Oracle® Reference Architecture

Cloud Foundation Architecture

Release 3.0

E24529-01

November 2011

ORACLE®

Cloud Foundation Architecture, Release 3.0

E24529-01

Primary Author: Mark Wilkins

Contributing Author: Stephen Bennett, James Baty

Contributor: Anbu Krishnaswamy Anbarasu, Mark Carlson, Margret Lee

Contents

4 Cloud Conceptual Architecture

List of Figures

Send Us Your Comments

Cloud Foundation Architecture, Release 3.0

E24529-01

Oracle welcomes your comments and suggestions on the quality and usefulness of this publication. Your input is an important part of the information used for revision.

- Did you find any errors?
- Is the information clearly presented?
- Do you need more information? If so, where?
- Are the examples correct? Do you need more examples?
- What features did you like most about this document?

If you find any errors or have any other suggestions for improvement, please indicate the title and part number of the documentation and the chapter, section, and page number (if available). You can send comments to us at its_feedback_ww@oracle.com.

Preface

The Oracle Reference Architecture (ORA) is a product-agnostic blueprint for a modern IT foundation. By identifying common business needs and mapping IT capabilities, principles, standards, and best practices, ORA describes an environment for consistency, interoperability, and longevity, minimizing the risk of product incompatibilities and obsolescence.

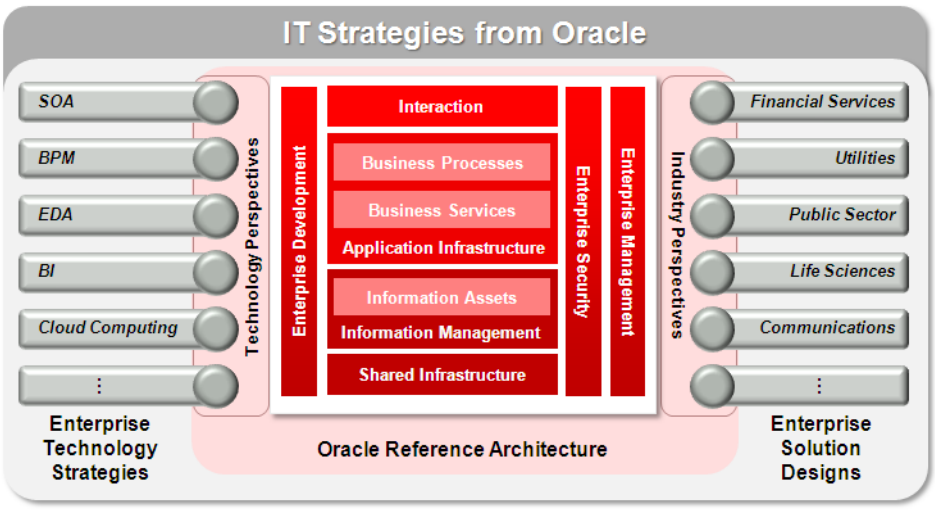
The ORA Cloud documents present the ORA concepts from the perspective of Cloud, highlighting the specific details of Cloud as an elaboration of the ORA core concepts. This ORA Cloud perspective comprises two documents:

- Cloud Foundation: primarily a conceptual architecture for Cloud, specifying architectural characteristics and expectations of Cloud at a business and operational level. Also included in this document are architectural principles, standards, concepts, a conceptual view for Cloud architecture, and its relationship to ORA.
- Cloud Infrastructure: relates the Cloud characteristics and requirements, as defined by the conceptual architecture, to the Oracle infrastructure and provides a number of architecture views to help architects and developers who are focusing on Cloud.

This document is part of a series of documents that describe the Oracle technology strategy for Cloud. This "Cloud Foundation" document provides the underlying architectural definitions for Cloud concepts. The Cloud Foundation document presents important basic concepts of Cloud that are fundamental to building applications and preparing the IT and the broader enterprise for Cloud adoption.

Introduction to IT Strategies from Oracle (ITSO)

IT Strategies from Oracle (ITSO) is a series of documentation and supporting material designed to enable organizations to develop an architecture-centric approach to enterprise-class IT initiatives. ITSO presents successful technology strategies and solution designs by defining universally adopted architecture concepts, principles, guidelines, standards, and patterns.



ITSO is made up of three primary elements:

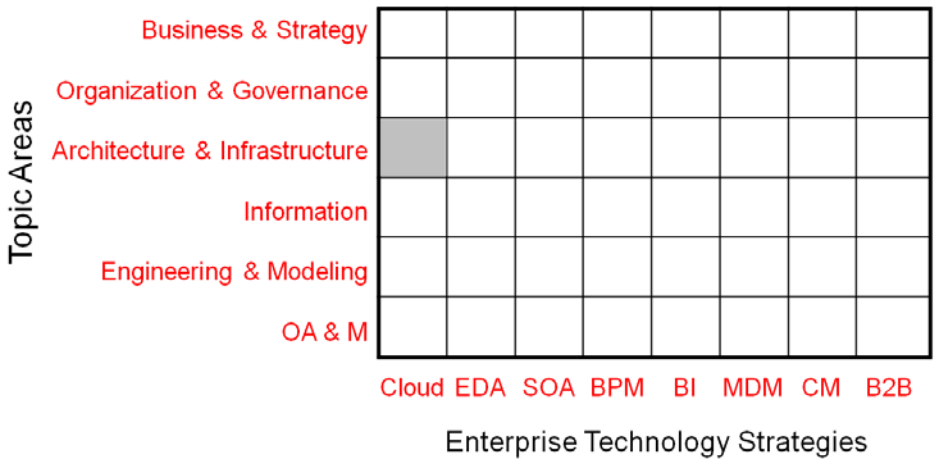
- **Oracle Reference Architecture (ORA)** defines a detailed and consistent architecture for developing and integrating solutions based on Oracle technologies. The reference architecture offers architecture principles and guidance based on recommendations from technical experts across Oracle. It covers a broad spectrum of concerns pertaining to technology architecture, including middleware, database, hardware, processes, and services.
- **Enterprise Technology Strategies (ETS)** offer valuable guidance on the adoption of horizontal technologies for the enterprise. They explain how to successfully execute on a strategy by addressing concerns pertaining to architecture, technology, engineering, strategy, and governance. An organization can use this material to measure their maturity, develop their strategy, and achieve greater levels of adoption and success. In addition, each ETS extends the Oracle Reference Architecture by adding the unique capabilities and components provided by that particular technology. It offers a horizontal technology-based perspective of ORA.
- **Enterprise Solution Designs (ESD)** are industry specific solution perspectives based on ORA. They define the high level business processes and functions, and the software capabilities in an underlying technology infrastructure that are required to build enterprise-wide industry solutions. ESDs also map the relevant application and technology products against solutions to illustrate how capabilities in Oracle's complete integrated stack can best meet the business, technical, and quality of service requirements within a particular industry.

This document is part of a series of documents that comprise the Cloud Enterprise Technology Strategy, which is included in the IT Strategies from Oracle collection.

Please consult the ITSO web site for a complete listing of Cloud and ORA documents as well as other materials in the ITSO series.

Document Purpose

ORA documents are mapped by technology concern to areas of interest in order to represent document purpose. This mapping for the Cloud Foundation Architecture is shown in the diagram below:



The goal of this foundation architecture is to define key entities (resource pools, control plane, deployable entities, etc.), functionality (service provisioning, user registration, etc.), design principles, (e.g., separation of resource and Cloud management), highlight relevant standards (Oracle Cloud API / DMTF use cases) and technologies.

Audience

This document is intended for enterprise architects, application architects, project managers and developers. The material is designed for a technical audience interested in learning about the Cloud Architecture and how to prepare to develop an enterprise class Cloud infrastructure.

Document Structure

This document is organized in chapters spanning introduction, background, conceptual architecture, standards and technologies, ORA interlock, and appendices. The main document chapters are as follows:

Chapter 1 provides brief introduction to Cloud and description of the scope and purpose of this document.

Chapter 2 outlines of the benefits, expectations, and challenges of the Cloud architecture.

Chapter 3 defines the terminology and concepts that are specific to Cloud architecture.

Chapter 4 presents the conceptual architecture for Cloud.

Chapter 5 describes the relationships between Cloud and other aspects of ORA.

Chapter 6 is a summary of the document.

Appendix A provide lists of relevant supplementary reading and references.

Glossary contains a list of terms used throughout the document.

How to Use This Document

This document should be read as a prerequisite to the ORA Cloud Infrastructure document.

Conventions

The following typeface conventions are used in this document:

Convention	Meaning
boldface text	Boldface type in text indicates a term defined in the text, the *ORA Master Glossary*, or in both locations.
italic text	Italics type in text indicates the name of a document or external reference.
underline text	Underline text indicates a hypertext link.

"Cloud" v. "Cloud computing" - The two terms, Cloud and Cloud computing are synonymous and are used interchangeably throughout this document.

The term ***"Cloud services"*** refers to the services offered by Cloud computing (to Cloud Consumers). Since the word "service" is used to refer to many different things, in order to avoid confusion, it will be preceded by the name of the domain to which it belongs throughout this document. This convention differs from the one applied by the SOA ETS documents.

1
Introduction

Cloud computing is a major step forward in the evolution of IT towards commoditization of hardware and other computing resources. Just as SOA has revolutionized the creation of business applications by abstracting technical implementation details to present robust business services for rapid composition, so Cloud abstracts computing resources (infrastructure, platform, or applications) for rapid assembly and deployment.

The opportunities Cloud computing offers, include substantial improvements in IT costs and agility (a variety of other benefits are described in Chapter 2). These benefits do not come without challenges and the need for change however, not only in the underlying IT architecture, but also in the development and operations strategies, the supporting organizations, and IT culture. This document focuses on the highest level architectural considerations as part of a broader, strategic assessment and roadmap for implementing Cloud.

Cloud computing, or any IT change, is not just about buying some new technology. A firewall appliance doesn't create security by itself, and just adding virtualization to your servers doesn't make a Cloud. This doesn't mean that Cloud computing is about more complex IT. In fact it can and should be about simplification in building and using infrastructure. This does however, require consideration in design and likely 'refactoring' of IT responsibilities. Some of the application architecture that is normally done up-front, shifts to 'run-time' architecture, activated by the Cloud consumers; while some of the traditional run-time operations, normally performed by system administrators, moves to a design-time architecture subsequently enabling self-service. In particular, Cloud computing is about selecting the most appropriate levels of abstraction, APIs, and standards that enable longevity in the selected IT investment. That's why architecture is important when considering Cloud.

A robust approach to Cloud must include the following:

- *Business Justification*. A perspective on the importance of specifying measurable transformational goals for the implementation of a Cloud computing.
- *Cloud Architecture*. A high level architectural framework highlighting not only the key components of a Cloud architecture, but also confronting some of the issues in implementing key logical abstractions.
- *Roadmap*. Having selected and developed a high level Cloud architecture, it is important to assess the existing environment, including the organization's maturity on several important dimensions (e.g., operations, governance, etc.) and create a roadmap that guides the implemention of that strategy.

The focus of this document is the starting point for Cloud Architecture: that is, specifically the Conceptual Architecture for Cloud, laying the foundation for the more

tangible architectural definitions that will be found in the companion ORA perspective, ***Cloud Infrastructure Architecture***.

While the business benefits and expectations are briefly covered here, in order to support the identification of architectural capabilities needed for a conceptual model, Roadmap Planning, and associated guidance outlined in the list above will be the subject of other ITSO Cloud Enterprise Technology Strategy documents.

1.1 What's different about Cloud Computing?

In a nutshell Cloud is a style of computing in which dynamically scalable and rapidly deployable IT resources are provided as a service over a network (typically the internet). Cloud computing is also commonly defined as the ability to provide (and consume) computing resources and other IT capabilities, on a subscription basis, over a network. The latter definition, however, appears to be nothing more than “hosting”. One of the key differentiators of cloud over “hosting” is the ability to expand and contract resources on-demand (i.e. “dynamically scalable”). This characteristic is also referred to as “elasticity” and is a key capability of Cloud computing that traditional “hosting” does not provide. Furthermore, cloud offerings are significantly broader than infrastructure and platform hosting and currently include such service as storage, information, security, testing, applications, all the way to full scale “virtual datacenter”.

How does Cloud differ from outsourcing and hosting? In many ways outsourcing and hosting have paved the way for Cloud, but Cloud is a major step forward on this evolutionary path. Traditional outsourcing provided IT services (applications, data storage, etc.) while abstracting away the technical details (uptime, system performance, disaster recovery, etc.) through legal contracts. More recently, hosting has provided more technical capabilities, aided by standardization (**LAMP** for example), to enable applications to be developed and hosted within discrete units of rented compute resources (servers, disk, network capacity, etc.). Cloud takes these basic concepts to another level by providing virtualized resources (computing infrastructure, platforms, and applications), various forms of self-service, and abstracted management and control.

Another difference with Cloud is the trend toward the creation and use of more well-defined services (which have their own interfaces, quality of service characteristics, etc.). The trend toward more of the service catalog approach helps IT organizations to better communicate the services they offer, the quality of service levels supported, and the costs associated with each.

1.2 Origins of Cloud Computing

In 1960s John McCarthy said: “computation may someday be organized as a public utility.”

The moniker of 'Cloud' computing came into use in 2006 when Eric Schmidt referred to Google's data centers as 'Cloud Servers' and Amazon launched Amazon Web Services. It is now accepted that this describes the next phase in the evolution of IT from mainframe, through client server, and n-tier, to an 'on-demand' infrastructure. The National Institute of Standards and Technology, Information Technology Laboratory (NIST) offers the following generally accepted definition:

“Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.”

NIST goes on to say “this Cloud model promotes availability and is composed of five essential characteristics, three service models, and four deployment models”, however, these characteristics and models do not apply equally to all 'clouds', for example, a self-service developer public Cloud likely has a completely different implementation from a government agency 'community' Cloud. So if we consider 'enterprise' Cloud computing, where does it stand in relation to the commonly accepted Cloud definitions? This is a core question that will be addressed by this document.

1.3 Scope of this document

The scope of this document is limited primarily to Cloud models suitable for the enterprise: this includes Private and Hybrid Cloud spanning Infrastructure, Platform, and Software as a Service (IaaS, PaaS, and SaaS). Within this scope the central focus of this document is the Enterprise perspective for public and private Cloud offerings (i.e. excluding the Cloud consumer).

Specialized, large scale computing needs (e.g. Scientific) are not addressed by this document.

1.4 Assumptions

The architecture for Cloud is still developing and there is currently little consensus in the industry on its architectural definitions, capabilities, and features. As such, the expectations for Cloud may change and the corresponding standards and technologies will evolve. This document attempts to provide a technology agnostic starting point for the Cloud architecture by exploring the architectural characteristics necessary to meet the currently understood needs from Cloud.

2

Benefits, Expectations, and Challenges

Cloud computing is generally viewed as an IT strategy that offers significant benefits in cost reduction and business flexibility. In 2008, IDC provided a seminal analysis of enterprise desires for Cloud computing in which 244 IT executives rated their perceived benefits from Cloud computing.

In the survey the number one benefit of sourcing IT from the Cloud is speed and ease of deployment. There is a strong belief that Cloud computing can address the slow pace of traditional IT development and deployment of business applications. The following three key benefits address improving the economics of businesses' use of IT:

- aligning costs with utilization
- reducing the need for in-house IT staff (and related costs)
- replacing large up-front financial outlays with streaming payments (or conversion from **CAPEX** to **OPEX**)

This last point highlights another important opportunity arising from the Cloud cost model which is that of removing the cost of up-front IT investment as a barrier to entry for new ideas and programs: whether in a small startup or large enterprise, a business plan for a technology intensive concept no longer needs to factor-in the high cost and time to deploy new hardware when they can simply create infrastructure, platform, or software services on demand and pay-as-they-go. An example of this effect is shown in more detail in Section 2.1.5, "Reduced Cost of Entry / Time to ROI."

An alternative to simple cost savings through reduction in staff, is to make an investment in retraining existing IT staff to maximize the benefits afforded by Cloud and thereby ultimately increase the productivity of IT. By cross-training IT staff into systems management or governance roles, for example, the focus of the cloud implementation moves beyond simple cost saving through consolidation to improvements in quality of service, efficiency, etc.

Another highly-rated anticipated benefit from Cloud services is the ability to keep business systems and services current in the market. Since Cloud services are based on a shared resources model, providing all users “instant” access to new functionality should be considerably easier for suppliers than through traditional models.

While the commonest motivation for migration to Cloud is cost saving, the number one perceived inhibitor is security. In reality it is likely that more significant, potentially unanticipated benefits will be realized, such as opportunities to offer core competencies as services for sale (and outsource non-core functions); while security in the Cloud should be more uniformly applied and ultimately superior to existing security across our industry.

In summary the expectations for Cloud include:

- reduced cost

- higher availability
- unconstrained growth of IT capabilities
- removal of barriers to technology intensive business initiatives
- increased speed to market
- access to more IT options

This list is not intended to be exhaustive, but it is a starting point towards establishing business requirements for a Cloud architecture.

2.1 Key Considerations

In the evaluation of business requirements for an enterprise seeking to develop a Cloud computing strategy it is important to consider the following five dimensions:

1. Efficiency vs. Agility - are the primary benefits sought in the nature of cost reductions, or shifts, or is there a desire to garner functional business improvements, say dramatic reduction in Time-To-Market (TTM) for new customer services.
2. Tactical vs. Strategic - are the benefits sought from Cloud computing tactical in nature, i.e., implementation of incremental Cloud functionality across the organization (e.g., virtualize everywhere), vs. embarking on a more transformational shift in IT strategy (create a new self service development environment and model).
3. Classic client-server vs. Web2.0/Rich Internet Applications - The enterprise needs to understand its application development model and choose between traditional enterprise computing or next generation web centric applications. Archetypal public Cloud services are not likely models for complicated enterprise application architectures, however, the enterprise may develop resource pools in a Public Cloud specifically to support more traditional application models or accommodate them in a Private Cloud.
4. How does this align with larger business strategy? For example, is the enterprise itself seeking increased centralization of business control or is it pursuing a decentralized resource model.
5. Are there key functional IT use cases under consideration? E.g., development and test, peak load management, resource consolidation.

Clearly, there are potential interdependencies between these dimensions, but there are also two extreme cases: (1) where the Cloud computing strategy is a major strategic project to adopt a new application development paradigm and thus establish a competitive differentiation in service delivery and (2) a tactical approach to reduce costs across the enterprise for existing applications and infrastructure. However, on the other hand, a plan to change the application development model could be to achieve a goal of cost saving. It is also possible to have a strategy that combines both tactical and strategic goals separately in multiple projects or initiatives. In any case, it is very important to define specific business goals both for the design of the infrastructure and to enable effective monitoring and assessment of the benefits.

The following sections consider each of these five business dimensions in more detail.

2.1.1 Efficiency vs. Agility

A dominant motivation for enterprises to adopt Cloud computing is to reduce IT costs. One major target of these IT costs is Hardware Utilization Efficiency (HUE - simply

the percentage of CPU, disk, memory, etc. resource utilization of a server). The general level of hardware utilization in data centers today is around 7-15%. Much of this is due to the compartmentalization of workloads on dedicated servers. Cloud computing architectures coupling application payload mobility with virtualization of pooled resources offer great promise to address this. However like traditional server consolidation efforts the measurement of enterprise ROI for Cloud computing is based on cost reduction. Such computations of cost reduction also include infrastructure power efficiency (DCIE - Data center infrastructure efficiency and PUE - Power utilization efficiency, ratios between IT equipment power and total facility power, proposed by Green Grid, etc.), software licensing, administrative costs, etc. The final computation will be particular to a given enterprise, but generally using Cloud computing to reduce costs is tangible and measurable.

Beyond simple power usage, there are many other direct cost components of overall Total Cost of Ownership (TCO). These include hardware and software expenses, operations and service desk labor, finance & administration costs, etc. While Cloud computing may easily reduce some of these capital and operating costs, it may add complexities to software licensing costs. This likely requires some mechanisms to monitor and record concurrent license usage.

In addition to these direct costs there are many indirect costs that may be addressed in a Cloud computing initiative. These indirect costs involve non-budgeted measurements of the efficiency of the IT group to deliver expected services to end users. If the IT management and solutions are inefficient, end users may spend significant time in self and peer support. These costs may be reduced in a Cloud computing environment, but if not planned for they could also increase as IT is moved to a self service model.

Beyond basic cost reduction Cloud computing also promises a range of benefits in improving overall business agility by enabling significant improvements in TTM of application development and service deployment. One of the classic Cloud computing use cases is to utilize pooled Cloud resources for development and test. However, fully exploiting Cloud computing for business agility might require many businesses to shift from traditional application architecture a more service-centric internet application model. This of course implies more complex ROI considerations and likely implies a strategic enterprise sponsorship outside a pure internal IT initiative.

2.1.2 Tactical vs. Strategic

As explained later in the discussion around applying an IT maturity model to Cloud computing initiatives, Cloud computing implies significant advances in developing and automating operational procedures. The extreme case promise is to move primarily to a self-service IT model. The adoption of a roadmap to Cloud computing is taken in context with the enterprise goals and its IT maturity. Goals that may require less IT maturity can be easily implemented in a tactical manner widely across the enterprise (e.g., the common 'virtualize everywhere' strategy). However, Cloud computing goals that require high levels of IT maturity are likely to need a strategic transformation project that is focussed on a narrowly defined business goal.

Either direction, tactical or strategic, pursued exclusively, will be likely to eventually result in pressure to adopt the opposite direction. For example, strategic transformation projects eventually merit more widespread adoption. Also, as in the area of efficiency and agility, where most enterprises are likely to pursue combined goals, an enterprise Cloud computing strategy may include both tactical wide diffusion efforts and targeted strategic projects.

2.1.3 Target Application Development Model

The Cloud computing model pioneered by Amazon Web Services (AWS) has been especially useful for the development of Rich Internet Applications (RIA). RIAs are Web applications which typically use a light client application running inside a networked end-user device to deliver the same features and functions normally associated with desktop applications. From the data center perspective these applications are built on stateless web services that can be easily load balanced and migrated across a dynamic resource pool. This contrasts with traditional enterprise applications that are both more complex and more difficult to dynamically provision. On the other hand, most enterprises are independently migrating to a more stateless, web-centric application architecture. The appropriateness of Cloud computing strategy for a given enterprise depends in great measure on whether the planned application development model is one based on Internet delivery or one that will impose design limitations such as, application statefullness and execution portability. Some of this challenge may be solved by additional service abstraction of existing application functionality, however, bottom line, moving to a Cloud computing model for most enterprises implies likely further evolution of existing web services and internet application development models.

There is no substitute however, for the orchestration of complex business functions and the maintenance of business entities commonly found in today's business processes and supported by traditional applications such as CRM, ERP, etc. While it is clear that the simple, highly scalable functions, such as search, are an easy fit for Cloud computing it is important to be able to apply the Cloud model to the broader enterprise needs. This can be achieved most effectively by decomposing the applications, in a SOA fashion, into smaller, portable, autonomous business functions while abstracting the business process, thereby facilitating Cloud migration.

2.1.4 Business Strategy Alignment

At a macro level, the Cloud computing strategy should also complement larger business strategy. For example, many industries today are dominated by overall business consolidation and associated mergers and acquisition growth strategies. In this case the general IT strategy and related Cloud strategy can likely play a significant part in supporting consolidation of existing IT infrastructure and the rapid integration of the IT applications accompanying acquisitions. On the other hand, there are industries where business growth is driven by expansion into new, perhaps overseas, markets. In these cases the Cloud strategy might be designed to support micro-scaling, or the sipping off of virtual business infrastructure to support independent operations into the new markets.

In other cases the high degree of competition in evolving industries may lead to competitive differentiation of services as a key strategy and the complementary Cloud strategy could support the rapid development and scaling of these services. In several areas the Cloud strategy may not simply support a given business strategy but be a significant enabler of that strategy. In any case the contribution of the Cloud strategy to overall business strategy should be considered.

2.1.5 Reduced Cost of Entry / Time to ROI

One of the key business drivers for new projects (whether within an existing enterprise or for an entirely new business) is the near total removal of computing costs as a barrier to entry. The "pay-as-you-go", "use-what-you-need" nature of Cloud is likely to increase the speed of innovation by deferring the cost of computing infrastructure while all but eliminating startup costs.

Traditionally, any new computing project required substantial up-front costs in hardware, setup, administration, and lead-time. The result of this fixed cost ensures a substantial lag before any measure of return on investment can be realized. Figure 2–1 shows a Cost-Volume-Profit analysis[1] in which a new project fails to reach its goal before profitability due to high fixed costs implied by this up-front capital investment.

Figure 2–1 Example traditional cost summary

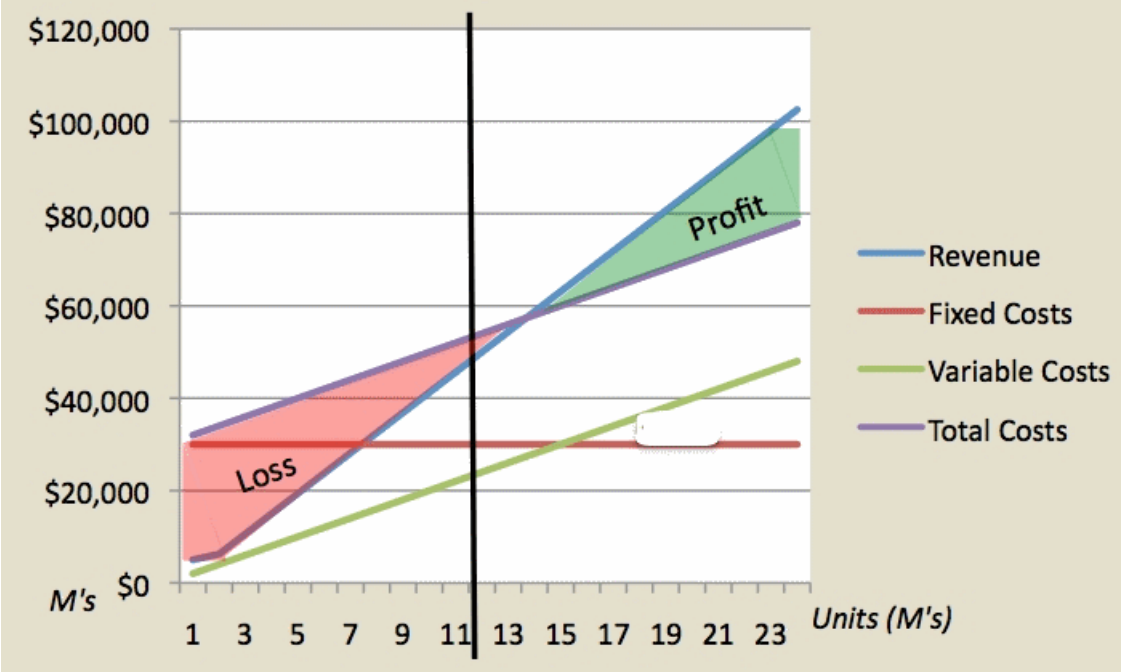
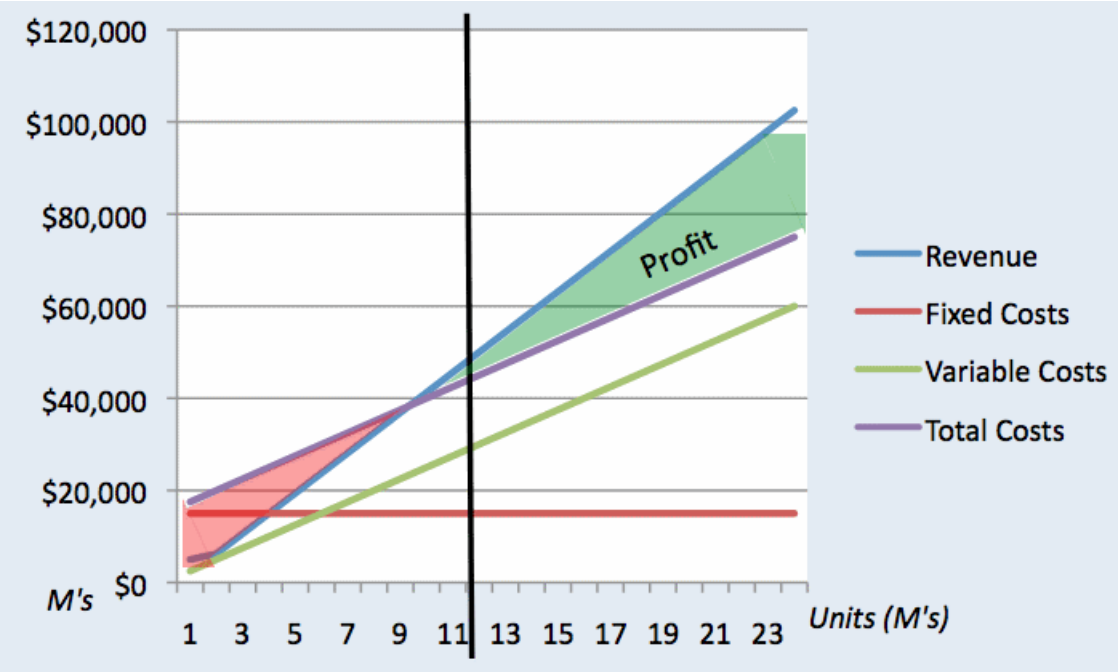


Figure 2–2 shows the corresponding cost analysis wherein the same projects succeeds in reaching its goal due to lower fixed costs even in spite of (25%) higher variable costs. This is expected in a Cloud environment in which resources are consumed (and paid for) in much smaller increments, as needed, leading to later accumulation of cost relative to revenue.

[1] Details of Cost-Volume-Profit analysis can be found at http://en.wikipedia.org/wiki/Cost-Volume-Profit_Analysis

Figure 2–2 Cloud costs for equivalent resources



This reduced cost of entry saving works best in a Public Cloud where the investment in Cloud infrastructure, facilities, management, etc. have been financed separately. In the case of Private Cloud, however, the initial cost of establishing the Cloud will most likely need to be justified (and absorbed) by a substantial application migration rather than a time-to-ROI model such as this.

2.2 Challenges to be addressed by Cloud architecture

Like SOA, Cloud is a new term for the evolution of an existing concept. “Outsourcing” and “hosting” are well established techniques for alleviating the complexities of system ownership with promises of reduced cost, more effective operations, etc. Also like SOA, Cloud is not a panacea and Cloud will not be without its own challenges.

As the title suggests the focus of this section is concerned with issues that must be addressed by the Cloud Architecture. It is worth noting that architecture cannot address all matters arising in the development of a new technology. For example, one of the major challenges facing Cloud providers is the discomfort caused by no longer owning tangible hardware assets. This is not necessarily true in the case of Enterprise Cloud, but even there the departments that used to own their own hardware will no longer have access to the centralized datacenter and those assets will likely become less and less tangible over time. This “emotional attachment” and other similar concerns relating to organization, governance, project management, etc. will not be addressed in this document.

Similarly, another risk arising from the ease of access and reduced cost of entry for new developments will be the temptation for business units to circumvent their corporate IT strategy by going to public Cloud services for their IT needs. This will lead to fragmentation of IT as it did with the advent of the mini-computer, which enabled a wave of departmental computing to break free from the constraints of

mainframe computing. This however, is a governance concern which is beyond the scope of this document.

The following list of challenges is intended to be, for the most part, unique to Cloud. This list is intended to highlight concerns that must be addressed by the Cloud Architecture, although solutions, beyond basic suggestions, are not explored in this section. It is important to point out also that this section is not intended to be taken as a list of reasons not to follow a Cloud strategy!

2.2.1 Delegation of control / lack of ownership / abstraction

A number of aspects of this broad set of concerns have already been identified in earlier sections of this document, but the purpose of this section is to consider the purely architectural perspective of these challenges.

This challenge is really about the effects of abstraction. Abstraction, of course, brings many benefits: it simplifies and streamlines the use of an object (IT resources in the case of Cloud), encourages standardization and modularity; however, it also "hides" the underlying details that should not be necessary to the use of the object or service. Therein lies the problem: the determination of which details are unnecessary from those that may be required at various stages throughout the lifecycle of the object or service. Furthermore, is it often difficult to expose certain types of information through the mechanisms available for the abstraction.

2.2.2 Application portability

As the logical models of the *ORA Cloud Infrastructure* document will show, Cloud computing is provided most effectively when certain assumptions are applied as logical architectural constraints: fundamentally, Cloud implementation is substantially simplified when it follows architectural strategies based on many small units of computer resources (CPU and memory) connected by a network that can be allocated on-demand. In the absence of other compute models (e.g. large-scale **SMP**, **NUMA**, etc.), applications, databases, and other platform software for that matter, which are not designed to scale in this environment, are not suitable for direct migration Cloud.

Applications (and other software) that expand their number of supported uses, transaction rate, or other measures of processing capacity by increasing tightly coupled computer resources e.g. multi-core chips and/or SMP motherboard, processor RAM or other directly shared memory (i.e. processors and memory connected by an internal bus) are said to scale vertically. In cases of large IO requirements some applications (primarily database) also require direct control of disk organization (e.g. database table and index partitioning, striping for performance, etc.), although this requirement is less common with the advent of SAN's on high capacity networks and caching techniques.

This fundamental constraint of application architectures requiring vertical scalability arises because of the need for specific coupling requirements between the application and infrastructure components which would be particularly difficult to provide on-demand. Vertically scaled applications require low-level hardware configurations that are particularly difficult to provide via software control. In a nutshell, applications that place specific requirements on the underlying platform architecture and design are not ideally suited for a Cloud environment.

This is certainly not a hard-and-fast rule however, with many variations enabling vertically scaled applications into the Cloud realm, such as:

- Software architectures designed with multi-tenancy (see Section 3.2.1 for definition) enable many distinct users to "share" a single application on a

vertically scaled platform (e.g. a traditional RDBMS on a very large SMP infrastructure)

- Consumers not requiring extreme scalability from the infrastructure (e.g. development environments using MySQL)

These workarounds however, will lack the elasticity and potentially other characteristics expected of Cloud unless highly sophisticated (and most likely proprietary) infrastructure management capabilities are designed to mitigate these deficiencies. Again, it is necessary to caveat this by pointing out that not all Cloud characteristics are necessary in every case (nor are their weights equal), so it may be prudent to abandon elasticity in favor of vertical applications.

For all the reasons discussed so far, an archetypal Cloud (and its closely linked technology, virtualization) more readily provides horizontal scalability. Horizontally scaling architectures enable platform and application growth by adding compute capacity in small fixed increments. A physical manifestation of this might involve a large farm of 2-CPU processors with a few Gbs of RAM with a similarly fixed network and disk IO bandwidth; a virtualized environment on the other hand might present a similar growth path to its consumer, while it is made up of more diverse infrastructure behind the scenes.

Considering the nature of many complex enterprise applications and the diversity of today's IT, a Cloud infrastructure supporting both horizontal and vertical application scalability (perhaps in conjunction with other compromises, such as application-level multi-tenancy) could provide many Cloud benefits while maintaining high performance and simplifying migration.

2.2.3 Security

Unfortunately, security is an emotive topic and there is, currently at least, an element of fear about putting the security of critical business information and operations into the realm of something as abstract as Cloud, whether it is a public or private offering.

Fundamentally, it is well accepted that "military levels of security" are achievable, but costs (and sometimes inexperience) drive compromises for most businesses; however, with the economies of scale offered by Cloud, combined with increased motivation of having more at stake, Cloud computing should lead to security worthy of Fort Knox.

2.2.4 Proliferation and Version control

Will decommissioning ever happen? Just as Parkinson's Law ("Work expands so as to fill the time available for its completion") has been applied to data, software, etc. (e.g. "software expands to fill available memory"), given the ease with which services are commissioned within a Cloud, it seems likely that server instances will be commissioned to fill available resources (in physics "nature abhors a vacuum"). The remedy to this all-too-common effect is discipline which is fundamentally a governance concern; however, in this case governance requires architectural support in the form of metering, even in a private Cloud. Without an effective means of accounting for resource utilization users will have no incentive to shutdown and decommission instances, release storage resources, or expire unused images from the library.

While accounting commonly results in billing in a public consumption model, there must be an equivalent in a private Cloud situation. This can take various forms, but the allocation of cost for resource utilization is a basic necessity to avoid the Parkinson effect.

Other opportunities to impose the necessary discipline to reduce sprawl include mandating an expiry for services in the Cloud. An extension, or variation, of this theme would be to limit the number of versions retained.

2.2.5 Geographic Location

In principle a virtualized service, server, or datacenter can be anywhere and given adequate network bandwidth they could span multiple physical locations at once.

In purely technical terms the location of the hardware is not important; however, many countries have a wide variety of laws mandating that data must not cross national (and sometimes state or other regional) boundaries.

In practice, network bandwidth between cooperative computing centers is likely to be an important architectural constraint on the assembly of resources, for example, between a database service and an application.

2.2.6 Transparency

How much do Cloud consumers need to know? If you are simply a consumer of an application in a SaaS model all you need to know is manageable within a SLA, that is, the number concurrent users it will support, the uptime and disaster recovery contingencies, etc. In this case the architecture of the system, the number of servers running, the processor architecture, etc. are not important to the SaaS consumer. However, even the SaaS model is rarely that cut-and-dried and when we consider PaaS and IaaS these details, and many more like them, become highly significant.

There are other dimensions to the question of what consumers need to know: for example, what about SOX and other national regulations? Should the Cloud provider be required to share information about other compliance issues, security violations, outages, etc?

Another concern for transparency overlaps with the question of geographic location - most Cloud consumers are going to need to know where their data resides and this will require the Cloud provider to constrain data movement on the one hand and expose data location information n the other according to specific requriements.

3

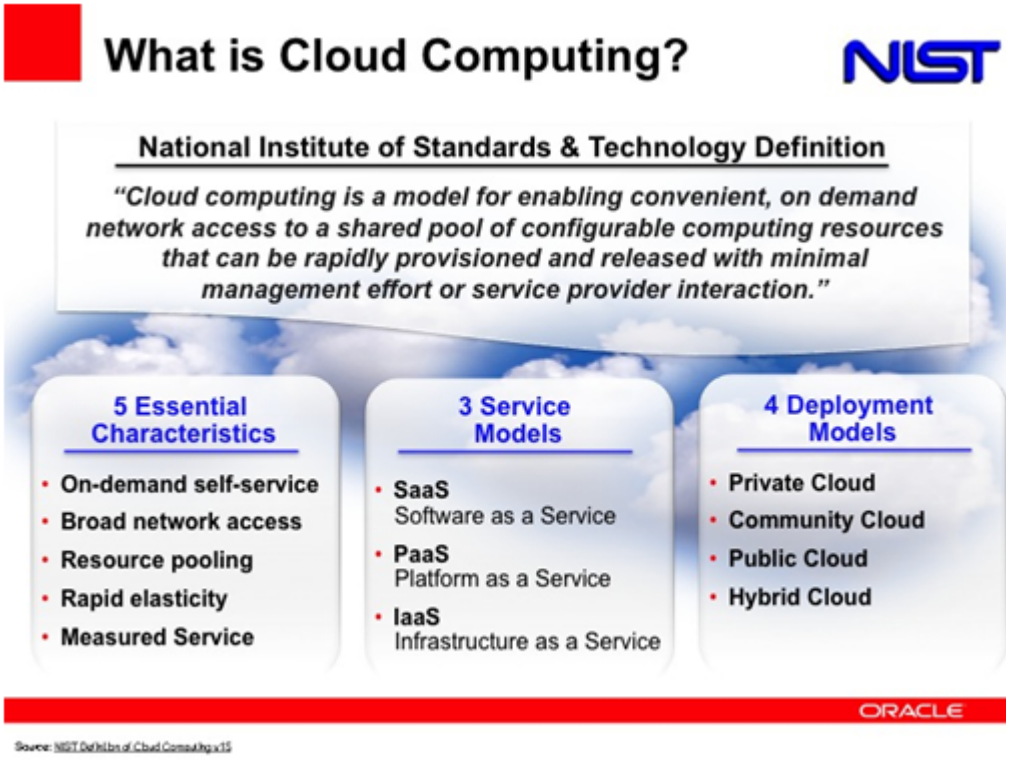
Terms and Concepts

This section introduces terms and concepts, Cloud characteristics, service and deployment models.

Based on today's widely accepted NIST definitions, Cloud computing is composed of five essential characteristics, three service models, and four deployment models. This document will describe these in some detail and explore addition potential considerations in order to develop a comprehensive set of architectural capabilities as the first step towards the definition of a conceptual architecture for Cloud.

The five essential characteristics, three service models, and four deployment models are summarized in Figure 3–1, "NIST Cloud Computing Definition".

Figure 3–1 NIST Cloud Computing Definition



Source: NIST Definition of Cloud Computing v15

The following sections define these characteristics and models in more detail.

3.1 Essential Characteristics

Not all clouds will implement the five essential characteristics of the NIST model in the same way or amount. For example, while 'rapid elasticity' for a public developer Cloud means instant end-user provisioning of a server, an enterprise Cloud might likely constrain what could be provisioned into a production Cloud, require operator validation of resource requests and have the compliance responsibility monitoring the respective payload. In such an enterprise environment 'on-demand self-service' might mean reducing the time to provision corporate IT months, two weeks or a few days - this still offers the potential of dramatic improvement.

3.1.1 On-demand self-service

A consumer can unilaterally provision computing capabilities, such as server time and network storage, as needed automatically without requiring human interaction with each service's provider.

Self service is a common aspiration for Cloud environments, but in an enterprise context this may still imply some level of operational control by IT, vs. a pure self-service model of consumer/public clouds.

In classic IT the request to allocate system resources or install application components typically involves the requesting user to interact with support personnel via complicated work-flow involving job tickets and likely in person phone calls, etc. Cloud computing seeks to automate these common provisioning processes to make it possible for the end user to directly provision resources seemingly without human intervention. This requires the obvious pre-provisioning of the target resources and the development and implementation of deployment models and payload infrastructure (e.g., service catalogs, configuration databases).

3.1.2 Resource pooling

The provider's computing resources are pooled to serve multiple consumers using a multi-tenant model, with different physical and virtual resources dynamically assigned and reassigned according to consumer demand. There is a sense of location independence in that the customer generally has no control or knowledge over the exact location of the provided resources but may be able to specify location at a higher level of abstraction (e.g., country, state, or datacenter). Examples of resources include storage, processing, memory, network bandwidth, and virtual machines.

Private clouds may also include other 'clouds', e.g., external public clouds, as one of the pooled resources addressed by an overall hybrid Cloud model. While hybrid clouds are often desired for low cost scaling, they represent special challenges for corporate information protection. These may be reserved for only certain use cases or classes of data.

3.1.3 Rapid elasticity

Capabilities can be rapidly and elastically provisioned, in some cases automatically, to quickly scale out and rapidly released to quickly scale in. To the consumer, the capabilities available for provisioning often appear to be unlimited and can be purchased in any quantity at any time.

As described under Elasticity above, the specific nature of 'rapid' may be accomplished via some level of corporate control, but provide dramatic improvements over the traditional IT procurement model.

3.1.4 Measured Service

Cloud systems automatically control and optimize resource use by leveraging a metering capability at some level of abstraction appropriate to the type of service (e.g., storage, processing, bandwidth, and active user accounts). Resource usage can be monitored, controlled, and reported providing transparency for both the provider and consumer of the utilized service.

In a private enterprise Cloud deployment there are likely significant requirements and opportunities to define new resource measurement and allocation models not previously implemented.

3.1.5 Broad network access

Capabilities are available over the network and accessed through standard mechanisms that promote use by heterogeneous thin or thick client platforms (e.g., mobile phones, laptops, and PDAs).

Enterprise / private clouds likely imply a higher degree of control over network access both to protect data in flight and to provide strong isolation between tenants (which may have different application or departmental information security requirements).

3.2 Additional Characteristics

The following lists some additional characteristics relevant to the development of a Cloud Architecture.

3.2.1 Multi-tenancy

Multi-tenancy is not called out as a separate essential characteristic of clouds in the NIST model but is specifically mentioned in the definition paragraph. While public clouds also must deal with the information protection issues of multi-tenancy, private enterprise clouds may need considerable reconsideration of their governance and compliance regimes to accommodate migration to a Cloud model, especially to be able to consider a hybrid model.

In classic IT environments systems infrastructure is purpose built for specific applications. Consolidation would typically combine applications with some affinity and common organizational ownership. Cloud computing implies diverse, heterogeneous multi-tenancy. This increases the operational risk security issues. The technology to accomplish this is not just OS virtualization. It can involve everything from lower levels OS partitioning to application server co-tenancy. Importantly Cloud multi-tenancy endeavors to make each tenant feel as though they are the sole user, with no visibility into other execution and user environments

3.2.2 Dev-Ops shift

Traditional IT implies not only the strong definition of separated roles (networking, operations, development, etc.) but typically the isolation of these roles in separate organizations. This exacerbates a classical challenge of trying to figure out how to scale a deployed application after it has been developed and deployed. The concept of dev-ops implies merging the roles and responsibilities of development and operations into combined teams that both develop applications and are responsible for moving them to production and on-going operations. The goal is radical improvements in Time-to-market, but this implies not just deploying new tools and technology, but shifting the organizations IT culture and organization.

3.2.3 Development Cycle

If the Dev-Ops model implies a refactoring or combination of previous roles, then a Cloud model may also imply a new separation of roles between building the infrastructure, and the independent building of services and applications. Historically an application would be deployed concurrent with its systems environment, at the time that it was built. The Cloud model isolates early resource pool build out, from later phase application service deployment. In its extreme this implies the delay of the instantiation of actual application 'architecture' to run-time. And similarly, operations, which use to be done reactively, would now be increasingly architected up-front into self-service automated procedures

3.2.4 Scale and Velocity

Much of the fundamental components of Cloud computing are classic IT principles, virtualization, consolidation, automation, etc. Perhaps the key difference is scale and velocity with which these technologies and architectural issues are applied. Simply put, a traditional IT environment may involve dozens or hundreds of systems, for which the configurations are changed a few times a year. By contrast, Cloud infrastructures, in combining multiple architectures into shared pools of large numbers of resources (servers, etc.) with almost constant change (daily or even hourly). This typically requires a fundamental shift in the way in which IT processes are implemented and carried out.

Of course enterprise Cloud computing may involve significantly different implementations of these Cloud attributes than public developer-centric Cloud services. Most enterprise Cloud environments will likely place restrictions on the type of code that may be deployed in the enterprise Cloud. Additional restrictions may be enforced due to logical and physical security concerns as well as software licensing. The level of on-demand elastic provisioning in an enterprise Cloud environment may require some level of organizational approval. Ultimately, enterprise Cloud computing will lead to dramatic improvements in service availability and significant reduction in costs, even while being more restrictive than the typical public Cloud environment.

3.3 Service Models

Cloud computing suggests everything previously provided by IT will be delivered as a service (“everything” as a Service is commonly referred to as “XaaS”). The concept of layered service types is not rigid, but the industry is generally dividing into three distinct environments and markets.

- Software as a Service (SaaS): 'Consumer' uses applications running on a Cloud infrastructure. The SaaS provider manages or controls the underlying software and infrastructure.
- Platform as a Service (PaaS): Consumers use programming languages and tools supported by the PaaS provider and then control the deployed application. The PaaS provider manages or controls the underlying Cloud platform, which includes everything below the run-time execution environment.
- Infrastructure as a Service (IaaS): Consumers deploy and run arbitrary software, and provisions processing, storage, networks, and other fundamental computing resources. The IaaS provider manages or controls the underlying physical Cloud infrastructure (i.e. everything below the operating system layer).

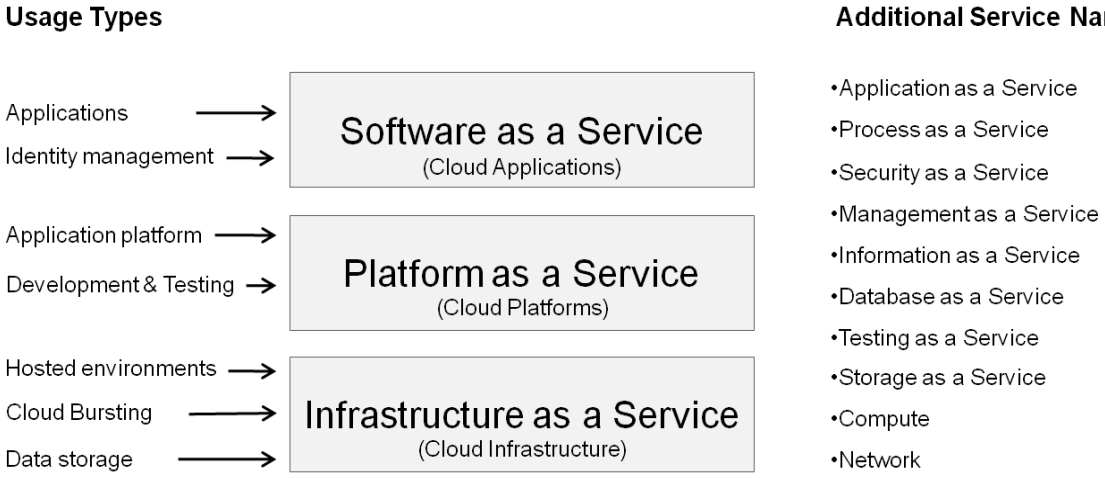
Other "X" as a Service models have been suggested, but commonly they are only specializations of the three described above.

The Storage Networking Industry Association (SNIA) defines a Data-storage as a Service (DaaS) that is distinct from and usable by any of IaaS, PaaS, and SaaS. This describes the current set of offerings of Cloud Storage that are typically independent of other services (although Amazon does link EC3 to S3).

Indeed, the concept of Data as a Service (DaaS), also called Cloud Data, may arise from raw data being extracted and packaged as a side benefit of broad analysis performed by a Cloud provider: this information, generated from the "exhaust" of many consumers of a vendor's primary services, becomes a resource in its own right.

The Following diagram shows the three primary service (without the addition of DaaS) below. The list to the left of the diagram shows types of usage for the various Service levels while the column to the right lists some of the more commonly found Cloud Service specializations and suggests an alignment with the more general Service types.

Figure 3–2 Cloud Service Model Hierarchy



The diagram in Figure 3 presents the Cloud Service types as a hierarchy only to represent the increasing scope of IT services rather than to suggest that the higher levels require the existence of lower levels. On the other hand, if Cloud Services are stacked in this fashion then the typical principle of architectural layering applies in which any layer can only communicate with its immediate subordinate. For example, SaaS should only take information from PaaS and not IaaS. The implication of this is that the PaaS should provide all the management information needed by the consumer regardless of the architecture or implementation of lower levels, thus maintaining separation.

Since these Service models each require a level of abstraction to expose only the necessary executable and management capabilities, it is not a requirement to layer them on top of each other.

For example, consider a company that outsources its Human Resources software needs (that is, it's IT application needs, rather than its HR capability): naturally it is interested in the functionality of the HR software and it also needs to manage (monitor, audit, etc.) HR operations. While this company has an interest in the availability of its HR software (uptime and disaster recovery) and the security of its

data, it is not typically interested in platform efficiency and utilization or allocation of physical resources.

Ideally the service presentation of any Cloud layer should abstract away the underlying implementation in such a way that the management of that layer (or the consumers of its services) should not need to be aware of the implementation details; however, many situations will arise in which optimizations will be required to support specific architectural needs. When these optimizations cannot be provided through a generic interface (e.g. control over physical organization of data across disks) it becomes necessary to design lower layers specifically to support architectural requirements of any given Cloud Service layer (e.g. storage and infrastructure are incorporated within a PaaS rather than layering a PaaS on an IaaS).

In conclusion, creating a Cloud service model at any given level (Software, Platform, or Infrastructure) could potentially benefit from the services provided by another layer (e.g., deploying PaaS on an IaaS infrastructure), but it is NOT essential that these models be deployed together and indeed, the approach may be precluded by constraints arising from the need for optimizations in supporting layers.

3.4 Deployment Models

Cloud computing can also be viewed as divided into four distinct deployment models.

- Private Cloud - Operated solely for a single organization.
- Community Cloud - Shared by several organizations in a related 'community'.
- Public Cloud - Infrastructure owned by an organization selling Cloud services is made available to the general public or industry.
- Hybrid Cloud - The Cloud is a composition of two or more clouds (private, community, or public) bound together by data and application portability.

Most enterprise Cloud efforts will fall distinctly into the two primary models:

1. The small and medium enterprises (SME) market and specific projects or functions of large enterprise may be off-loaded to public Cloud service providers.
2. The large enterprises market, seeking lower cost, flexible IT, but still concerned about data ownership, reliable service etc., will also increasingly move to a private Cloud model.

In most cases the community Cloud is applicable to government service bureaus supporting multiple agencies, but some of this may be applicable for enterprise IT environments supporting acting as a service bureau multiple independent operating companies. Of course the logical nirvana of Cloud computing is the hybrid Cloud, where a significant portion of the IT load runs on internal services, but these applications are flexed elastically and invisibly out on to public Cloud providers. At this point there are still major efforts in establishing industry wide standards for hybrid Cloud APIs and management models, so in the short term the hybrid model is likely to be limited to two areas:

1. autonomous, relatively stateless, low security workloads e.g., web content services or project oriented data processing
2. areas where the public and private Cloud both support a specific proprietary application environment or Cloud management abstraction

The term Hybrid Cloud refers to combinations of computing models. One commonly anticipated scenario for a Hybrid Cloud arises when application development and

testing are performed in a Cloud environment while the business is not initially prepared to put mission critical applications into the Cloud, instead retaining a traditional datacenter environment for production deployments.

3.5 Choosing between service & deployment options

This is essentially a topic for a Cloud assessment model and practitioners guide, however, initial distinctions can be seen in architectural capabilities later in this document.

3.6 Terminology

While many terms are defined in the glossary some clarifications should be understood before proceeding any further through this document.

3.6.1 Services and Service Offerings

Cloud services must not be confused with SOA (or any other) services. The terms “services” and “service offerings” are used in many industries to mean difference things. In the case of Cloud a service is a packaged IT capability provided to consumers as service offerings (by service providers). These terms will be used throughout this document in this context (unless otherwise qualified).

4

Cloud Conceptual Architecture

Unlike application and middleware Technology Strategies, Cloud architecture is heavily focused on provisioning and operations and as such, much detail should be expected in a document about infrastructure; however, we must first take a non-technical view and consider the business needs that a technical solution should address.

The purpose of the conceptual architecture for Cloud is to create a foundation to provide the benefits, conform to the essential characteristics, and address challenges.

This conceptual architecture is completed with a description of architectural constraints, principles and guidelines, and a brief look at technologies and standards available to move the architecture to the next stage - the *ORA Cloud Infrastructure*.

4.1 Requirements of Cloud Architecture

In order to focus this conceptual architecture on business, rather than technical concerns, a summary of business requirements is included here. Not all requirements will have a simple tangible manifestation in a conceptual architecture, but it is still important to start with a checklist of this type to refer to through all stages architecture development.

The end goal of any business computing platform is to utilize a computerized application system to support and augment the running of a business. Many requirements for IT are derived from the foregoing statement, however, a subset of the generic requirements reappear here because they should be further improved by the Cloud architecture.

Common specific business requirements for Cloud include the following:

- Improve the operation of a business (this is a generic IT requirement, but Cloud should deliver improvements)
- Enable effective measurement of business activities
- Support business scale
- Scale with the business growth
- Support (provide or enable) innovation
- Increase efficiency / reduce costs
- Improve quality of the business product
- Improve employee satisfaction and quality of employment
- Longevity (TCO and flexibility to meet future needs)

- Support business agility (accommodate business changes)
- Technological flexibility (support adoption of future technological improvements)

All these are potentially connected in some way to the common generalized goal of increasing shareholder value. The next step is to consider how Cloud help with these requirements.

General Cloud opportunities:

- Abstraction and isolation (from the SOA model)
- Potential for scalability
- Improved capacity management and utilization

Opportunities specific to 3rd party Cloud services ("outsourced" services)

- Allow the business to focus on its core business activities by removing the distractions of IT. Elimination of non-core activities is a common business strategy that has been the justification for IT (and other) outsourcing. Cost restructuring.
- Knowledge - Access to intellectual property and wider experience and knowledge.
- Contracts with 3rd party improves manageability by making services more predictable and protects from underperforming SLAs (financial compensation).
- Access to talent
- Operational expertise
- More choice
- Economies of scale
- Catalyst for change - the outsourcer becomes the change agent
- Liability - Organizations choose to transfer liabilities inherent to specific business processes or services that are outside of their core competencies.

The following conceptual model addresses many of these requirements while the broader disciplines associated with Cloud (and the Oracle Cloud Technology Strategies) also provide benefits associated with standardization, consolidation/rationalization, and simplification. The Cloud architecture and associated disciplines yield measurable benefits in terms of simplicity, reduction of technology and process diversity, etc. enabling organizations to focus their business rather than their IT.

4.2 Conceptual Architectural View

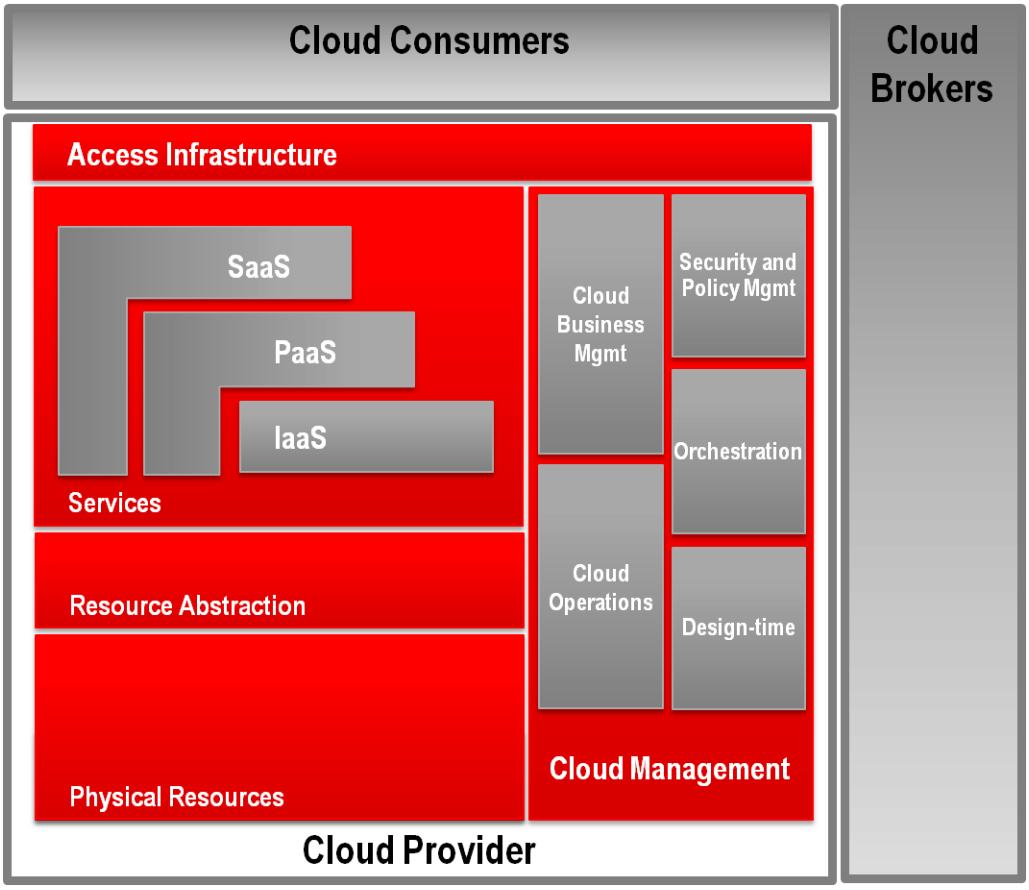
The Oracle conceptual architecture for Cloud has three main areas of focus:

- The interface to Cloud to all the various design-time and runtime consumers, appearing as the Access layer
- Cloud management capabilities which are grouped into the Management layer
- Deployable Entities which consume resources and present them in the Service layer

Underlying computing and application resources are not considered part of the conceptual model, instead they will appear in a Resource layer in the Cloud Infrastructure document.

Figure 4–1 below shows these three layers and key sub-components along with supporting elements (monitoring, security, and virtualized resources), but without concern for their relationships at this conceptual stage.

Figure 4–1 Conceptual Cloud Architecture



Much like in a n-tier architecture model, each of these layers of the conceptual architecture view are intended to address key requirements while still presenting a loosely coupled architecture to satisfy key non-functional requirements such as scalability and high availability.

To understand this architecture, it is important to separate "the Cloud that runs applications" (i.e. the Cloud itself) from "applications that run in the Cloud" (i.e. Cloud services, deployable entities or virtual data centers). This architecture largely centers on "the Cloud", which refers to the capabilities, resources and services that work together to provide IaaS, PaaS, and SaaS services to developers, end users and application owners. It is the infrastructure for "applications that run in the Cloud." As such, this view mainly concerns itself with the characteristics that are important to application owners, Cloud operators and developers, not Cloud application end users.

This conceptual diagram highlights several key considerations for the architecture:

- Deployable Entities can be composed of many types of Cloud service models.
- Clear Separation of responsibilities between Cloud management and resource management.
- The key underpinning elements in monitoring, security, and virtualization.

In the following sections, we will explore the details of each layer.

4.2.1 Access Layer

The access layer enables end users, developers, and application owners access to the service layers (and hosted Cloud applications) as well as the Cloud management interface. This is comparable with a traditional n-tier application architectural model's presentation layer, with similar capabilities and functions. However, unlike in traditional n-tier architectures, this layer exposes many application management capabilities through the Cloud management interface.

4.2.2 Management Layer

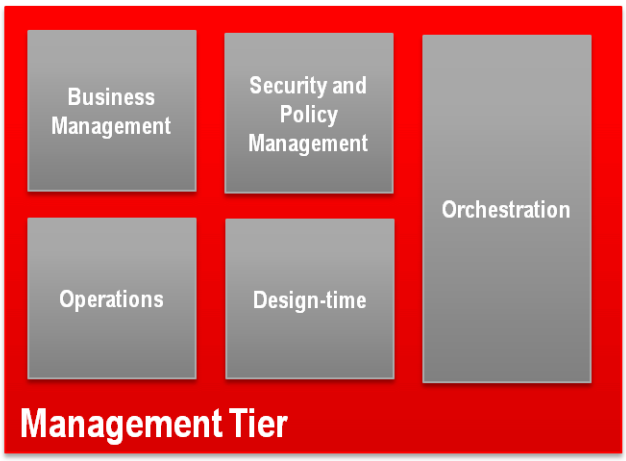
The Cloud management layer exposes the Cloud logic needed to design, provision, and manage Cloud services for developers and application owners. It also provides the control plane for the Cloud operator to manage the underlying Cloud infrastructure. Again comparing this to the traditional n-tier architectural model, this layer could be considered a combination of an enterprise application's business logic and management plane.

Oracle envisions five major areas of concern within the Cloud management layer:

- Business management, which covers the business management aspects of the Cloud.
- Operations, which supports the runtime capabilities for the Cloud infrastructure.
- Security and policy management, which support the security and policy management requirements of the Cloud infrastructure.
- Design-time, which supports the model management and design-time tools
- Orchestration, which provides the capabilities for orchestrating the key management activities within the Cloud infrastructure.

These are shown in Figure 4–2.

Figure 4–2 Cloud Management Layer



It is important to note that the management layer sits between the access and resource layers to mediate all communications. Communication from the access layer cannot talk directly to the resources.

Cloud computing users should not need not have knowledge or expertise in, or control over the underlying infrastructure in the "cloud" that supports the services provided to them. As enterprises migrate their existing IT infrastructures and resources to a sharable cloud paradigm, it is imperative for cloud enablers to provide a uniform API that these enterprises can use to tailor the cloud to their business processes and economic models. As IT deployments becoming more complex, an abstraction of the infrastructure resources become more relevant to address concerns of compliance and configuration. Furthermore, such abstractions enable consumers to both self-serve the exact service they need, and to operationally control these services without any significant administrator involvement.

Such an API enables an infrastructure provider to service its customers by allowing them to

- Browse templates that contain definitions and metadata of a logical unit of service
- Deploy a template into the cloud and form an IT topology on demand
- Perform operations (such as ONLINE, OFFLINE, take backups, etc.) on the resources

The *ORA Cloud Infrastructure document* describes the management layer in great detail.

4.2.3 Services Layer

As discussed in the previous sections, the services layer contains the deployable entities built from the Cloud's Infrastructure, Platform, and/or Software services.

Deployable Entities are instances of useful aggregated resource that consumers can interact with (see section 4.3.2 below).

4.2.4 Resources

The resource layers have two goals:

- Aggregate and manage physical or virtual resources.
- Expose the virtualized or physical resources pools to the management layer so that they can be orchestrated into Cloud services.

Physical and virtual resources may be a variety of simple or complex objects. For example, this may be as simple as a virtualized server or file system mount in an IaaS Cloud. It also may be a much more complex object such as a clustered J2EE container or message queue. The resource layer also represents an important integration point both hybrid clouds and legacy resources.

4.2.5 Cloud Consumers

Cloud consumers in the diagram of the conceptual architecture in Figure 4–1 represents all types of users of Cloud capabilities. Other than the discussion of the access layer (through which the consumers access the Cloud) the consumption of Cloud services is beyond the scope of this document.

A special case of Cloud service consumption (and provisioning simultaneously provisioning) arises in the form of Cloud brokering described in Section 4.2.6.

4.2.6 Cloud Brokers

Cloud brokers represent a special class of both Cloud provider and consumer. Brokers exhibit many of the architecture characteristics of a Cloud provider, particularly in the

management layer, but generally excluding contents of the resource layers. The broker's resources are Cloud services consumed from other Cloud providers and offered (presumably in a modified form) to its own consumers.

As in any other business scenario a broker must provide some added value, ranging from better marketing/visibility of someone-else's services to the composition of unique, high value services achieved through the orchestration of other provides' offerings while potentially incorporating services of its own.

4.3 Architectural Concepts

In order to develop a comprehensive architecture for Cloud we must first understand the capabilities it provides. At this conceptual level, capabilities are derived from statements of expected business benefits and characteristics of Cloud outlined earlier (more detailed, technical capabilities will be described in subsequent architectural views found in the Cloud Infrastructure Architecture document). Capabilities describe what the architecture needs to provide and are used to create the conceptual model in this chapter.

The conceptual capabilities will be used to further develop important functional and non-function requirements within this architectural description. To complete the conceptual architecture description this section finishes with a set of architectural principles that must be followed by all Cloud implementations.

4.3.1 Virtualization

Prior to the emergence of Cloud concepts virtualization was commonly defined along the lines of the following statement:

"Enables multiple operating system environments to run on a single host computer. The environment contains the operating system and all supporting processes packaged together. Virtual environments can be deployed and migrated between physical machines."

In the Cloud paradigm there is a potential for virtualization of many elements of a traditional computer system besides just processor virtualization. Examples include storage virtualization, operating system virtualization, even application virtualization.

Typically, the part that is virtualized is the functional interface of the system rather than the management interface.

4.3.2 Deployable Entities

While many people talk about the Cloud service models as strictly IaaS, PaaS, or SaaS, the reality is that most enterprise level applications will encompass elements of one or more service models (and perhaps even non-Cloud services and resources). For example, while a J2EE application may execute on a PaaS service, it may access data through a non-Cloud storage resource (perhaps a corporate SAN) or database running in a Cloud resource such as Data-storage as a Service, or a Cloud database as in Database as a Service (DBaaS). Some applications may be strictly SaaS applications but might need to extract data to run for end-of-quarter batch processes, which will be run on an IaaS solution. To encompass all of these possibilities, Oracle uses a concept called deployable entities. Deployable entities are a collection of one or more Cloud services (and possibly static resources) and their accompanying meta-data that work together to provide end user computing within the Cloud infrastructure. On a more logical level, deployable entities may consist of:

- Service Offering (Payloads)

- Cloud IaaS, PaaS or SaaS resources
- Non-Cloud IT resources like a corporate storage area network (SAN)
- Resource templates

- Service Contracts (Metadata)
  - Runtime policies
  - Deployment scripts and data
  - Configuration information

While deployable entities are a concept and not a product, subsets of their functionality can be found logically in virtual data centers (vDC found in Oracle's API specifications or Service Offerings in DMTF) and physically within assemblies (produced by Oracle's Virtual Assembly Builder product).

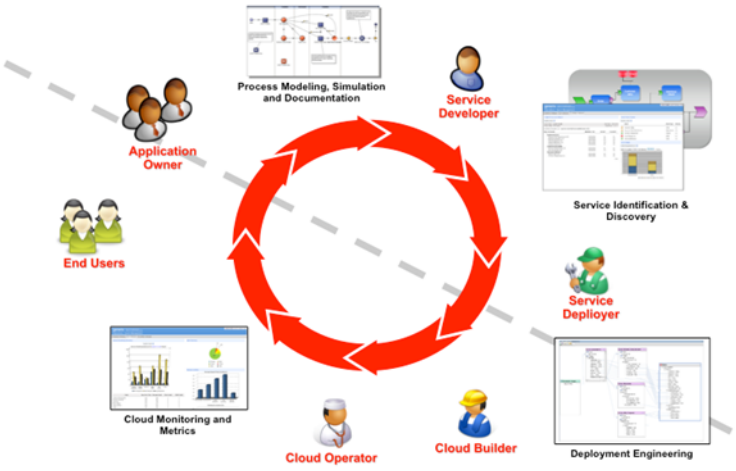
4.3.3 Velocity of Change

In the traditional computing model, an enterprise data center might have 100 servers and these might be configured/re-configured twice a year. These 200 changes can be handled manually or automated with scripts. In the new 'Cloud model,' you may have many more virtual servers and reconfigure machines on an hourly basis to accommodate load. Traditional, manual configuration management processes and systems simply will not be able to keep up with this rate of change.

4.3.4 Rich Support for Both Build-time and Run-time

Unlike traditional deployment environments, the Cloud needs to provide capabilities and infrastructure to support both the build-time of an application (development environment) and the run-time of an application (deployment environment).

Figure 4–3 Build-time versus Run-time Roles



Build-time characteristics may include:

- Standardized packaging tools and formats
- Service catalogue
- Deployment & Control APIs

- Debugging and Validation
- Cloud Service Directory

Key runtime characteristics may include:

- Metering
- Monitoring
- Capacity planning and management
- SLA enforcement

4.3.5 Support for Diverse Roles

Cloud infrastructures will need to support a diverse categories of roles:

- Cloud Service Provider: The operator and provider of the Cloud computing infrastructure. Roles within this category will include Cloud builder and Cloud operator.
- Cloud Service Developer: These are the roles responsible for designing, creating, packaging, and deploying Cloud applications for end-user consumption. Typical roles within this category include service developer, application owner and service deployer.
- Cloud Service Consumer: End-users of Cloud computing. These include the actual users of the Cloud application as well as the application owners.

4.3.6 Heterogeneous Resource Management

A Cloud architecture should be able to support a variety of resources pools and pool managers, whether they are differing types of resource (compute, storage, etc.), different vendors of resources, differing deployment models (public or private), or different products from the same vendor. These should be based upon differing quality of service attributes (performance, cost, availability, etc.) and covered by the service definition that maps to qualities and metered costs.

4.3.7 Multi-Locale Capable

Just as any typical IT environment spans several physical locations, a Cloud architecture will too. However, unlike a current IT environment, the Cloud management will need to be aware of the difference in its physical data centers to appropriately deploy applications. This may be completely hidden from the user (and generally will be in SaaS or PaaS Clouds) or exposed as a configuration option through the deployment interface.

4.4 Architectural Constraints

Very large scale computing of kind already seen at Google, Amazon, and others requires special attention to architecture. Specifically, computing at this scale requires parallelization of data, infrastructure, platforms, and applications.

Traditional database for transaction management adhere to ACID (Atomicity, Consistency, Isolation, Durability) principles. Under this type of database, horizontal scaling mandates distribution of data across many nodes (in a cluster or grid). There are various techniques for distributing data, commonly by separation of tables or columns (the term “partitioning” is used in relational data parlance, but in this discussion the term has a different meaning) or distribution by rows (using

"horizontal partitioning"). Unfortunately, this leads to an availability problem: when a node fails (which may occur frequently in a system of many nodes) "partitioning" is said to occur. A traditional cluster requires redundancy and data replication to protect from this kind of partitioning, however, this approach incurs significant cost and in a very large, distributed system becomes impractical.

Sometimes, Consistency is absolutely necessary (e.g. financial transactions via an AMT), but this is not always the case and an alternative approach can be applied using the principle of BASE (Basically Available, Soft-state, Eventually consistent). BASE is the logical opposite of ACID and sacrifices the guarantee of Consistency in favor of Availability in a system of many nodes. The expression Eventual consistency refers to the optimistic belief that changes will be propagated through systems of replication and distribution over time.

The relational database approaches to scaling applications in this way include optimistic locking (see OCC) and sharding. Optimistic locking can often be applied in environments with low data contention, but if conflicts happen occur, the cost of repeatedly restarting transactions can significantly impact performance. Sharding, on the other hand, distributes table data across multiple instances of a schema. Sharding has the advantage that a search load for the large distributed table can be split across multiple nodes, unlike basic horizontal partitioning of data which is constrained to a single schema. The primary architectural difference is that horizontal partitioning requires tight coupling of servers while sharding applies a shared nothing approach: once sharded, each shard can live in a separate logical schema instance across multiple servers, which may even be geographically separated. As a consequence, sharding is also useful for geographic distribution of applications, where network bandwidth between data centers would otherwise be a bottleneck.

Sharding however, requires a replication mechanism between schema instances, so tables are kept as closely synchronized as the application demands. This means sharded systems require some careful architectural choices .Ideal uses are those in which data is "nearly read-only", meaning updates are rare and performed in batch. In cases where data is changing more frequently, tables may be dynamically replicated, however, this approach reduces some of the benefits.

4.4.1 Brewer's CAP Theorem

Eric Brewer (originally at the ACM Symposium on the Principles of Distributed Computing) claimed there are three core systemic requirements that exist in a special relationship when it comes to designing and deploying applications in a distributed environment. These three requirements are: Consistency, Availability, and Partition Tolerance (CAP).

Consistency and Availability are already well understood (refer to other ITSO docs), but large scale computing mandates parallelization and this causes Partitioning of computing resources (in this context referring to unintended network separation or other node failure) and raises the question of Partition Tolerance (Gilbert & Lynch defined partition tolerance as, "No set of failures less than total network failure is allowed to cause the system to respond incorrectly").

Fundamentally, Brewer says you can't have all three (Consistency, Availability, and Partition Tolerance) all the time and in large scale parallel / distributed systems in which node failure (partitioning) is a statistical inevitability partition tolerance may be more important than ultimate Consistency. That is to say it is sometimes preferable to permit some data inconsistency in favor of performance or avoidance of a complete outage due to a single node failure.

The principles of CAP therefore require a choice between ACID and BASE for every aspect of a systems data and typically, diverse business requirements will lead to a spectrum of approaches that should be applied meet to specific needs. Ultimately, a choice in favor of Eventual consistency pushes the consistency concern off to clients, who will then need to do consistency checking in their own applications.

4.5 Cloud scenarios / use cases

Without regard to unique business requirement that are several archetypal Cloud use case scenarios that both are likely to provide benefits and readily achievable.

4.5.1 Cloud-Bursting

Most IT departments incur substantial cost and effort preparing for peaks in load (e.g. a flood of orders resulting from a marketing campaign) while much of this additional capacity lies idle for the rest of the time. Cloud bursting is a technique that enables a company to run its own IT infrastructure at a capacity that suits a “normal load” while augmenting its compute resources during peak load with resources in an external, shared Cloud environment.

In this way a company is able to substantially reduce capital expenditure, paying only for the Cloud resources during times of peak load. The consumer benefits from this arrangement in particular by optimizing the balance of capital costs and Cloud utilization. While initially retaining the independence of owning its own IT resources, over time it may be determined that the Cloud cost model is so compelling that this approach will be seen as a migration strategy.

At first sight it may appear that the cost of the excess capacity required to meet the company's peak needs has merely been shifted to another business entity; however, as the external Cloud takes on more tenants the effects of load peaks, unique to each tenant, are smoothed to a relatively steady state of consumption.

The primary challenge of Cloud bursting is to establish an application architecture that is most suitable for Cloud (that is to say seamlessly operated in a highly distributed environment). Another important consideration is that the data that the “bursting” application requires needs to already be in the cloud. To avoid significant start-up delays the Cloud must be primed and ready to run. While not permanently consuming CPU and memory resources this data is consuming storage resources even if it's not being used. The necessity for synchronous copying of changes to the data into the cloud will also consume network resources.

4.5.2 Development and Test

Perhaps the most often cited one of these is on-demand development and test environments. The simple act of moving from more dedicated and constantly re-wired development environments to one where pooled resources are dynamically managed, provides several levels of TTM (time to market) impact. This certainly reduces the time to use from the potential months to order and deploy new systems to days or hours.

4.5.3 Elastic Scaling

An ongoing IT challenge is the cost and time to provision an IT environment and the added cost of provisioning for peak loads (which lead to low overall utilization). Pre-provisioning resources in common dynamically configurable shared resources enables on demand scaling. Of course, this assumes that the various peak loads are non-concurrent. Thus, Cloud computing is the new 'consolidation'.

4.5.4 Consolidation

An ongoing IT challenge is the management of resource costs associated with specific applications of services. Dedicated, stovepiped resources, where each application has dedicated excess capacity, results in low overall utilization and higher costs. Historically enterprises engaged in consolidation projects to move multiple applications to consolidated infrastructure. Cloud computing is the logical extension of this enterprise IT strategy, only Cloud computing offers the opportunity to consolidate much larger numbers of applications, databases, services, etc. unto shared infrastructure resource pools.

Much of the planning and implementation is the same. Applications must be inventoried to determine their compatibility and affinity (similar technical requirements, business priorities, etc.) and then projects must be implemented to move these physically isolated services to virtual resources.

4.5.5 Disaster Recovery

Mission critical applications require contingency plans for disaster recovery, often not just for business purposes, but also to meet regulatory requirements. These architectures require not only the additional resources but plans to synchronize data, migrate control and eventually restore original production. Even where costly hot standby facilities can be justified, there are still some failure events that may jeopardize both the production and disaster recovery sits. A Cloud strategy may provide a disaster recovery strategy that not only results in lower costs, but potentially even higher overall reliability levels.

Cloud based disaster recovery could include more shared use of dynamically provisioned private Cloud resources, or could rely on contracts with public Cloud providers, or utilize a hybrid mix of public and private Cloud resources, with the potential of lower cost and more scalable failover. Of course using a public Cloud for disaster recovery would imply some significant baseline costs for pre-provisioning of data and application resources.

4.5.6 Transient Load

Perhaps one of the most discussed enterprise opportunities for Cloud computing is off-loading of peak or transient load from dedicated enterprise (private Cloud) infrastructure to public Cloud providers. Similar to the use case of disaster recovery, an enterprise may potentially be able to build out for dominant average loads and utilize a hybrid Cloud model for peak load. This would be especially attractive where the peak loads are unpredictable and potentially large. As in the case of disaster recovery, this likely requires pre-positioning of applications and data and resulting synchronization challenges. Off course, some peak loads (annual financial close) may be more predictable. In addition, Cloud resources, whether public or private could make an ideal platform for high load associated with temporary projects.

4.6 Architecture Principles and Guidelines

This section contains high-level architecture principles that must be followed for a successful Cloud architecture. Principles are stated as rules that the Cloud architecture must adhere to. For each principle, a rationale is provided to explain why this principle is required. In addition, one or more implications are listed to indicate on what might be involved in satisfying the principle.

Principles are used at various levels of granularity to define the rules that must be followed to keep within architectural constraints, to conform to standards, and

ultimately to create a successful architectural solution. This Cloud Foundation Architecture provides the highest level (most coarse grained) principles, while more implementation specific principles can be found in the Cloud Infrastructure document.

Guidelines are similar to principles except that they are not required in every situation or may be just recommendations (guidelines may not be tracked as rigorously in architecture governance).

The following list is not indented to be an exhaustive set of Cloud architecture principles; instead it provides a starting point for the formulation of more specific Cloud architectures.

4.6.1 Conformity to standards

Principle	Conformity to standards
Statement	Cloud interfaces and formats must conform to relevant industry standards.
Rationale	Conformity to standards is required to ensure interoperability at all levels of Cloud architectural concern (i.e. development, operation, and runtime use).
Implications	■ Standards must be carefully evaluated and selected for alignment with the ultimate goals of the specific architecture. ■ Architects must be knowledgeable about emerging standards to avoid point solutions at the expense of interoperability.

4.6.2 Perceived Simplicity

Principle	Perceived Simplicity
Statement	The system must present only the information (interfaces etc.) necessary to perform each specific function.
Rationale	"Everything should be as simple as possible, but no simpler" (Albert Einstein). All computing systems are inherently complex, but do not need to be complicated (i.e. difficult to understand or to use). Cloud architectures must avoid the overhead that results from complexity in order to achieve scalability.
Implications	■ Complexities (or details) that are not necessary to perform a specific function should be abstracted away from the consumer. ■ It is likely that many abstractions will be necessary to present the system capabilities to different consumers. ■ Automation may be required to simplify many tasks (e.g. expansion of resources). ■ Policies should replace routine questions (e.g. "warn me when we exceed a growth/spending threshold" rather than "how much disk space will you need?")

4.6.3 Visibility

Principle	Visibility
Statement	The architecture should provide monitoring of all aspects of resource usage for the various dimensions required by both the Cloud consumer and provider.
Rationale	The system cannot be managed if it cannot be measured.
Implications	■ Isolating resource usage by consumer application instance will be challenging for underlying system (e.g. disk IO for a single consumer application on a physical bus shared by many tenants with many applications). ■ Profiling application resource consumption might yield misleading results in a shared, virtualized environment (e.g. disk IO wait time is likely to be distorted by other users in a shared environment). ■ The system might be expected to provide "intelligence" options beyond mere resource utilization data.

4.6.4 Transparency

Principle	Transparency
Statement	Any Cloud provider's claims of Reliability, Availability, Security, and Performance must be verifiable.
Rationale	Success of Cloud relies heavily on trust due to the need to abstract many details (such as number of physical servers, location, etc.). Business continuity (RASP, etc.) is vitally important and its parameters must be fully disclosed.
Implications	■ With a potentially variable consumer base, each with different service level requirements, a complex risk model must be constantly updated and necessary actions identified. ■ Independent verification agencies are likely to be needed to avoid false claims that could arise if public Cloud providers are left to police themselves.

4.6.5 Fail in Place

Principle	Fail in Place
Statement	Availability should not be limited by inevitable hardware failures.
Rationale	In cases of extreme scale, availability may be most effectively achieved through software means (ignoring, deselecting, or decommissioning a failed node and dynamically rerouting work) rather than risking potential delays (from fixes, replacement, and restarts) associated with hardware replacement.
Implications	■ Horizontal, pools vs. servers.

4.7 Technology and Standards

The following is a brief summary of relevant Cloud standards.

4.7.1 National Institute of Standards and Technology (NIST)

NIST - Definition of Cloud computing:

http://csrc.nist.gov/groups/SNS/Cloud-computing/Cloud-def-v15.doc

NIST defines four Cloud deployment models:

- public clouds (Cloud infrastructure made available to the general public or a large industry group)
- private clouds (Cloud infrastructure operated solely for an organization)
- community clouds (Cloud infrastructure shared by several organizations)
- hybrid clouds (Cloud infrastructure that combines two or more clouds)

4.7.2 DMTF

DMTF - Cloud computing standards

http://dmtf.org/standards/Cloud

Provides descriptions of roles, services model, and a Reference Architecture.

The DMTF has transitioned its Cloud Standards efforts into a working group and is expected to release a standard IaaS self-service management interface in early 2012.

4.7.3 Storage Networking Industry Association (SNIA)

http://www.snia.org/

The SNIA has created a standard cloud storage interface: Cloud Data Management Interface (CDMI) for Data-storage as a Service (DaaS). CDMI is both a data path to the cloud as well as a control path for managing the data and data requirements via standardized metadata. Advanced data services such as Backup, Archive, and Retention are standardized through this interface.

The SNIA has plans to submit CDMI for international standardization later in 2011.

Storage client consumers interface data either directly (addressing physical storage resources) or through logical containment representations via various standardized and proprietary protocols (e.g. HFS, WebDAV, SQL*net, etc.).

Clients' management of Cloud Storage can be standalone or part of the overall Cloud management.

4.7.4 Others

OGF - Open Cloud Computing Interface (OCCI)

- Standardizes a cloud management model and protocol
- Implemented mainly by universities and research institutes

4.7.5 Cloud APIs

Oracle Cloud API

See
http://www.oracle.com/technetwork/topics/Cloud/oracle-Cloud-resource-model-api-154279.pdf

(soon to be updated based on our actual implementation of it in 12c).

The Oracle Cloud API defines an Application Programming Interface (API) to consumers of IaaS clouds based on Oracle's solution stack.

This Oracle Cloud API enables an infrastructure provider to service their customers by allowing them to:

- Browse templates that contain definitions and metadata of a logical unit of service
- Deploy a template into the Cloud and form an IT topology on demand
- Perform operations (such as ONLINE, OFFLINE) on the resources
- Take backups of the resources

The RESTful (Representational State Transfer) API presented here focuses on the resource models and their attributes.

The specification of this Cloud API includes:

- Common behaviors that apply across all requests and responses, error messages, common resource attributes
- Resource models, which describe the JSON data structures used in requests and responses
- The requests that may be sent to Cloud resources, and the responses expected. Common behaviors would not be described for each resource

4.7.6 MapReduce and Hadoop

MapReduce and Hadoop are archetypal applications for the Cloud. These applications are not “Cloud” in themselves (a common misconception), but instead they are good examples of highly scalable software architecture best suited to massively parallel Cloud infrastructure.

MapReduce is a software framework invented by Google to solve the problem of search across massive datasets. It employs a parallel application architecture designed specifically for loosely coupled distributed computing on a massive scale, working on large data sets operated on by clusters of commodity (cheap) computers.

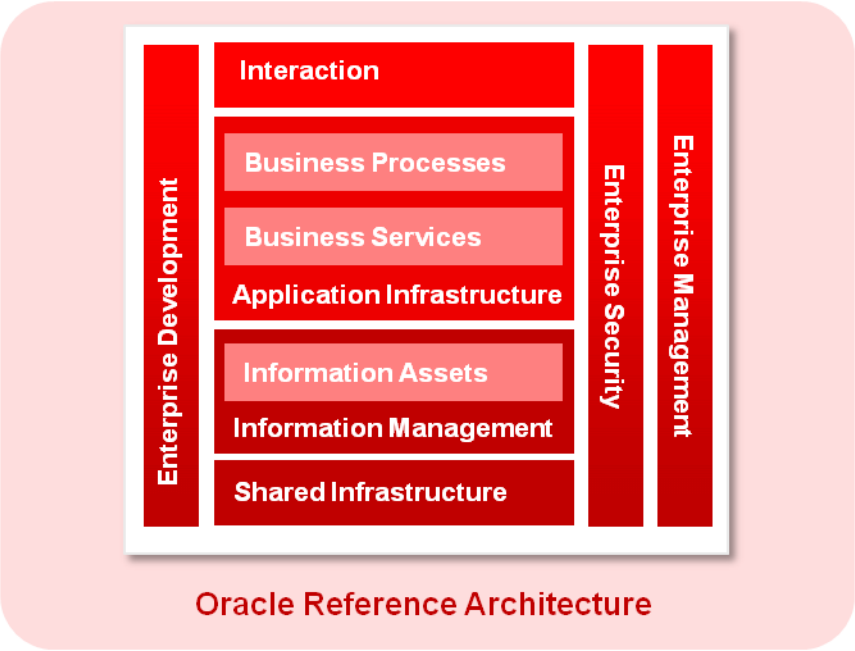
Hadoop is an open source software framework for data-intensive distributed applications. It enables applications to work with thousands of nodes and petabytes of data. Hadoop was inspired by Google's MapReduce and Google File System (GFS) papers. Hadoop is an Apache project written in Java programming language. Yahoo has been the largest contributor to the project, and uses Hadoop extensively across its businesses. Yahoo is in the process of launching a commercial product focusing on creating a Hadoop-based product for enterprise adoption. the joint-venture will be named “Horton Works”.

5

ORA Interlock

IT Strategies from Oracle (ITSO) was introduced at the beginning of this document as Series of documentation and supporting collateral designed to help organizations plan, execute, and manage enterprise architecture and IT initiatives. At the core of the ITSO model is the Oracle Reference Architecture (ORA) shown in Figure 5–1.

Figure 5–1 Oracle Reference Architecture



ORA is a single, unified reference architecture spanning the full Oracle technology space and for each element of the diagram it defines the following architectural characteristics:

- Architecture Concepts
- Principles & Guidelines
- Architecture Views
- Component Drill Downs
- Product Mappings

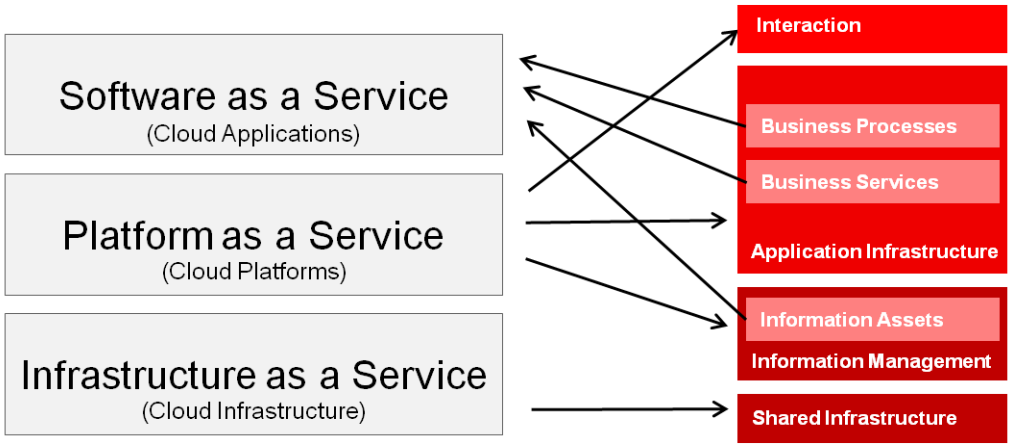
The ORA Cloud perspective documents provide the definitions for these architectural characteristics where they are unique to Cloud (relying on the common ORA core definitions where there is uniformity with more general IT architecture) and thereby extending ORA to support the Cloud Enterprise Technology Strategy (ETS). This document, the ORA Cloud Foundation, has focused on Cloud concepts, definition of terms, principles and guidelines, and standards. The following sections briefly describe the role of Cloud architecture within each of the elements of the ORA model.

5.1 ORA Horizontal Layers

The horizontal layers of the ORA model correspond to the most common layers of any modern architecture. Each layer relies on the layer below it and while traversing up the stack the layers provide more refined capabilities towards the ultimate goal of providing business services.

In general the horizontal layers or of the ORA model align with Cloud service models as depicted in Figure 5–2.

Figure 5–2 Mapping Service Models to ORA



The left-to-right arrows in the diagram in Figure 5–2 indicate an opportunity for a particular Cloud deployment model to support an ORA architectural grouping (e.g. PaaS may support Information Management, Application Infrastructure, and/or Interaction). The right-to-left arrows show the more specific elements of the ORA model providing capabilities within a Cloud deployment model.

Unlike the ORA stack however, Cloud service models are not dependant on each other (as explained in Section 3.3 of this document). What this means is that, any layer of the ORA model may be provided as a Cloud service, but it may in turn be supported, above and below, by the traditional architectural strategies of ORA model. For example, a SaaS Cloud may supported by traditional architectural approaches described by ORA information management and accessed using ORA interaction.

5.1.1 Shared Infrastructure

The bottom of the ORA stack bears the most obvious association with Cloud computing and this is clearly the main area of focus for IaaS. Shared infrastructure, however, takes many forms, some of which do not necessarily conform to Cloud's essential characteristics, so the ORA core architecture provides a broader, more generic, definition.

5.1.2 Information Management

Information Management may be satisfied by a number of variations on the theme of Cloud Data such as, Data-Storage as a Service, Database as a Service, Information as a Service, etc.

5.1.3 Application Infrastructure

Application Infrastructure mainly encompasses Platform as a Service (PaaS), but the inclusion of business processes and business services extends the scope of this layer to include Software as a Service and its associated business service variations.

5.1.4 Interaction

ORA Interaction describes the architectural strategies for human interfaces to business applications and services. The architecture of human interfaces is primarily addressed by Cloud applications (e.g. web interfaces to business applications), however, other delivery channels, such as mobile, TV, etc., may be provided by Cloud infrastructure and/or platforms.

5.2 Supporting Capabilities

The vertical layers of the ORA model represent the supporting capabilities of any system. Two of the three ORA vertical layers (Management and Security) are represented in the Cloud conceptual model while development, in the context of Cloud, is not sufficiently unique to be covered in this document.

6

Summary

Cloud is a broad and complex IT strategy with origins in hosting and SOA, born out of simplistic service proliferation by the likes of Google and Amazon. If Cloud is to become the next paradigm shift for IT then it needs to address the broader needs of enterprise applications along the way to the utopian state of utility computing.

In addition to providing fundamental definitions of Cloud concepts this document has attempted to identify the benefits available towards meeting the needs (and addressing concerns) of business consumers of IT through the use of Cloud. The output of this approach is a business-focused, non-technical conceptual architecture that provides the foundation for the subsequent document, the ***ORA Cloud Infrstructure*** document.

A

Further Reading

The ***IT Strategies From Oracle*** series contains a number of documents that offer insight and guidance on many aspects of technology.

In particular, the following documents may be of interest:

ORA Cloud Foundation is part of a series of documents that comprise the Oracle Reference Architecture, which is included in the IT Strategies from Oracle (ITSO) collection. Although this document was written as a standalone document and contains sufficient background information to introduce the topic of Cloud, it assumes the reader is familiar with supporting concepts.

A.1 Related Documents

Oracle Reference Architecture (ORA) consists of a set of documents, each targeting a specific aspect of the Oracle computing environment. This document covers aspects of Event Driven Architecture.

The reference architecture material also includes the following related documents:

ORA Monitoring and Management - This document provides a reference architecture for designing a management and monitoring framework to address the needs for the modern IT environment.

ORA Security - The ORA Security document describes important aspects of security including identify, role, and entitlement management, authentication, authorization, and auditing (AAA), and transport, message, and data security.

A.1.1 Suggested Pre-reading

The following documents are suggested pre-reading for those that would like to more fully understand the concepts this document builds upon.

- ***ITSO Overview*** - provides an understanding of the scope and documentation approach of ITSO and ORA.

Glossary

The following Cloud specific terms and abbreviations are included here for easy reference. Please see the ***ORA Master Glossary*** for other terms used in the various ORA documents.

CAPEX

A common business term meaning capital expenditure. A capital expenditure occurs when a business spends money on tangible assets.

LAMP

Linux, Apache, MySQL and PHP (Perl, or Python). A very common and highly portable, free, open source, server software stack making up the core components of a general purpose web server.

NUMA

Non-Uniform Memory Access: a computer memory design in multiprocessing which takes advantage of the fact that a processor accesses local memory faster than memory allocated to (or shared with) another processor.

OPEX

A common business term meaning operational expenditure. Operational expenditure refers to the running costs of a business.

SMP

Symetric MultiProcessing: a computer hardware architecture in which two or more CPUs shared the same memory and are controlled by the same operating system.

Optimistic Concurrency Control (OCC)

Also known as optimistic locking is a concurrency control method that assumes that multiple transactions can complete without affecting each other, and that therefore transactions can proceed without locking the data resources that they affect. Before committing, each transaction verifies that no other transaction has modified its data. If the check reveals conflicting modifications, the committing transaction rolls back.[1]

OCC is generally used in environments with low data contention. When conflicts are rare, transactions can complete without the expense of managing locks and without having transactions wait for other transactions' locks to clear, leading to higher throughput than other concurrency control methods. However, if conflicts happen often, the cost of repeatedly restarting transactions hurts performance significantly; other concurrency control methods have better performance under these conditions.