

EVALUATING ARCHITECTURE AND DEPLOYMENT STRATEGIES: WHICH IS BEST FOR MY ORGANIZATION?

Executive Summary

This white paper is intended to address the differences between a traditional client/server or silo deployment model versus a multi-tenant Cloud/SaaS platform. Its purpose is to provide those responsible with more information as to how each of these deployment strategies affect the end client, therefore better enabling them to make the technology decision best for their organization.

Introduction

The trend in the marketplace is for companies to have greater integration of data and information, with a broader footprint across the Internal and External enterprise environments. Historically, feature and functionality were primary considerations; today, it's architecture and deployment. As such, those responsible for deploying information technology have a strategic consideration to make which effects their organization both today and into the future. In evaluating one's business needs against what is available in the market, how does one determine an architecture and deployment solution best for them?

Before we can evaluate specific criteria, we must first examine **A)** where my organization is today from an architecture and deployment perspective; and **B)** the different types of architecture and deployment approaches (or models) most prevalent today.

A) Where is my organization at from an architecture and deployment perspective?

The chart below illustrates how information technology has evolved with regard to architecture and deployment. It is easy to see that the trend over the last 30 years is to more powerful and flexible systems, available to anyone and anywhere. In addition, it is now possible to support a single instance¹ system supporting many simultaneous clients. Architecture and rapid deployment is a key component to the degree of value which can be created for an organization seeking more leverage from technology, inclusive of cost efficiency, integration of processes and information, and collaboration inside or outside the organization.

Evolution Of Architecture and Deployment

System Type	Comments	Architecture	Deployment	1940's	50's	60's	70's	80's	90's	00's	10's
A. Silo/Pillar - Batch	Origin of modern Data Processing	Server Centric Design, Limited Human Interaction	Pre-Internet, only internal to Enterprise								
B. Silo/Pillar	Primarily within the enterprise	Single-Tenant, Client Server, operated within the Enterprise, Human interaction	Pre-Internet & Post internet, only internal to Enterprise								
C. Single-Tenant Hosted/ASP/On-Demand	Primarily within the enterprise, with Web Add-on Capability	Single-Tenant, Client Server	Post internet, hosted by 3rd Party, with Web Screens Add-on's, typically external EDI Van's								
D. Multi-Tenant/SaaS/Cloud	Many Clients Simultaneously using A Silo	Many different clients, operating within a Silo/Pillar System	Deployed Over Web, Typically External EDI Van's								
E. Multi-Tenant SaaS/Cloud Platform	Many Clients Simultaneously using the Platform	Many different clients, operating on a platform, with customized features deployed, With embedded B2B Collaboration	Deployed Over Web, with B2B Collaboration Embedded								

The trend of Multi-Tenant, SaaS/Cloud² Platforms is evidenced by several leading technology service providers across the entire spectrum of information technology:

1. Email Communications: Microsoft® Exchange and Outlook® (Single Tenant, Single Instance) to Google™ Gmail™ (Multi-Tenant, SaaS/Cloud Platform).
2. Operating Systems: Microsoft Windows® (Single Tenant, Single Instance) to Google (Multi-Tenant, SaaS/Cloud Platform).
3. Customer Relationship Management(CRM): Siebel Systems{Oracle®}, SalesLogix®{Sage}, GoldMine{©FrontRange Solutions} (Single Tenant, Single Instance) to Salesforce.com® (Multi-Tenant, SaaS/Cloud Platform).
4. Business/Office Applications: Microsoft Office®, IBM® Lotus® Notes® (Single Tenant, Single Instance) to Google's Google Apps™ and Cisco®'s WebEx® (Multi-Tenant, SaaS/Cloud Platform).

B) Two main architecture and deployment approaches most prevalent in the market today, per items B and E respectively in chart above, are:

Silo or Client/Server (Single Tenant):

A Silo/Client-Server model presumes a single client is deployed for each instance of a system. It is a Silo, and therefore unrelated to other deployments of the same software application. Typically, Silo or Client/Server models are deployed within a firewall and have little web presence. This Single Tenant approach refers to a single client installation (instance) and all of their trading partners.

SaaS/Cloud Platform (Multi-Tenant):

An Cloud/SaaS (software as a service), or Multi-Tenant Platform, is a single-instance system which supports unlimited unique clients, with unique context attributes³, with unlimited unique trading partners simultaneously on the same platform. A Multi-Tenant, SaaS/Cloud platform does not consist of a single instance of software per client, as in the Silo or Client/Server approach, but rather unlimited clients operating simultaneously on the platform with deployed features and functionality unique to that client's needs.

Evaluation Criteria

Now that we've established historical trends and defined key points of reference, let's evaluate key criteria for both architecture and deployment models in order to ascertain the differences and value/benefits of each. There are several criteria we deem important to consider when evaluating a new architecture and deployment model as it relates to Enterprise Resource Planning(ERP) and Class and Supply Chain Management(SCM) software. They include:

1. Speed and Timing
2. Compatibility
3. Cost-Effectiveness
4. Simplicity
5. Flexibility
6. Integration, Translation⁴ and B2B Collaboration⁵

1. Speed and Timing

- a. The Silo or Client/Server scenario (i.e. one shipper/channel with many trading partners) is one in which the shipper/channel itself is the foundation for the whole system. In this scenario, speed and timing are a considerable factor during initial implementation and subsequent customizations as the need may exist to deploy new hardware or operating systems; load, configure and test databases; and customize to the client's needs prior to and after initial deployment. Therefore, the client/server model may result in deployment timelines ranging from months to years for each client because a whole new instance is deployed each time for each customer.
- b. The SaaS/Cloud platform scenario is one in which the client is not the foundation for the whole system, but rather just another customized client configuration on the platform. The need to add new clients or whole facilities, for example, is simply a matter of configuration and testing. The platform scenario has the ability for rapid deployment because the base platform, database, and B2B collaboration are already established within its overall architecture.

2. Compatibility

- a. The Client/Server scenario, when deployed, must support different types of databases in the market, different hardware/servers, and simultaneously, all current software versions. Therefore, this approach may be more complex with regard to compatibility. In these scenarios, one particular client may not be current with software versions, as compared to other clients. Updates/fixes are the responsibility of the client to initiate, and can cause expense, resource time, and potential system downtime.
- b. The SaaS/Cloud platform scenario is always compatible with the last update and application version for all clients, as there is only one instance of the system requiring update and in turn, all clients are instantly on the current version. Therefore, one instance and one version to apply and support by the developer. The platform developer/service provider is responsible for compatibility with all client integrations and external systems when designing and testing updates prior to rollout.

3. Cost-Effectiveness

- a. In a Client/Server scenario, each client or ASP assumes the risk and cost associated in terms of incremental hardware, software licenses, collocation, backup/redundancy systems, internal IT labor and time. This can be very costly, which is why ERP and CRM software implementation can take a long time to realize a ROI. This holds true for Hosted/ASP models because many of these systems are still Single-Tenant, single instance systems.

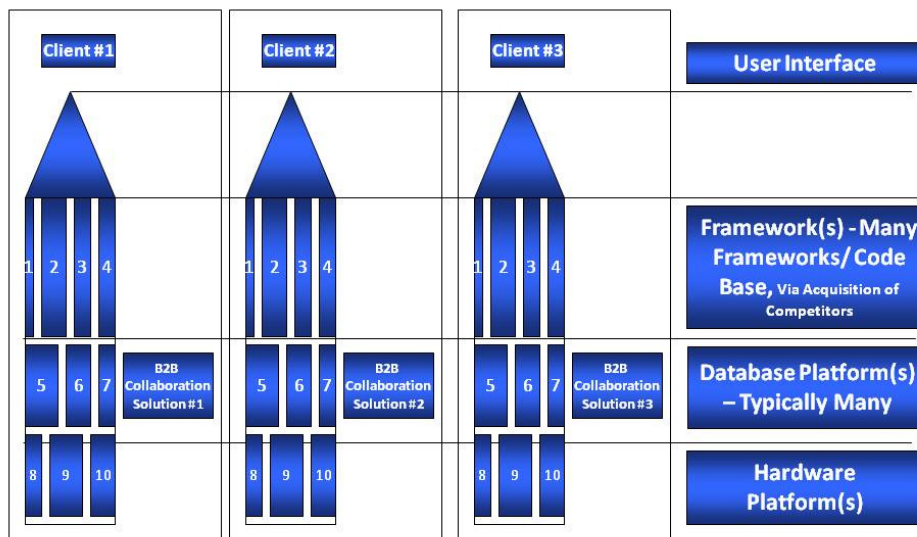
- b. By the nature of its architecture, an SaaS/Cloud platform model is the most cost-effective and efficient method to deploy information technology. To the system developer, there is only one instance to maintain, administer, backup/restore, and design against. To the client, there is no investment in hardware/software, additional IT staff, user licenses, etc. required for initial deployment or ongoing expansion. The only technology required for a client and its trading partners to interact with the system is a supported browser. To the organization's trading partners, all can adjoin any end client via a single platform login. For example, a user at a trading partner location (i.e. supplier, customer, 3PL, carrier, warehouse, etc.) can conduct business with all their end clients on the platform using a single login, providing substantial labor efficiency. B2B collaboration is also labor and cost-efficient because each trading party creates only one integration with the platform, not each client as in the Silo or Client/Server model; all subsequent trading parties added to the platform use the pre-existing integration and only require their respective data be tagged appropriately.

4. Simplicity

- a. The design of many Silo or Client/Server models is typically established the day the code is written for that system. How easily things can be changed after implementation is a consideration, especially as time passes, needs change, and the value the technology provides wanes due to its lack of simplicity and adaptability. Many Client/Server models began as a single module or application purchased by an organization/developer. Over time, the application's development team may have chosen to broaden the product offering or become an ERP; or be acquired by or merge with a competitor. The result is a large, complex code base requiring more support, longer development cycles for enhancements, more complexity for updates, and more time for testing/QA. In this scenario, the original single Client/Server model is no longer; it has evolved through development, acquisition, or merger to 1+ different Silo or Client/Server systems below the surface of the overall system - each independent of the other with the exception of a user interface and any limited integration the developer created in order to have a broader, more marketable solution. This is true of individual client instances, or an ASP/Hosted/SaaS/Cloud approach with individual client silo instances. See Figure 4.a.

Figure 4.a.

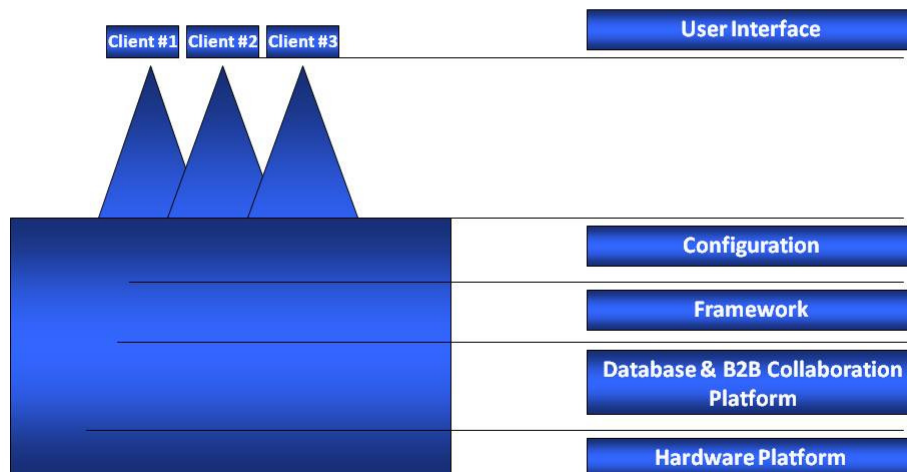
Traditional Silo or Client/Server Architecture



- b. In the Multi-Tenant, SaaS/Cloud platform scenario, though more difficult to build and start with from a technology development perspective (as compared to the Silo or Client/Server model), is more efficient, simpler to maintain, and easier to expand upon in the long run, being a single code base design. Customizations are simpler to implement and rapid to deploy because all roles and integrations of the system are native to the platform itself. With regard to client requested enhancements, in a Single Tenant Silo solution each client instance is a one-off of its own, meaning the customization is a unique development effort designed to a specific Silo (client) deployment. This isn't so for an SaaS/Cloud, Multi-Tenant Platform solution. Any new client enhancements created in a platform model are designed as generic customizations and delivered as client-requested customization via the platform's context attributes and operations. In turn, all clients on a Multi-Tenant, Cloud/SaaS platform have access to all customizations (unless exclusivity is required) because all on the platform utilize the same, current version. *See Figure 4.b.*

Figure 4.b.

Many To Many, On-Demand Platform



5. Flexibility

- a. In a Single-Tenant, Client/Server scenario, the technology foundation is centered on an entity i.e. warehouse, shipper, etc. and its direct trading partners. The technology may have many different frameworks and layers from which new development must be coded to and tested against, resulting in longer cycles for development and customization efforts. This architecture model, relative to a platform model, is less flexible in its ability to scale up and out to meet future business needs.
- b. In a Multi-Tenant, SaaS/Cloud platform scenario, all context attributes/variables which define a user are associated entirely with the login, a dependency of the overall client configuration. New development therefore, is a process of adding features, testing and deployment. Since the platform is designed with the context attributes inherent to its database, the result is an extremely flexible architecture and deployment model. And, its hardware platform, underlying the software platform, can be leveraged if additional resources are needed to support throughput by adding computing resources to the platform itself, rather than a specific client or process. This lends to a platform's ability to scale up and out efficiently.

An example of value in a platform model is the ability to centralize many different client communities⁶ in order to create a central call center, aggregated reports, centralized tariff database, or centralized inventory management and purchasing functions for different divisions with ease as the rights for specified users can include not one, but many business units/client communities.

The most evolved solution on the market today, is one that is effectively a Cloud/SaaS model, that also supports On-Premise as well. There will always be a need for both models in the market. Clients will elect to choose one over the other, but there is also client need for both at the same time, which is effectively a Simultaneous Cloud/SaaS/On-Premise model, for both production and joint development.

In addition, the Multi-Tenant, SaaS/Cloud platform can be the application in many areas of the Internal and External enterprise processes i.e. TMS, OMS, WMS, MRP, MFG etc., while it may be a wrapper around a Silo or Client/Server application for other areas at the same time. This mitigates many Client/Server challenges, and yet provides the synergies of the SaaS/Cloud platform in a cost-effective manner.

Finally, a powerful Multi-Tenant SaaS/Cloud platform can be deployed either as an SaaS/Cloud solution, or the client instance can be housed entirely within the client's firewall, and yet still be supported by the developer/service provider.

6. Integration, Translation and B2B Collaboration

- a. In a Client/Server or ASP scenario, B2B Collaboration is a third-party application bolted on; an internally built application with limited capability; an external Value Added Network(VAN); or all of the above. Translation is limited to the developer's view of what *should be* imported or exported for B2B Collaboration. Visibility and management of the data streams, both input and output, are typically non-existent via all the methods above mentioned. Integrations for Instance A, are limited to Instance A, and there is no synergy with any integrations that have already been built to any other instance of the same application with another client. If companies choose to acquire additional Client/Server applications, they will have to re-create B2B Collaboration solutions/services, often with the same trading parties, as well as integrate internal Silo or Client/Server(s) with each other.
- b. In an SaaS/Cloud Platform scenario, each trading party integrates to the platform once, regardless of the number of clients that require data streams. Administration of the data streams is provided via web interface to the data itself in real-time. Any type of data/translation is acceptable because an SaaS/Cloud Platform's B2B Collaboration solution is in effect a generation evolved beyond an EDI VAN, rather than an add-on solution to a Client/Server application. It is a fully scalable solution up and out. The Platform's B2B Collaboration solution can also serve as the client's Silos B2B Collaboration solution at the same time via single-point integration to the Platform, alleviating the complexity of individual integrations for each Silo and each trading party.

A critical component to an information technology solution today, is its ability to support any type of B2B collaboration. In addition to Web interfaces, organizations must have a collaborative B2B solution in order to create efficiencies and maximize value with their trading parties(Customers, Suppliers, 3rd Parties, etc.) and peripherals (RFID, Barcode Scanning, production line equipment, scales, time clocks, etc) . Therefore, in order for a platform model to be successful, it must support all translation and communication protocols.⁴ In a platform model, B2B collaboration is an integral component, meaning for example the not just order information but complete real-time EDI data managed within the platform itself. In a client/server model, B2B collaboration is typically facilitated by a VAN, meaning the data

flowing through it is not visible within the client/server or silo, and typically not even via EDI VAN, and cannot be managed by the end user.

Conclusions

So, can an organization have the best of both worlds? What they have today and a look to the future? An important consideration is that many organizations have legacy systems. Perhaps a third option is to continue use of current Client/Server or silo solutions of value, and migrate to a platform model which can wrap around Client/Server legacy solutions. This evolution to new technology can reap the benefits of a platform model without having to abandon all Client/Server solutions critical to a business's operations.

The need is for comprehensive technology that retains customers, builds business partners, and grows businesses beyond what is seen today. A primary consideration to architecture and deployment, and secondary features and functionality strategy chosen today must account for this, and be able to address future needs across industries. With rapid changes in technology combined with an organization's need to be responsive to changing market conditions, the trend toward a platform strategy may be here to serve as a single point for all business transactions. Yet, the final iteration of architecture and deployment is one deployed either inside or outside the firewall, with no difference from the client's perspective. The challenge for organizations is to consider all the criteria, and find the best of the best solution for their business today, with a look toward tomorrow.

About NTE

NTE LLC delivers Simultaneous Cloud/SaaS/On-Premise technology solutions to improve supply chain efficiencies globally across all industries. The NTE Solutions - Order Management, Warehouse Management, Transportation Management, Materials Requirements Planning, Document generation imagines and storage, Manufacturing Process and B2B Collaboration - provide retailers, manufacturers, distributors, non-profits/disaster relief, construction, third-party and fourth-party logistics providers with the ability to improve decision-making, reduce costs, and increase customer service across their internal and external enterprise. Through innovation and leading-edge technology, NTE provides a complete platform for ERP and SCM planning and execution, and managing B2B business processes. For more information, visit www.nte.com.

Explanatory Notes

¹ An *instance* represents a single installation of a system. In the Client/Server or Silo approach, it means a single installation of a Single Tenant software on a server(s) for a single client to function. In the Platform approach, the instance is the single installation of the platform (Multi-Tenant) where all clients function. (Page 1)

² *Cloud* computing is computation, software, data access, and storage devices that do not require end-user knowledge of the physical location and configuration of the system that delivers the services. (Page 2)

³ A *context attribute* is a variable in the software the platform uses that tells the platform how to uniquely behave for each user login, relative to a client's overall deployment features. These attributes or variables are stored in the platform database itself, not the software application. (Page 2)

⁴ *Translation* includes electronic communications (EDI, XML, and ERP Proprietary) to personal communications (text, email, PDF, Excel). *Communication protocols* include FT, AS2, HTTP, HTTPS, email, VPN, etc.) (Page 3; page 6)

⁵ *B2B Collaboration* is computer to computer, business to business communication over the internet via EDI, XML, etc. A Value Added Network (VAN) is commonly used as add-ons to a client/server or silo solution to provide integration and B2B collaboration. (Page 3)

⁶ *Client Communities* are defined as a customer and their trading partners. (Page 6)