calero mdsl

WHITE PAPER

The History, Evolution and Future of SaaS

www.caleromdsl.com

Table of Contents

The History, Evolution and Future of SaaS

Long before the introduction of the Apple Computer, the IBM Personal Computer taught us that software is something that comes packaged in shrink-wrap, software was delivered as a service from "service bureaus" that enabled connected "dumb terminals" to "time-share" on their mainframe systems.

In 1960 John McCarthy, the same Stamford/MIT professor who would first coin the term "artificial intelligence," proposed the idea of a "time-sharing" operating system to give many users access to a new IBM 709 "transistorized computer" that would soon be delivered.

Users would sit at their terminals connected to the mainframe and they would select "jobs" to run that consisted of software stored on the mainframe processing data workloads also resident on the mainframe. Worst case, new software and new data were input into the mainframe using cards punched with holes in the "Hollerith" code, or paper tape.

It's Not the Hardware, it's the Software

The aforementioned Apple Computer languished for some time as a novelty shared among the members of the fabled Home-Brew Computer Club operating out of Menlo Park, CA. Commercial sales to the public were unremarkable. Most people thought of computers as those big multi-million dollar machines that big companies used.

That thinking, and the world itself, changed in January 1978 when Harvard Business School student Dan Bricklin and his MIT colleague Bob Frankston formed the Software Arts corporation which soon partnered with entrepreneur Dan Fylstra of Personal Software to bring to market VisiCalc – The Visible Calculator. The Fylstra connection stemmed from his being the Apple II computer Bricklin and Frankston borrowed to build their first VisiCalc prototype.

It was prophetic that Fylstra referred to his company as the "publisher" of VisiCalc. Software wasn't something that was "published." Software was something one ran on a distant computer they had leased access to through time-sharing. Software was delivered as a service. From a service bureau.

Fylstra would publish VisiCalc on magnetic media, floppy disks, that would be inserted into the Apple or IBM PC and run locally. The product itself changed the world. How it was delivered was quietly revolutionary.

Application Delivery & Software Streaming

What goes around comes around. What's old is new. This is said about everything, but nowhere is a clearer example offered than in the case of software.

One visionary who foresaw the limitations of packaged software was former and long-time Citrix CEO Mark Templeton. Credited with creating such key terms as "client/server computing" and "thin-client", Mark is a constant proponent of "any-any-any" computing. That is, any software running on any device across any network anywhere and at any time.

From the beginning, the Citrix concept of application delivery required software to actually be executed on a server with the results streaming to and from a client device with minimal processing capability. All the processing took place on the server with the results viewed on the "thin" remote device operated by the user. Sound familiar? Software was once again being delivered as a service, one that could be highly managed, finely tuned, and well-supported.

A Brief Stopover in the 1990's

In the early 1990's, Robert Frankenberg was CEO of Novell, at that point the largest IT networking platform. At the outset of a presentation, Frankenberg planted his flag in the history of computing.

"Work," intoned Frankenberg, "is an activity. Not a destination."

It would take the world 26 more years and a global pandemic to catch up with him.

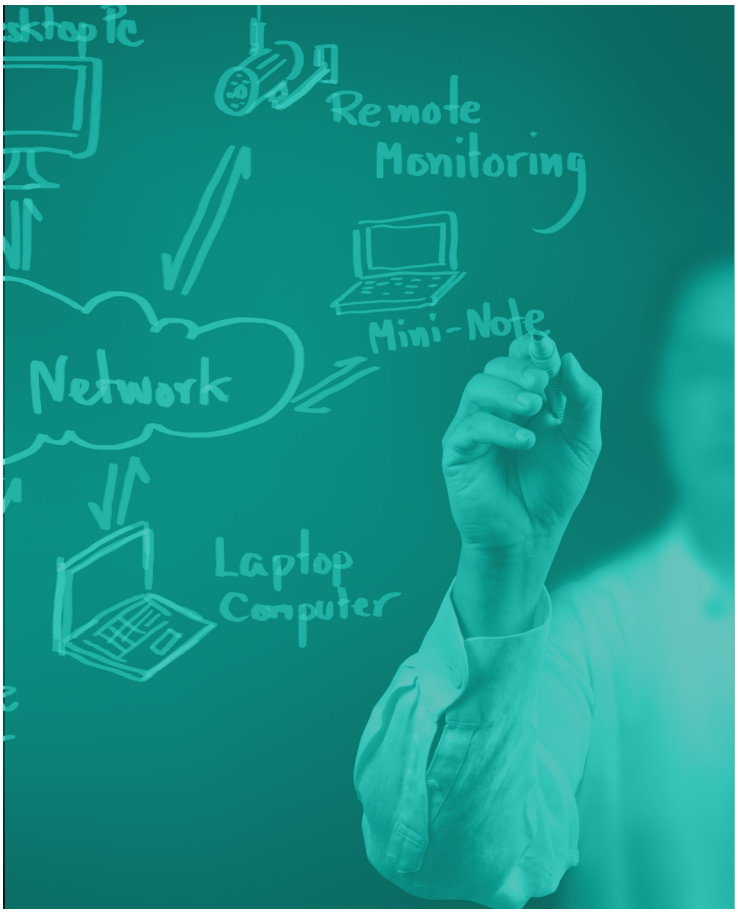
A False Start

Many IT industry veterans would rather forget the Application Service Provider (ASP) initiative that rose and fell during a brief period. The driving force behind ASP was the commitment that no company wanted to run its own IT. Nobody wanted to own and operate servers, storage, switches, routers, they simply wanted their people to use them. In fact, one of the earliest programs was named Utility Computing in which HP attempted to create a parallel between the power, phone, or gas utilities and information technology. Always available login instead of dial-tone. Just plug in and compute.

The creators of ASP failed to take one key element into their planning; cost! ASP moved the infrastructure which was still user-owned to a remote data center. The infrastructure costs continued to accrue now joined by transport costs to connect the users to the remote data center across leased lines. In fact, ASP ended up being far more expensive than traditional in-house computing.

The ASP movement also lacked perspective into something that would become all too apparent for them later on. The core problem with ASP was that it was "single-tenant", that is, one customer paid all the expenses attendant to the server and storage they were using. The entire cost of operation was significantly higher than before, and they shared it with nobody.

This would be resolved in dramatic fashion with the introduction of server virtualization. Now, many, many instances of server could be run on the same piece of server hardware. All that was required was "hypervisor" software enabling the processing of multiple "instances" of server. An enterprising cloud entrepreneur could build the digital equivalent of an office or apartment building, renting out a server "apartment" from the virtualized machine. Multi-tenant occupancy brought with it shared cost-recovery. Each "tenant" paid a fraction of the actual operating cost to receive more service than ever before, yet each cloud provider received far more income from all the tenants taken together. Better service at lower cost; the cloud revolution was in full swing.



Along Comes Cloud Computing

Legend has it that an IBM engineer was whiteboarding a network diagram for an audience of non-technical people. When he reached the point where he was supposed to draw the internet, he instead just drew a "blob" to represent it, explaining that the internet itself was far too large and complex to illustrate. A member of the audience thought the "blob" looked like a "cloud" and the rest is, as they say, history.

On October 15, 2011 the National Institute for Standards in Technology (NIST) issued what would be one of, if not their most important press release ever. It began:

"After years in the works and 15 drafts, the National Institute of Standards and Technology's (NIST) working definition of cloud computing, the 16th and final definition has been published as The NIST Definition of Cloud Computing (NIST Special Publication 800-145)."

No history of SaaS would be complete, or even exist, without this pivotal four-page document. It may be that no more concise a description of cloud computing exists than in this introductory press release:

"Cloud computing is a relatively new business model in the computing world. According to the official NIST definition, "cloud computing is a model for enabling ubiquitous, 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."

The discussion then turns to the core fundamentals of this definition.

"The NIST definition lists five essential characteristics of cloud computing: on-demand self-service, broad network access, resource pooling, rapid elasticity or expansion, and measured service. It also lists three 'service models' (software, platform and infrastructure), and four 'deployment

models' (private, community, public and hybrid) that together categorize ways to deliver cloud services."

The three service models are further defined as 'software-as-a-service (SaaS)', 'platform-as-a-service (PaaS)', and 'infrastructure-as-a-service (IaaS). The definition of SaaS is:

"Software as a Service (SaaS). The capability provided to the consumer is to use the provider's applications running on a cloud infrastructure.

(A cloud infrastructure is the collection of hardware and software that enables the five essential characteristics of cloud computing. The cloud infrastructure can be viewed as containing both a physical layer and an abstraction layer. The physical layer consists of the hardware resources that are necessary to support the cloud services being provided, and typically includes server, storage and network components. The abstraction layer consists of the software deployed across the physical layer, which manifests the essential cloud characteristics. Conceptually the abstraction layer sits above the physical layer.)

The applications are accessible from various client devices through either a thin client interface, such as a web browser (e.g., web-based email), or a program interface. The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, storage, or even individual application capabilities, with the possible exception of limited user-specific application configuration settings."

Worth noting is the fact that nowhere here, and in fact nowhere in the document, is mention made of anything being at a remote location. In fact, the NIST definition of cloud computing doesn't care where the infrastructure is located. Of greater focus is the fact that the underlying technology is obscured from the user via the abstraction layer. In other words, the user just uses the resources to get their work done. One could operate cloud servers on one's own premises were they to wish to.

The Most Important Layer in the Cloud Model is Often Overlooked

In 1983, the International Standards Organization (ISO) introduced a useful model for networked computing called the Open Systems Interconnection (OSI) model consisting of seven sequential layers that networks go through "from the ground up," literally..

Moving outward from the user, data is entered into the network through software running on the **Application** layer. This application is running on a device-based operating system at the **Presentation** layer which is signed in through the **Session** layer. Data is moved from that user to another destination by the **Transport** layer which uses the **Network** layer to connect to that destination. This connects to the actual network via a network interface card at the **Data-Link** layer which, finally, connects to the actual cabling and wireless infrastructure at the **Physical** layer.

Arriving at the other end, the data travels back up the seven layers to arrive at its intended destination. Each layer has its own protocols and other communication standards that govern its efficient operation.

When it comes to cloud-based networks delivering SaaS, one must add an eighth layer, the **Expense** layer.

SaaS Billing Models

One of the most frequently cited strong selling points for SaaS is flexible billing. You may choose to pay by the seat for all you can eat! Or my may choose consumption-based billing in which you only pay for those services which you actually use while you're actually using them. This can be challenging to manage, given that many companies choose to use by-the-seat for some applications, and consumption-based for others. Not only must technology expense management (TEM) track which departments are using which services based on which billing, they must also track with far greater granularity

to determine precisely who is using what services for what projects in which departments and even down to which specific users.

Mainstream SaaS

What was old is new once again. The concept of shrink-wrapped software is disappearing, replaced by streaming and other online services similar to the service bureaus of old. Soon the phrase "software-as-a-service" will become simply "software," services that you pay for by the microsecond.

The functions SaaS takes over become even more fundamental. PSTN, PBX, and other telephone services will eventually disappear, replaced by Unified-Communication-as-a-Service (UCaaS) which uses internetworking protocol (IP) to route all voice along with all video and other data types across the internet. Big benefits to users include the elimination of an entire second network, the telephone network, to monitor, manage, and spend on. Collaboration is facilitated to the point where co-operative work on documents, video conferencing, casual messaging, and more become nearly effortless.

To make this work best, companies need to begin making decisions. Who will own the actual mobile devices the users use, and who will pay for the wireless services they consume? Will these be handled as expenses, or consolidated into comprehensive programs?

How will security be enforced when software is delivered from cloud services? How will this impact data sovereignty? Will customers need to specify where in the world their data may be stored? Even now as some services are cloud-based these issues are constantly being reviewed and revised. How will that look when literally all information movement is powered outside the enterprise?

There really are no complete answers to these questions as yet, but the ubiquity of cloud and SaaS as a natural component is becoming matter-of-fact. As the concept of owning software becomes as archaic as concepts like "clockwise" and "dialing" the telephone, the most important thing we can hope to do is learn from these evolutionary processes of the past and pave the best possible path to software being a service, not a product.

calero mdsl

For more information about Calero-MDSL,
please contact us:

+1.866.769.5992

info@caleromdsl.com

www.caleromdsl.com