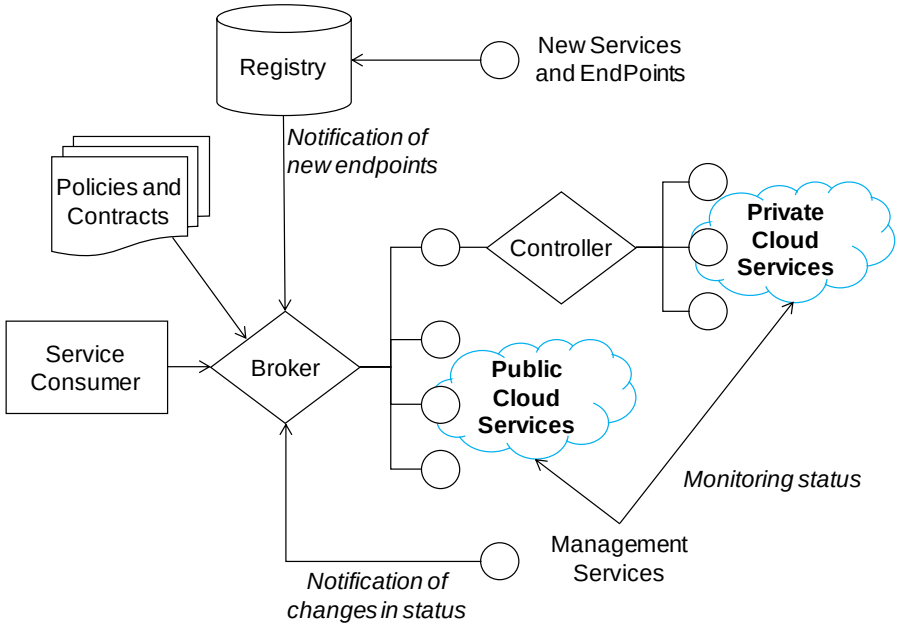


Figure 11 – Service Broker Pattern

Hence if the Service Manager or controller redeploys the endpoint and the deployment artifacts to a new node, the broker can be notified of that change and route requests accordingly without impact on the Service Consumer. This may be for reasons of scalability, load balancing, backup and failover, or new instances of Services that have been registered that can be used because they conform to policies and contracts, not because the Service Consumer need have explicit knowledge of their existence.

Of course this implies that the Services provided, and the assemblies that consume them have been designed with SOA principles such as loose coupling very much in mind.

The Service Broker doesn't have to be a centralized node. Federated brokers may resolve endpoints in many nodes on the network, including at the Service Consumer's node, before dispatch.

Nor does the broker capability have to be that sophisticated, it may be just a straightforward configuration file that is referenced at runtime in the operating environment, but that can at the same time be poked with new information.

That said, the capabilities provided by many Enterprise Service Bus (ESB) products may be ideal in this respect and provide more sophisticated capabilities that would suit the large enterprise.

Summary

In this report we have shown that CBDI-SAE readily accommodates the use of Cloud Services and Cloud Deployment. We have shown that some extensions to the SAE approach are necessary to establish visibility and governance over Cloud related architectural decisions. This is particularly important in areas relating to security and integrity. Equally the basic SAE approach must be adjusted to extend classification

systems and to refine how the Views are defined in order to reflect earlier decisions on deployment in the Business and Specification Views.

The prime objective for architects is to ensure appropriate policies are set to govern the use of Cloud Services, to ensure deployment decisions reflect and are traceable to business requirements and that Cloud deployments maintain appropriate levels of security, integrity and agility whilst reducing total cost of ownership.

---

[1] http://www.cbdiforum.com/bronze/webservices_part1/sld009.php3
[2] National Institute of Standards and Technology (NIST)Cloud Computing http://csrc.nist.gov/groups/SNS/cloud-computing/index.html
[3] http://www.cloudsecurityalliance.org/
[4] Figure 1, SOA Operational Infrastructure Adoption Roadmap. CBDI Journal, June 2006
[5] http://aws.amazon.com/
[6] Underlying Services - Architecture existing and acquired resources, CBDI Journal August 2009
[7] http://aws.amazon.com/rds/
[8] http://aws.amazon.com/ec2/

About CBDI

CBDI Forum is the Everware-CBDI research capability and portal providing independent guidance on best practice in service oriented architecture and application modernization.

Working with F5000 enterprises and governments the CBDI Research Team is progressively developing structured methodology and reference architecture for all aspects of SOA.

CBDI Products

The CBDI Journal is freely available to registered members. Published quarterly, it provides in-depth treatment of key practice issues for all roles and disciplines involved in planning, architecting, managing and delivering business solutions.

Visit www.cbdiforum.com to register.

Platinum subscription – A CBDI Forum subscription provides an enterprise or individual with access to the CBDI-SAE Knowledgebase for SOA delivering ongoing practice research, guidance materials, eLearning, tools, templates and other resources.

Everware-CBDI Services

At Everware-CBDI we enable large enterprises and governments to become more agile by modernizing their business systems. We have repeatable processes, resources, tools and knowledge-based products that enable enterprises to transform their current applications in an efficient, low risk manner, into an optimized service-based solutions portfolio that supports continuous, rapid and low cost evolution. Our consulting services range from providing practices and independent governance to architecture development, solution delivery and service engineering.

Contact

To find out more, and to discuss your requirements visit www.everware-cbdi.com or call

USA and Americas: 703-246-0000 or 888-383-7927 (USA)

Europe, Middle East, Africa, Asia, and Australasia: Telephone: +353 (0)28 38073 (Ireland)

www.everware-cbdi.com