

Cloud Computing: the Rise of Service-Based Computing

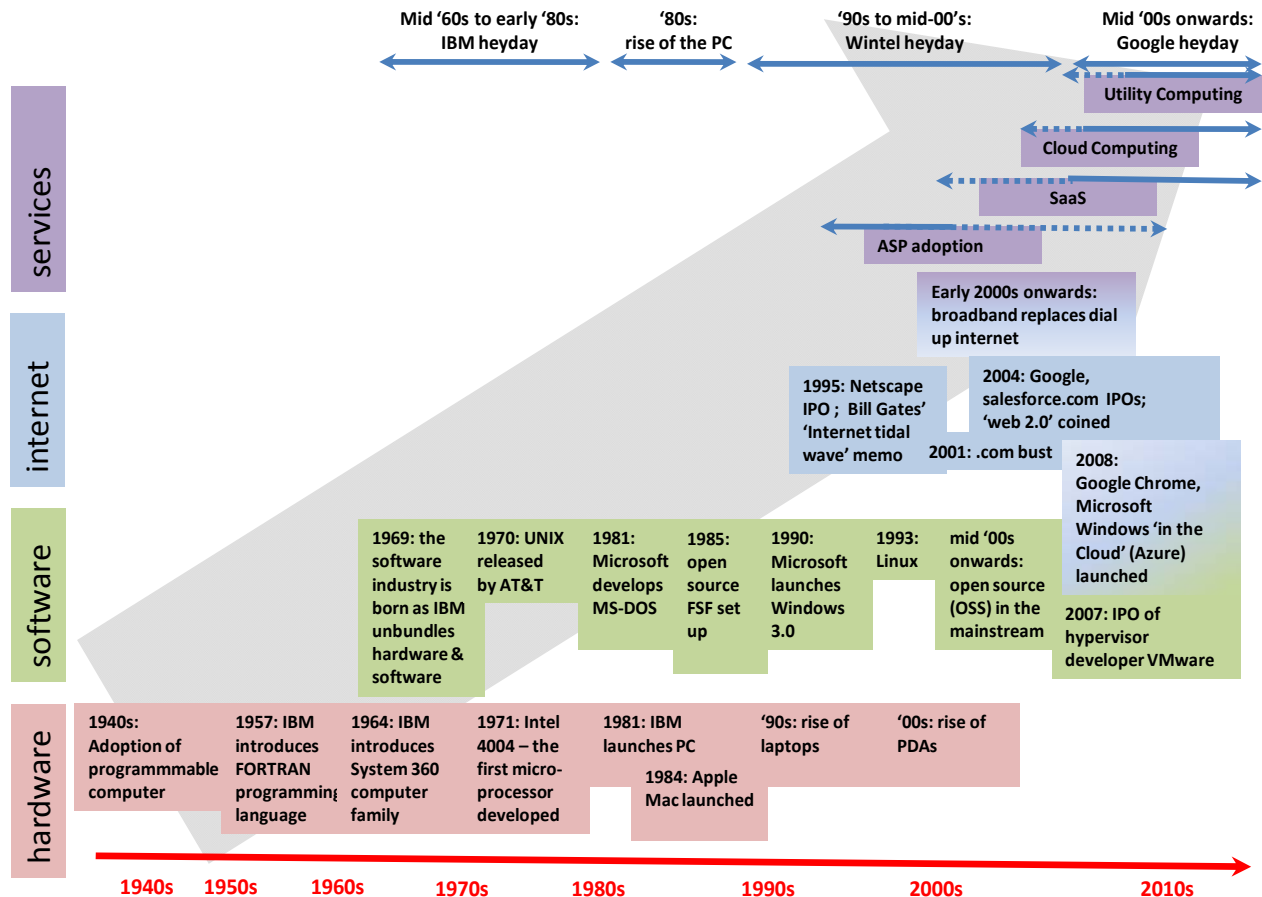
Introduction

Another year, another set of TLAs (three letter acronyms) and buzzwords. This year in service-based computing it’s SaaS, SOA, ASP, virtualisation and the hypervisor, and on demand, cloud, grid and utility computing – and if you want to go straight to the answers, see the short glossaries in the three tables in this ‘hot topics’ article.

This article considers cloud computing in the context of the rise of service-based computing generally and reviews what it is, where it has come from and where it is going; then considers the associated software deployment techniques like OSS, virtualisation and SOA that are accelerating the availability and adoption of service-based computing generally; before briefly looking at the ins and out of service contracts.

Service-based computing is all about the Internet, a digital river in full spate. The Internet is enabling new service delivery techniques for software that are displacing the traditional software licensing models that have held sway since the software industry was born when IBM first unbundled software from hardware in 1969 (see figure 1: the rise of service-based computing).

Figure 1 – the rise of service-based computing



The four stages of service-based computing: ASP, SaaS, cloud and utility

Easy to use, faster, and more secure, reliable and scalable broadband Internet communications connections lie at the heart of the rise of service-based computing, with ASP (application service provision), SaaS (software as a service), cloud computing and utility computing as stages one to four. Table 1 below describes the key attributes of each stage.

Internet access, periodical payment on a subscription basis, and provision on a ‘one to many’ basis characterise all these stages, but ASP is properly characterised as ‘hosted application management’ and although still popular today is viewed as rather cumbersome outside the enterprise (large system/requirement) market.

There is a step change between ASP and SaaS, in turn characterised by:

- ‘built for the web’ applications;
- supply of ‘web ready’ services for the customer;
- addressing consumer as well as professional markets;
- supply through a ‘one to many’ secure delivery model;
- incremental ‘pay per use’ basis.

Table 1: four stages of service-based computing

<p>1. ASP (Application Service Provision)</p>	<p>The first stage of service-based computing, enabled by the advent of faster modems and routers from the mid 1990s onwards. ASP substituted software on servers at the customer’s computer room with software on servers at the ASP supplier’s centrally managed (‘one to many’) data centre, plus Internet access and an HTML web interface at the client PC. Properly viewed as ‘Hosted Application Management’, ASP has benefited from bigger bandwidth but still relies (at the start of the relationship) on significant configuration/installation services and (throughout) on support services, typically supplied separately.</p>
<p>2. SaaS (Software as a Service)</p>	<p>The second stage of service-based computing. SaaS is an Internet-based, ‘built for the web’, ‘web ready’, ‘one to many’ model for secure, centralised software delivery. SaaS benefits from faster and more extensive development and feature updating. It is typically priced on a periodical, per-seat/user basis, scaled according to service features, resilience level and storage space. The generic nature of the SaaS model makes it inherently scalable and adaptable to different lines of business (like CRM - Customer Relationship Management – HR/payroll and Accounts) and market segments (consumer and small/medium business, in addition to larger organisations). In accessing software in this way, a customer does not need to buy/license, install or run the software on its own computers and so eliminates the need to maintain or update the software. <i>Example:</i> Salesforce.com with its market leading SaaS/cloud computing CRM service.</p>
<p>3. cloud computing</p>	<p>The third stage of service-based computing. The ‘cloud’ is the traditional metaphor for the Internet, and cloud computing is the mass market availability through the Internet of the whole range of scalable computer technology-enabled resources provided as a service. <i>Examples:</i> MySpace, searching for flights online. The release of Google’s web browser Chrome and Microsoft’s Windows Azure (Windows in the Cloud) look like a major inflection point along the trajectory from software as a licence to utility computing.</p>
<p>4. utility computing</p>	<p>The fourth stage of service-based computing. Utility computing is the aggregation and packaging up of different computing resources (input, processing, storage, programming, output, communications, etc) for supply on a metered basis, like electricity or any other utility.</p>

The evolution from SaaS to cloud computing is marked by the ‘massive scalability’ (in the words of a Gartner report from 2008) of computing services provided through the cloud, the traditional metaphor for the Internet as the communications network between the customer and the computer resources it is accessing. In

practical terms, the ‘massive scalability’ of cloud computing means moving to bigger and bigger data centres – think of TV images of futuristic, low, windowless warehouses the size of five football fields on the banks of the Columbia River in Oregon, USA. The CRM service of Salesforce.com is classic market leading SaaS and cloud computing service.

Line of business based services likely to prove fertile ground for the uptake of cloud computing include, in addition to CRM, HR and accounts, web-conferencing and collaboration; supply chain; budget/expenses management; web content management; e-commerce; and email and email marketing. Social networking sites like MySpace and flight searching services are contemporary examples of cloud computing. Arguably at the ‘hockey stick’ inflection point of demand growth right now, Merrill Lynch in May 2008 predicted that the cloud computing market could reach \$160bn by 2011.

Utility computing will perhaps at some point in the next decade represent the ultimate stage in this journey, where all the computing resources needed for performing particular business and consumer tasks – data input, processing, programming, storage, output and communications - will be able to be assembled and packaged into the required customer service for supply on a metered basis like electricity.

Cloud and utility vs grid and ‘on demand’

In order to clarify the confusion around taxonomy on service-based computing, table 2 aims to complete the picture by distinguishing cloud computing and utility computing (on the one hand) from ‘grid computing’ – a form of distributed computing – and ‘on demand computing’ - a portmanteau term that can be used to cover any computing not based on locally resident and used software – on the other.

Service-based computing, particularly the stages after ASP, means that the customer does not need to make significant investments in its own infrastructure and resources – servers, space, security, staff, etc – as it does in order effectively to use a particular software developer’s licensed-in solution. Supplier dependence is reduced, and switching to an alternative service provider becomes a practical possibility. Avoiding the need for high sunk costs and the ability to switch suppliers are direct consequences of the rise of service-based computing and represent far-reaching changes to the software business model. They will lead to more choice, greater competition, faster innovation and lower prices. These trends will develop more quickly in the straitened economic climate likely to prevail over the next few years where the accent is very much on using technology to reduce business cost and risk.

Table 2: cloud and utility vs grid and on demand computing

cloud computing	The ‘cloud’ is the traditional metaphor for the Internet, and cloud computing is the mass market availability through the Internet of the whole range computer and communications technology-enabled resources provided as a service. <i>Examples:</i> MySpace, searching for flights online.
grid computing	Effectively a form of distributed computing that constitutes a virtual (i.e temporary) computer capability from separate computers, grid computing is a type of platform virtualisation involving the harnessing of a number (which may be very many) of different computers by means of software that allocates the work and tasks to be carried out.
on demand computing	A generic name for computing that is service-based – including ASP, SaaS, cloud computing and utility computing . <i>Example:</i> ERP software provider Oracle offers its software as a licence, or ‘on demand’ as a managed subscription service - ‘Your Oracle software managed by Oracle’s experts’.
utility computing	Utility computing is the aggregation and packaging up of different computing resources (input, processing, storage, programming, output, communications, etc) for supply on a metered basis, like electricity or another utility.

Virtualisation, OSS and SOA: accelerators for service-based computing

The rise of cloud computing is being accelerated by a number of rapidly evolving software development, assembly and integration techniques that all themselves benefit from and rely on the Internet. These techniques are virtualisation, OSS (open source software) and SOA (service oriented architecture) and they are described in table 3.

Essentially, virtualisation’s software hypervisor enables different computers to be yoked together; and individual servers can be used more efficiently by replicating separate instances of the server’s operating system to ‘reach the parts ordinary software cannot reach’ and use otherwise unutilised resources of the server, raising sometimes tricky licensing issues for software licensed on a per server basis.

OSS¹ is a software licensing technique which can come with a sting in the tail, enabling IT departments to download free of charge from Internet sites like <http://sourceforge.net> software that performs basic tasks so that their expensive in house programming resource can work in higher value projects: free availability makes OSS the CIO’s boon, the licensing sting in the tail makes it the GC’s burden.

SOA consists of three key elements - orchestration (the menu), the Enterprise Service Bus (ESB - the message centre) and application software - which together enable users to map their required business processes to software able to perform those processes and to associate the processes and software by selecting, linking and sequencing them in the right way. The way SOA works is ideal for the remote and distributed processing central to service-based computing and SOA is currently undergoing significant increase in take up.

Table 3: virtualisation, OSS and SOA - software development, assembly and integration techniques accelerating the rise of service-based computing

Hypervisor	In the context of virtualisation , the hypervisor is the software that allows the creation or supervision of multiple virtual operating systems running simultaneously on the same computer – effectively creating multiple virtual platforms on the same hardware. <i>Example:</i> VMware’s ESX Server 3i hypervisor.
OSS (Open Source Software)¹	Software provided under a licence that meets the three key requirements of the Open Source Definition (OSD – see www.opensource.org/docs/osd): <ul style="list-style-type: none"> • the software must be available for redistribution without payment; • the software must be distributed either with the source code or well publicised access to the source code; and • software modification and distribution of derived works must be permitted.
Metadata	‘data about data’: digitised classification data which describes the context, content and structure of other digitised data – including business and personal data, music, films, video, etc (see SOA).
SOA (Service Oriented Architecture)	SOA is built (<i>architected</i>) around associating (<i>orienting</i>) the business processes (<i>services</i>) in the customer’s requirement with the particular services and processes performed by the supplier’s software. SOA has three essential elements: <ul style="list-style-type: none"> • orchestration software: effectively a Metadata list of available application software to choose from, ‘orchestration’ is the software representation of the process of selecting, linking and

¹ A general discussion of OSS is outside the scope of this article, but for further information see Kemp Little Introduction to Open Source at http://www.kemplittle.com/PDFs/Article_IntroductionToOpenSource.pdf

	<p>sequencing the services to be performed by the application software to meet the customer’s business process requirement;</p> <ul style="list-style-type: none"> • the ESB (Enterprise Service Bus): ‘middleware’ software that sits below the orchestration software and above the application software. The ESB is a messaging framework that enables the available software applications and languages (e.g. XML, FTP, JMS, Web Services, JDBC, HTTP) to be connected and data exchanged between them. More technically, the ESB provides the abstraction and messaging systems layers that enable integration; and • application software: the software that sits under the ESB that is selected through the orchestration software and integrated through the ESB to perform the functions required by the customer. <p><i>Example:</i> Starwood Hotels and Resorts Worldwide new SOA service to replace its legacy room-reservation system.</p>
virtualisation	<p>The technique of using software to run one or more operating systems on a host computer, including operating systems not written specifically for that hardware (<i>platform virtualisation</i>) or to reach the computing resources that ordinary software cannot reach by aggregating a large number of individual computing resources into a smaller number of more powerful resources (<i>resource virtualisation</i>).</p>

Service-based computing: legal issues checklist

Just as software customers have got used to the ins and outs of software licensing and all the contractual points arising, along comes the new service-based computing technique. But although they come in a software or computing wrapper, most of the contractual points customers and suppliers will need to think about will be familiar from other ‘grown up’ services contracts. By way of checklist, they can conveniently be grouped under four heads – supplier stability issues (what if the supplier goes bust?); customer service/dependence issues (what’s the worst that can happen?); lifetime contract issues (redressing the balance); and regulatory issues (data protection, data security, audit, sector-specific and generally applicable regulation).

Supplier stability – what if the supplier goes bust? From the customer perspective, things that the customer will need to think about in current economic conditions include:

- supplier financial stability: do your credit searches (bearing in mind that information more than 3 months old is likely to be too out of date to be of much help) and customer and supplier references;
- supplier dependence: check whether the supplier itself depends on particular resources and the financial stability of the providers of those resources to the Supplier; and
- what are the supplier’s own disaster recovery/business continuity arrangements?

Customer service/dependence – what’s the worst that can happen? Review, assess and if necessary remediate:

- customer’s dependence on the supplier: what will be the impact on the customer’s business in each case of a small, moderate or severe service outage; and what happens if the services are not performed at all or up to scratch?
- customer’s ability, time required, etc to switch to an alternative source of supply;
- ensure the customer can effectively monitor and operate any supplier contract requirements on security, passwords, etc: it is embarrassing if staff are swapping passwords to access the service where this is not permitted under the contract;
- as ever, think through the exit/disengagement strategy before contracting, and make sure the customer puts in place an effective, workable exit plan. Key questions include:

- what are the supplier’s commitments on return of customer data both during and after the contract?
- in what form will the data be returned?
- how long is it from customer request to data return?
- will the customer be able to use the data easily in the form in which it is returned?
- if not, what extra needs to be done to make sure the customer can effectively use that data?
- will the supplier ‘play nice’ whatever the reason for the return of the data – and deal with the customer’s replacement provider as supplier’s successor if the customer requests?
- make sure the customer has full, unrestricted right and title to intellectual property in the customer data and that the supplier is appropriately bound by confidentiality and non-disclosure obligations.

Lifecycle contract issues – redressing the balance in negotiations. SaaS and cloud computing providers’ standard form contracts are generally long on supplier rights and customer obligations and short on supplier obligations and customer rights. That is because they have many contracts in the field, so the agreements become a sort of probability theory with the accent on generics. The customer of course wants a deal that meets its requirements with the accent on specifics. Against this background consider to what extent the customer can realistically negotiate a better deal (which in the service-based computing world is likely to involve higher fees) on:

- service levels and service credits for performance below the contracted standard;
- pricing and price benchmarking, etc during the agreement;
- changes the customer may want to make at any time or times during contract lifecycle;
- business continuity/disaster recovery – the customer may get enhanced back up service if it is prepared to pay the higher fees. Don’t forget to test at least annually (and more frequently in appropriate cases) the resilience of the back up and data return arrangements put in place under the agreement;
- as customer, check that you’re happy with the position on:
 - supplier use of sub-contractors to provide the service; and
 - communications requirements – are you buying VPN or other Internet access services from the supplier or is the supplier saying this is down to the customer? If the supplier is providing them, check they’re adequate (and try and get the benefit of market pricing pressures, especially in long term deals, as bandwidth prices decline). If not, make sure your communications requirements are adequate for, and ideally back to back with, the services supply arrangements.

Regulatory issues to be addressed in contract negotiations. Regulatory issues are taking up more paper in service-based computing deals. Consider:

- treatment of personal data and compliance with applicable data protection laws. Does the agreement cover personal data? If so, will it be processed outside the EEA – in the USA or India for example? If so, who is the data controller and who is the data processor, and how are the transborder data issues dealt with?
- what are the protections for, and controls on, data security?
- in what circumstances can third parties obtain contract data?
- if customer is operating in a particular sector (for example, financial, professional or healthcare services) make sure the supplier is required contractually to comply with the regulations that apply by operation of law to the customer and endeavour to ensure back to back compliance; and
- ensure adequate audit rights and that these enable customer to comply with any audit carried out by the customer’s regulator.